

# Preface

## What Is Refinement?

Refinement is one of the cornerstones of a formal approach to software engineering. The traditional purpose of refinement is to show that an implementation meets the requirements set out in an initial specification. Armed with such a methodology, program development can then begin with an abstract specification and proceed via a number of steps, each step producing a slightly more detailed design which is shown to be a refinement of the previous specification.

In order to help verify refinements, different notations use different approaches, and in Z, Object-Z and other state-based languages, the techniques of downward and upward simulations emerged as the standard method of verifying refinements. However, modern specification languages and practices offer challenges, and variations to the model of computation, such as partial specifications, informal specifications, and combinations of state-based and behavioural (e.g. process-algebraic) styles. These required generalisations of the existing formal basis, and have led to new applications. The purpose of this book is to bring together that work in one volume.

## Z and Object-Z

Refinement has been studied in many notations and contexts, however in this book we will concentrate on Z and Object-Z.

Z is a formal specification language, which uses mathematical notation (set theory and logic) to describe the system under consideration. Z, along with Object-Z, (Event-)B, ASM, Alloy, and VDM, is often called a state-based language because it specifies a system by describing the states it can be in, together with a description of how the state evolves over time. In effect a specification builds a model of the system.

Object-Z is an object-oriented extension of Z, which adds the notion of class as an additional structuring device, enabling an Object-Z specification to represent a number of communicating objects.

The importance of Z for refinement is that Z specifications are abstract and often loose, allowing refinement to many different implementations. In this book we discuss some of the techniques for doing so.

## Structure of the Book

The book is structured into a series of parts, each focusing on a particular theme.

In Part I we provide an introduction to data refinement, and look at its applicability in Z. This leads to the formalisation of both downward and upward simulations in the Z schema calculus. We then discuss how refinements can be calculated and look at the application of this work to the derivation of tests from formal specifications. We also show how we can derive a single complete simulation rule by using possibility mappings.

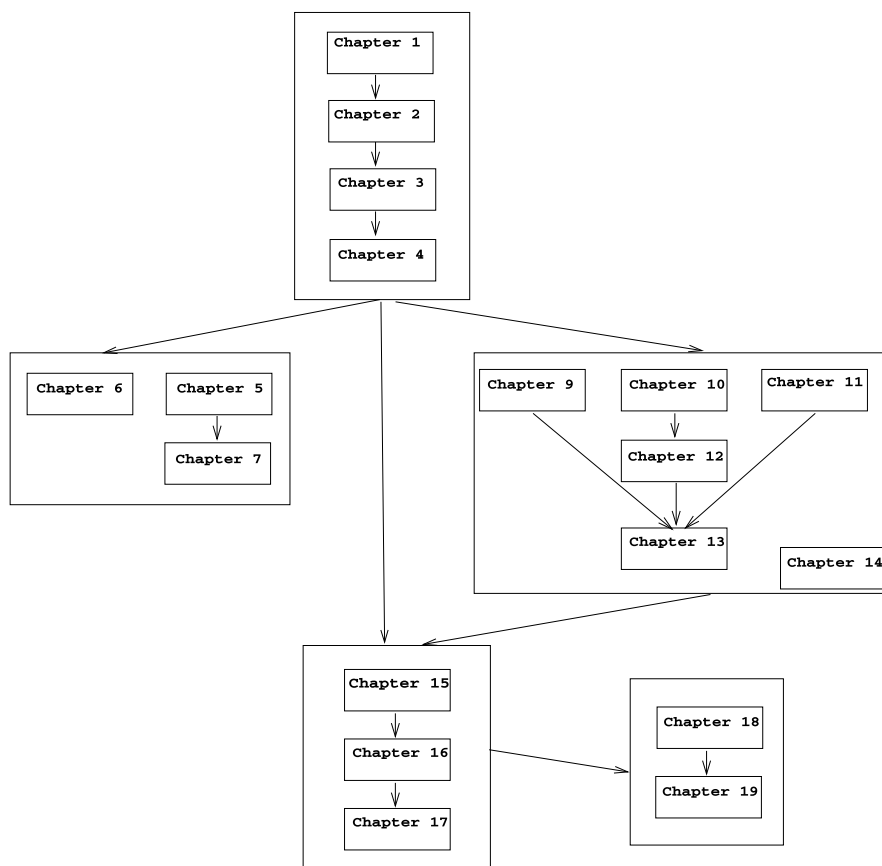
Part II looks at generalisations of refinement which allow refinement of the interface and atomicity of operations. The standard presentation of refinement requires that the interface (i.e., input/output) of an operation remains unchanged, however for a variety of applications this is unsuitable and there is a need for more general refinements which this part discusses. The refinement of an interface is intimately tied up with the problem of non-atomic refinement, i.e., where we decompose a single abstract operation into a number of concrete operations upon refinement. This part discusses this problem in some detail, presenting refinement rules for specifications containing internal operations, as well as the general approach to non-atomic refinement in Z.

In Part III we look at how the theory of refinement can be used in an object oriented context. To do so we introduce the Object-Z specification language as a canonical example, and show how refinement between classes (and collections of classes) can be verified. We also discuss the relationship between refinement and inheritance and look at the formalisation of interface refinement and non-atomic refinement in Object-Z.

Finally, in Part IV we turn our attention to combining notions of state and behaviour when specifying and refining systems. The use of Object-Z provides a notion of state, which we combine with the notion of behaviour that is provided by the process algebra CSP.

In a concluding chapter we also briefly discuss a number of related formalisms, like ASM, B and Event-B, the refinement calculus, and VDM.

The dependencies between the chapters of the book are shown in Fig. 1.



**Fig. 1** Chapter dependencies

## Readership

The book is intended primarily as a monograph aimed at researchers in the field of formal methods, academics, industrialists using formal methods, and postgraduate researchers. By including an introduction to the languages and notations used, the book is self contained, and should appeal to all those with an interest in formal methods and formal development.

The book is also relevant to a number of courses, particularly at postgraduate level, on software engineering, formal methods, object-orientation and distributed systems.

There is an associated web page to accompany this book:

<http://www.refinenet.org.uk/refinezoz/>

## Second Edition

The first edition of this book incorporated most of our research into Z and Object-Z refinement up to that point. The major changes to this second edition are based on new research insights developed since then, particularly from our research programme on “relational concurrent refinement” as well as on action refinement and other topics. In addition to numerous small changes throughout the text, this second edition differs from the first edition in the following aspects.

- The fundamental Chaps. 3 and 4 have been extended to also include the trace refinement relation, based directly on partial relations rather than through totalisation.
- In Part II, Chap. 11 contains an updated discussion on divergence, and Chap. 12 a discussion on coupled simulations as a means to verify non-atomic refinements. In addition, Chap. 14 now contains material on approximate refinement.
- In Part III on Object-Oriented Refinement we have strengthened several specifications involving object aggregates with framing conditions.
- In Part IV on modelling state and behaviour we have added to Chap. 18 some material on differing semantics of operations and outputs and how they affect the abstraction of models written using Object-Z and CSP. In addition, Chap. 19 has been significantly revised, based on research into combining state-based and behavioural specification methods in the last ten years. In particular, we give a fuller account of the relationship between relational refinement and various models of refinement in CSP.
- Bibliographic notes at the end of each chapter have been extended with more recent research.
- The concluding chapter has been extended with discussion of ASM and Event-B.

## Acknowledgements

Fragments of Chap. 3 and some later chapters were reprinted from: *Proceedings of Mathematics of Program Construction 2000*, R.C. Backhouse and J.N. Oliveira (Eds.), Lecture Notes in Computer Science 1837, E.A. Boiten and J. Derrick, “Liberating data refinement”, pp. 144–166, copyright (2000), with kind permission of Springer Science and Business Media.

Fragments of Chap. 5 were reprinted from: *Information and Software Technology*, Volume 41, J. Derrick and E.A. Boiten, “Calculating upward and downward simulations of state-based specifications”, pp. 917–923, copyright (1999), with permission from Elsevier Science.

Fragments of Chap. 7 are reproduced with permission from: *Software Testing, Verification and Reliability*, Volume 9, J. Derrick and E.A. Boiten, “Testing refinements of state-based formal specifications”, pp. 27–50, 1999 © John Wiley & Sons Limited.

Fragments of Chap. 8 were reprinted from: *The Journal of Logic and Computation*, Volume 10, Number 5, J. Derrick, “A single complete simulation rule in Z”, pp. 663–675, copyright (2000), by permission from Oxford University Press.

Fragments of Chap. 9 were reprinted from: *International Refinement Workshop & Formal Methods Pacific '98*, J. Grundy, M. Schwenke and T. Vickers (Eds.), Series in Discrete Mathematics and Theoretical Computer Science, E.A. Boiten and J. Derrick, “Grey box data refinement”, pp. 45–60, copyright (1998), with kind permission of Springer Science and Business Media.

Fragments of Chap. 11 were reprinted from: *Formal Aspects of Computing*, Volume 10, J. Derrick, E.A. Boiten, H. Bowman and M.W.A. Steen, “Specifying and refining internal operations in Z”, pp. 125–159, copyright (1998), with kind permission of Springer Science and Business Media.

Fragments of Chap. 11 were reprinted from: *Formal Aspects of Computing*, Volume 21, E.A. Boiten, J. Derrick, and G. Schellhorn, “Relational concurrent refinement Part II: internal operations and outputs”, pp. 65–102, copyright (2009), with kind permission of Springer Science and Business Media.

Fragments of Chap. 12 were reprinted from: *FM'99 World Congress on Formal Methods in the Development of Computing Systems*, J.M. Wing and J.C.P. Woodcock and J. Davies (Eds.), Lecture Notes in Computer Science 1708, J. Derrick and E.A. Boiten, “Non-atomic refinement in Z”, pp. 1477–1496, copyright (1999), with kind permission of Springer Science and Business Media.

Fragments of Chap. 12 were reprinted from: *ZB 2003: Formal Specification and Development in Z and B, 3rd International Conference of B and Z Users*, D. Bert, J.P. Bowen, S. King and M.A. Waldén (Eds.), Lecture Notes in Computer Science 2651, J. Derrick and H. Wehrheim, “Using coupled simulations with non-atomic refinement”, pp. 127–147, copyright (2003), with kind permission of Springer Science and Business Media.

Fragments of Chap. 14 were reprinted from: *ZB 2005: Formal Specification and Development in Z and B, 4th International Conference of B and Z Users*, H. Treharne and S.A. Schneider (Eds.), Lecture Notes in Computer Science 3455, E.A. Boiten and J. Derrick, “Formal program development with approximations”, pp. 374–392, copyright (2005), with kind permission of Springer Science and Business Media.

Fragments of Chap. 18 were reprinted from: *ICFEM 2002: Formal Methods and Software Engineering, 4th International Conference on Formal Engineering Methods*, C. George and H. Miao (Eds.), Lecture Notes in Computer Science 2495, G. Smith and J. Derrick, “Abstract specification in Object-Z and CSP”, pp. 108–119, copyright (2002), with kind permission of Springer Science and Business Media.

Fragments of Chap. 19 were reprinted from: *Formal Aspects of Computing*, Volume 15, J. Derrick and E.A. Boiten, “Relational concurrent refinement”, pp. 182–214, copyright (2003), with kind permission of Springer Science and Business Media.

Many people have contributed to this book in a variety of ways. We have had numerous discussions on aspects of specification and refinement and benefited from

related work in this area with other researchers including: Ralph Back, Richard Banach, Howard Bowman, Michael Butler, David Cooper, Jim Davies, Brijesh Dongol, Steve Dunne, Clemens Fischer, Lindsay Groves, Ian Hayes, Martin Henson, Karl Lermer, Peter Linington, Christie Marr (née Bolton), Carroll Morgan, Ernst-Rüdiger Olderog, Steve Reeves, Gerhard Schellhorn, Steve Schneider, Kaisa Sere, Maarten Steen, Susan Stepney, Ketil Stølen, David Streader, Ramsay Taylor, Marina Waldén, and Jim Woodcock.

Thanks are especially due to Behzad Bordbar, Rob Hierons, Ralph Miarka, Graeme Smith, Chris Taylor, Ian Toyn, Helen Treharne, and Heike Wehrheim, for their careful reading of earlier drafts of the first edition and helpful suggestions. Thanks also to the reviewers of the first edition, in particular Perdita Stevens, for their useful comments.

We also wish to thank staff at Springer UK for their help and guidance in preparing the first and second editions of this book: Rosie Kemp, Steve Schuman, Beverley Ford, and Ben Bishop.

Finally, thanks to our friends and families who have provided love and support throughout this project. We would not have completed this book without them.

Refinement in Z and Object-Z  
Foundations and Advanced Applications  
Derrick, J.; Boiten, E.A.  
2014, XVIII, 492 p., Hardcover  
ISBN: 978-1-4471-5354-2