Annotated Z Bibliography

Jonathan Bowen
*Oxford University Computing Laboratory,*
*Wolfson Building, Parks Road, Oxford OX1 3QD, UK.*

Susan Stepney, Rosalind Barden
*Logica Cambridge Limited,*
*Betjeman House, 104 Hills Road, Cambridge, CB2 1LQ, UK.*

1 Introduction

This annotated Z bibliography contains a selected list of some pertinent publications for Z users. Most of those included are readily available, either as books or in journals. A few unpublished items have been included, where they are particularly relevant and can be obtained reasonably easily.

Some references are accompanied by an annotation. This may include a contents list (of a book), a list of the titles of Z related papers (in a collection) with cross references to the full details, or a summary of the work.

2 Cross references

The bibliography in the last section lists all references in alphabetical order by author. In this section papers are arranged by subject (with authors and brief details of the subject matter), together with cross references to the full details in the bibliography.

2.1 Management, style, and method

For justifications for using formality, and quick introductions to Z, see:

[63, 199] Cohen/McDermid. Justification of formal methods and notations
[204] Meyer. On formalism in specifications
[269] Spivey. Introduction to Z
[305, 306] Wing. General introduction to formal methods including Z
[311] Woodcock. Structuring specifications

For discussion about using formal methods in practice, see:
[22, 45, 46, 118, 201] Barroca/McDermid, Bowen/Stavridou and Gerhart et al.
  Formal methods and safety-critical systems
[116] Gerhart. Applications of formal methods
[124, 40, 41, 42] Hall and Bowen/Hinchev. Myths and guidance about formal
  methods
[319] Worden. Fermenting and distilling ideas

Educational issues are presented and discussed in:

[68] Cooper. Educating management
[113] Garlan. Effective integration of formal methods into a professional master of
  software engineering course
[252] Saiedian. The mathematics of computing
[290] Swatman. Educating information systems professionals

Various papers describing good specification style are:

  concentrates on making large specifications more understandable
[92] Duke. Enhancing structure
[122, 123] Gravell. Minimization in specification/design and what makes a good
  specification
[189] Macdonald. Usage and abusage

Much work has been done on attempting to integrate Z with traditional structured analysis
methods. Some of this is described in:

[14] Aujla et al. A rigorous review technique
[55] Bryant et al. Structured methodologies and formal notations
[90] Draper. Z and SSADM
[83] Giovanni and Iachini. HOOD and Z
[239, 240, 238] Polack et al. SAZ Method – Structured Analysis and Z
[165, 166] Josephs and Redmond-Pyle. Entity-relationship models, structured
  methods, and Z
[256] Semmens and Allen. Yourdon and Z
[257] Semmens et al. Integrated structured analysis and formal specification
  techniques
[299] van Hee et al. Petri nets and Z

Other work towards the development of a ‘method’ for Z itself include:

[127] Hall and McDermid. Towards a Z method using axiomatic specification in Z
  (using order sorted algebra and OBJ3 in particular)
[219] Neilson. A rigorous development method from Z to C
[308] Wood. A practical approach using Z and the refinement calculus

The application of metrics to formal specifications has been studied:

[302, 17] Whitty and Bainbridge et al. Structural metrics for Z specifications

A formal specification in Z can be useful for deciding test cases, etc. Work on testing is reported in:

[8, 9] Ammann and Offutt. Functional and test specifications based on the category-partition method
[60] Carrington and Stocks. Formal methods and software testing
[77] Cusack and Wezeman. Deriving tests for objects specified in Z
[128] Hall. Testing with respect to formal specification
[133] Hayes. Specification directed module testing
[281] Stocks. Applying formal methods to software testing
[282] Stocks and Carrington. Deriving software test cases from formal specifications
[283, 284] Stocks and Carrington. Test templates: a specification-based testing framework and case study

2.2 Application areas

Surveys of formal methods, including Z users, are reported in:

[19] Barden et al. Use of Z (in the UK)
[71, 72, 73, 117, 118] Craigen et al. An international survey of major industrial formal methods applications, including a number using Z

One of the high profile users of Z is IBM UK Laboratories at Hursley for specification and development of the CICS transaction processing system. General descriptions of the CICS project include:

[64] Collins et al. Introducing formal methods: the CICS experience with Z
[147] Houston and King. CICS project report
[228] Nix and Collins. Use of software engineering and Z in the development of CICS
[235] Phillips. CICS experience throughout the lifecycle
[320] Wordsworth. The CICS Application Programming Interface (API) definition

Specifying secure systems is discussed in:

[160] Jones. Verification of critical properties
[267] Smith and Keighley. A secure transaction mechanism (SWORDB secure DBMS)
Not all Z specifications are of software systems. Much interesting and important work has been done on formally specifying hardware, including microprocessors. The Inmos T800 transputer Floating Point Unit microcode development is a major real example where formal methods have saved time by reducing the amount of testing needed.

[196, 198, 197, 261, 260] May, Shepherd et al. T800 transputer FPU development

More technical papers on hardware applications (including embedded software) are:

[21] Barrett. A floating-point number system (IEEE standard)
[29, 30, 33] Bowen. Microprocessor instruction sets (Motorola M6800 and transputer)
[81, 82, 114, 135] Delisle/Garlan and Hayes. Oscilloscopes, including reuse of specifications
[167, 168] Kemp. Viper microprocessor
[266] Smith and Duke. Cache coherence protocol (in Object-Z)
[271] Spivey. Real-time kernel

Communications systems and protocols are specified in:

[29] Bowen et al. Network services via remote procedure calls (RPC)
[57] Butler. Service extension (PABX)
[98] Duke et al. Object-oriented protocol specification (mobile phone system, in Object-Z)
[121] Gotzhein. Open distributed systems
[132] Haughton. Safety and liveness properties of communication protocols
[194] Mataga and Zave. Formal specification of telephone features
[236] Pilling et al. Inheritance protocols for real-time scheduling
[293] Till and Potter. Gateway functions within a communications network
[323] Zave and Jackson. Specification of switching systems (PBX)

The following papers describe the use of Z for various graphics applications, standards (especially GKS), and human computer interfaces:

[10, 11] Arnold et al. Configurable models of graphics systems (GKS)
[31, 34] Bowen. Formal specification of window systems (X in particular)
[53] Brown and Bowen. An extensible input system for Unix
[88] Dix et al. Human-Computer Interaction (HCI)
[91] Duce et al. Formal specification of Presentation Environments for Multimedia Objects (PREMO)
[156] Johnson. Specification and prototyping of concurrent multi-user interfaces
[157] Johnson and Harrison. Declarative graphics and dynamic interaction
[214, 215] Narayana and Dharap. Formal specification of a Look Manager and a dialog system
[217] Nehlig and Duce. Formal specification of the GKS design primitive
[287] Sufrin. Formal specification of a display-oriented editor
[286, 289] Sufrin and He. Effective user interfaces and interactive processes
[294] Took. A formal design for an autonomous display manager

An important application area for formal methods is safety-critical systems where human lives may depend on correctness of the system.

[22, 45, 46, 118, 201] Barroca/McDermid, Bowen/Stanridou and Gerhart et al. Surveys covering formal methods and safety-critical systems
[35, 46, 39] Bowen et al. Safety-critical systems and standards

Some examples of the application of Z to safety-critical systems are:

[172] Knight and Kienzle. Using Z to specify a safety-critical system in the medical sector

Other papers describing a variety of applications using Z include:

[1] Abowd et al. Software architectures
[32] Bowen. A text formatting tool
[36, 178] Bowen, Lano and Breuer. Reverse engineering
[54] Brownbridge. CASE toolset (for SSADM)
[56] Butcher. A behavioural semantics for Linda-2
[70] Craig. Specification of advanced AI architectures
[78, 79] de Barros and Harper. Formal specification and derivation of relational database applications
[104] Fenton and Mole. Flowgraph transformation
[216] Nash. Large systems
[248] Reizer et al. Requirements specification of a proposed POSIX standard
[276] Stepney. High integrity compilation
[288] Sufrin. A Z model of the Unix make utility
[315] Woodcock et al. Formal specification of the UK Defence Standard 00-56
[325] Zhang and Hitchcock. Designing knowledge-based systems and information systems

2.3 Textbooks on Z

[139] Hayes et al. Specification case studies (the first book on Z, now in its 2nd edition, containing an excellent selection of example Z specifications)
[149] Imperato. An introduction to Z
[150] Ince. An introduction to discrete mathematics and formal system specification (2nd edition)
[183] Lightfoot. Formal specification using Z
[229] Norcliffe and Slater. Mathematics of software construction
[241] Potter, Sinclair and Till. An introduction to formal specification and Z (a popular first textbook on Z)
[247] Ratcliff. Introducing specification using Z

A video course is also available [230, 231].

2.4 Language details

Z’s syntax, semantics and mathematical toolkit are being internationally standardized under ISO/IEC JTC1/SC22. A draft version of the standard is available:

[52] Brien and Nicholls. Z Base Standard, version 1.0

The definition of the Z syntax and mathematical toolkit used by many practitioners is:


More technical works describing Z’s formal semantics are:

[298] Diepen and van Hee. The link between Z and the relational algebra
[112] Gardiner et al. A simpler semantics
[268] Spivey. Understanding Z
[274] Spivey and Sufrin. Type inference

Z is often compared and contrasted with VDM (Vienna Development Method). The following papers show the cross-fertilization and comparisons between the two:

[26] Bera. Structuring for the VDM specification language, in response to the Z schema notation
[119] Gilmore. Correctness-oriented approaches to software development in which the Z, VDM and algebraic styles are compared
[141] Hayes et al. Understanding the differences between VDM and Z
[184, 185] Lindsay. A VDM perspective on reasoning about Z specifications and transferring VDM verification techniques to Z
[186] Lindsay and van Keulen. Case studies in the verification of specifications in VDM and Z

Reasoning about Z specifications is addressed in:
[208] Morgan and Sanders. Laws of the Logical Calculi
[309] Woodcock. Calculating properties (preconditions)
[314, 193] Woodcock/Brien and Martin. $\mathcal{W}$, a logic for Z.

Work on refining Z-like specifications towards an implementation (see also section 2.5) includes:
[21] Barrett. Refinement from Z to microcode via Occam
[16] Bailes and Duke. Class refinement
[23] Baumann. Z and natural semantics programming language theory for algorithm refinement
[85, 86] Diller. Hoare logic and Part II: Methods of Reasoning
[119] Gilmore. Correctness-oriented approaches to software development (Z, VDM and algebraic styles are compared)
[144] He et al. Foundations for data refinement
[155] Jacob. Varieties of refinement
[163] Josephs. Data refinement calculation for Z specifications
[171] King and Sørensen. Specification and design of a library system
[177, 180] Lano and Haughton. Reasoning and refinement in object-oriented specification languages
[190, 191, 192] Mahoney/Hayes et al. Timed refinement
[218, 219] Neilson. Hierarchical refinement of Z specifications and a rigorous development method from Z to C
[258] Sennett. Using refinement to convince (pattern matching in ML)
[259] Sennett. Demonstrating the compliance of Ada programs with Z specifications
[289] Sufrin and He. Specification, analysis and refinement of interactive processes
[304] Whysall and McDermid. Object-oriented specification and refinement
[307] Wood. Software refinery
[312] Woodcock. Implementing promoted operations in Z

The 'refinement calculus’ approach to refinement is espoused in:
[170] King. Z and the refinement calculus
[210] Morgan and Vickers. Collected research papers
[308] Wood. A practical approach using Z and the refinement calculus
The related B-Method, with associated B-Tool, B-Toolkit and Abstract Machine Notation (AMN), have been developed by Abril et al., also the progenator of Z:

[80] Dehbonei and Mejia. Use of B in the railways signalling industry
[87] Diller and Docherty. A comparison of Z and Abstract Machine Notation
[220, 221] Neilson and Prasad. ZedB (a prototype B-based proof tool)
[249] Ritchie et al. Experiences in using the Abstract Machine Notation in a GKS graphics standard case study
[285] Storey and Haughton. A strategy for the production of verifiable code using the B-Method

Execution of formal specifications is a subject of perennial debate. See:

[140] Hayes and Jones. Specifications are not (necessarily) executable

A retort may be found in:

[111] Fuchs. Specifications are (preferably) executable

Animating Z specifications is discussed in:

[51] Breuer and Bowen. Correct executable semantics for Z using abstract interpretation, including an informal taxonomy of approaches
[84] Dick et al. Computer aided transformation of Z into Prolog
[89] Doma and Nicholl. EZ: automatic prototyping
[120] Goodman. Animating Z specifications in Haskell using a monad
[131] Hasselbring. Animation of Object-Z specifications with a set-oriented prototyping language
[158] Johnson and Sanders. Functional implementations (Z to Miranda)
[187] Love. Animating Z specifications in SQL
[280] Stepney and Lord. An access control system (Z to Prolog)
[296] Valentine. Z−−, an executable subset of Z
[300] West and Eaglestone. Two approaches to animation (Z to Prolog)

Specific language features are addressed in:

[13, 263] Arthan and Smith. Free types in Z (including recursion)
[134] Hayes. A generalization of bags
[136] Hayes. Interpretations of schema operators
[137] Hayes. Multi-relations in Z (a cross between multi-sets and binary relations)
[188] Lupton. Promotion and forward simulation
[310, 312] Woodcock. Proof rules for promotion and implementing promoted operations
Some research has been undertaken in using and adapting Z to model concurrent systems:

[66] Coombes and McDermid. Specifying distributed real-time systems  
[101, 102] Evans. Visualising, specifying and verifying concurrent systems using Z  
[156] Johnson. Applying temporal logic to support the specification and prototyping of concurrent multi-user interfaces  
[175] Lamport. TLZ: Temporal Logic of Actions (TLA) and Z  
[215] Narayana and Dharap. Invariant properties in a dialog system  
[255] Schuman et al. Object-oriented process specification  

In particular, there has been some work on combining Z and CSP (Communicating Sequential Processes), a formal process model with associated algebra for concurrent systems:

[164] Josephs. Theoretical work on a state-based approach to communicating processes  

Researchers have also considered modelling and reasoning about real-time systems, for example, by combining temporal logic with Z.

[66] Coombes and McDermid. Specifying temporal requirements for distributed real-time systems  
[99] Duke and Smith. Temporal logic and Z specifications  
[100] Engel. Specifying real-time systems with Z and the Duration Calculus  
[105] Fergus and Ince. Model logic and Z specifications item[[106]] Fidge. Specification and verification of real-time behaviour using Z and RTL  
[143] He Jifeng et al. Provably correct systems, including the use of Duration Calculus with schemas for structuring  
[156] Johnson. Applying temporal logic to support the specification and prototyping of concurrent multi-user interfaces  
[175] Lamport. TLZ: Temporal Logic of Actions (TLA) and Z  
[190, 191, 192] Mahoney/Hayes et al. Timed refinement  
[215] Narayana and Dharap. Invariant properties in a dialog system using Z and temporal logic  
[236] Pilling et al. Inheritance protocols for real-time scheduling  
[264] Smith. An object-oriented approach including a formalization of temporal logic history invariants
2.5 Conferences proceedings

Regular Z User Meetings are organized by the Z User Group and have had published proceedings since the 4th meeting:

[38] Bowen and Hall. 8th Z User Meeting, Cambridge, 1994
[43] Bowen and Hinchey. 9th Z User Meeting, Limerick, 1995

The annual Refinement Workshop is organized by BCS-FACS. Papers cover a variety of refinement techniques from specification to code, and include some Z examples.


FME Symposia are held every 18 months, organized by Formal Methods Europe. These grew out of the the later VDM-Europe conferences which included papers on Z:

[28] Bloomfield et al. VDM’88, Dublin
[27] Bjørner et al. VDM’90, Kiel
[242, 243] Prehn and Toetenel. VDM’91, Noordwijkerhout
[316] Woodcock and Larsen. FME’93, Odense
[213] Naftalin et al. FME’94, Barcelona

2.6 Tools

The ZIP Project tools catalogue lists some tools that support formatting, checking and proof of Z specifications:

[234] Parker. Z tools catalogue

Details of individual tools may be found in:

[12] Arthan. A proof tool based on HOL which grew into ProofPower (see below)
[37] Bowen and Gordon. Z and HOL (a tool based on higher order logic)
[110] Flynn et al. Formaliser (editor and type-checker)
[161] Jones. ICL ProofPower (a commercial tool based on HOL)
[162, 295] Jordan et al. CADIZ (formatter and type-checker)
[220, 221] Neilson and Prasad. ZedB (a prototype B-based schema expansion and precondition calculator tool)
2.7 Object-Oriented Approaches

There has been much work recently to enhance Z with some of the structuring ideas from object-orientation. Overviews and comparisons can be found in:

[58] Carrington. ZOOM workshop report
[181] Lano and Haughton. Object-oriented specification case studies, many using extensions to Z
[278, 279] Stepney et al. Collected papers and a survey on object-orientation in Z

Object-Z is the best-documented and probably most widely used object-oriented extension to Z. The definitive description of the language is:


Other Object-Z papers include:

[59] Carrington et al. Object-Z: an object-oriented extension to Z
[91] Duce et al. Formal specification of Presentation Environments for Multimedia Objects (PREMO)
[93] Duke and Duke. Towards a semantics
[98] Duke et al. Object-oriented protocol specification (mobile phone system)
[131] Hasselbring. Animation with a set-oriented prototyping language
[244] Rafsanjani and Colwill. From Object-Z to C++
[266] Smith and Duke. Cache coherence protocol

Descriptions of other object-oriented approaches in conjunction with Z may be found in:

[75] Cusack. Inheritance in object-oriented Z
[125, 126] Hall. A specification calculus for object-oriented systems and class hierarchies in Z
[129] Hammond. Producing Z specifications from object-oriented analysis
[176, 179, 36] Lano/Haughton et al. Z++: an object-orientated extension to Z
[254, 255] Schuman, Pitt et al. Object-oriented subsystem and process specification
[301] Wezeman and Judge. Z for managed objects
[303, 304] Whysall and McDermid. Object-oriented specification and refinement
3 On-line information

The BibTeX source for this bibliography and related information is available on-line via the World-Wide Web under the following URL (Uniform Resource Locator):

http://www.comlab.ox.ac.uk/archive/z/bib.html

The bibliography is searchable. The user may provide a regular expression or select from a number of predefined keywords. Hyperlinks are included to documents that can be accessed on-line.

4 Acknowledgements

We would like to thank all those who suggested references for inclusion in this bibliography. It has been adapted from the ZIP project bibliography [277], the on-line Z bibliography held at the Oxford University Computing Laboratory [47], including more recent additions.

Jonathan Bowen is funded by the UK Engineering and Physical Sciences Research Council (EPSRC) grant no. GR/J15186.

5 Bibliography


[5] J.-R. Abrial. *The B-Book*. Cambridge University Press, 1995. To appear. Contents: Mathematical reasoning; Set notation; Mathematical objects; Introduction to abstract machines; Formal definition of abstract machines; Theory of abstract machines; Construction large abstract machines; Example of abstract machines; Sequencing and loop; Programming examples; Refinement; Construction large software systems; Example of refinement; Appendices: Summary of the most current notations; Syntax; Definitions; Visibility rules; Rules and axioms; Proof obligations.


The paper describes a general framework for the formal specification of modular graphics systems, illustrated by an example taken from the Graphical Kernel System (GKS) standard.


13


This paper presents a formalization of the IEEE standard for binary floating-point arithmetic in Z. The formal specification is refined into four components. The procedures presented form the basis for the floating-point unit of the Inmos IMS T800 transputer. This work resulted in a joint UK Queen’s Award for Technological Achievement for Inmos Ltd and the Oxford University Computing Laboratory in 1990. It was estimated that the approach saved a year in development time compared to traditional methods.


The 3rd VDM-Europe Symposium was held at Kiel, Germany, 17–21 April 1990. A significant number of papers concerned with Z were presented [61, 93, 114, 83, 121, 125, 170, 253, 275, 298, 318].


The 2nd VDM-Europe Symposium was held at Dublin, Ireland, 11–16 September 1988. See [3, 26].


The Z notation is used to define the Motorola M6800 8-bit microprocessor instruction set.


Three existing window systems, X from MIT, WM from Carnegie-Mellon University and the Blit from AT&T Bell Laboratories are covered.


This article is part of a special issue on Formal aspects of microprocessor design, edited by H. S. M. Zedan. See also [260].


This paper asks whether window management systems would not be better specified through a formal methodology and gives examples in Z of X11.


Proceedings of the Eighth Annual Z User Meeting, St. John’s College, Cambridge, UK. Published in collaboration with the British Computer Society. For individual papers, see [23, 37, 48, 47, 51, 60, 62, 87, 100, 102, 113, 126, 127, 129, 131, 175, 194, 238, 265, 301, 315, 319]. The proceedings also includes an Introduction and Opening Remarks, a Select Z Bibliography [48] and a section answering Frequently Asked Questions [47].


This article deals with further myths in addition to those presented in [124].


Proceedings of the Seventh Annual Z User Meeting, DTI Offices, London, UK. Published in collaboration with the British Computer Society. For individual papers, see

The proceedings also includes an Introduction and Opening Remarks, a Select Z Bibliography and a section answering Frequently Asked Questions.


A survey on the use of formal methods, including B and Z, for safety-critical systems. A previous version is also available as Oxford University Computing Laboratory Technical Report PRG-TR-5-92.


16

This is the first publicly available version of the proposed ISO Z Standard. See also [273] for the current most widely available Z reference manual.


Available from EUUG Secretariat, Owles Hall, Buntingford, Hertfordshire SG9 9PL, UK.


This paper records the activities of a workshop on Z and object-oriented methods held in August 1992 at Oxford. A comprehensive bibliography is included.


Also available as Technical Report 94-4, Department of Computer Science, University of Queensland, 1994.


This book contains two rather large (and relatively complete) specifications of Artificial Intelligence (AI) systems using Z. The architectures are the blackboard and Cassandra architectures. As well as showing that formal specification can be used in AI at the architecture level, the book is intended as a case-studies book, and also contains introductory material on Z (for AI people). The book assumes a knowledge of Z, so for non-AI people its primary use is for the presentation of the large specifications. The blackboard specification, with explanatory text, is around 100 pages.


Unlike most work on the application of formal methods, this research uses formal methods to gain insight into system architecture. The context for this case study is electronic instrument design.


This book offers a comprehensive tutorial to Z from the practical viewpoint. Many natural deduction style proofs are presented and exercises are included. Z as defined in the 2nd edition of The Z Notation [273] is used throughout.

Contents: Tutorial introduction; Methods of reasoning; Case studies; Specification animation; Reference manual; Answers to exercises; Glossaries of terms and symbols; Bibliography.


PREMO (Presentation Environments for Multimedia Objects) is a work item proposal by the ISO/IEC JTC1/SC24 committee, which is responsible for international standardization in the area of computer graphics and image processing.


The most complete (and currently the standard) reference on Object-Z. It has been reprinted by ISO JTC1 WG7 as document number 372. A condensed version of this report was published as [96].


This is an introduction to a special issue on Formal Methods with an emphasis on Z in particular. It was published in conjunction with special Formal Methods issues of IEEE Transactions on Software Engineering and IEEE Computer. See also [82, 124, 214, 271, 305].


Several commercial and exploratory cases in which Z features heavily are briefly presented on page 24. See also [173].

This PhD thesis provides a critical evaluation of Z, VDM and algebraic specifications.


Formal methods are difficult, expensive, and not widely useful, detractors say. Using a case study and other real-world examples, this article challenges such common myths.


This is a revised edition of the first ever book on Z, originally published in 1987; it contains substantial changes to every chapter. The notation has been revised to be consistent with The Z Notation: A Reference Manual by Mike Spivey [273]. The CAVIAR chapter has been extensively changed to make use of a form of modularization.
Divided into four sections, the first provides tutorial examples of specifications, the second is devoted to the area of software engineering, the third covers distributed computing, analyzing the role of mathematical specification, and the fourth part covers the IBM CICS transaction processing system. Appendices include comprehensive glossaries of the Z mathematical and schema notation. The book will be of interest to the professional software engineer involved in designing and specifying large software projects.

The other contributors are W. Flinn, R. B. Gimson, S. King, C. C. Morgan, I. H. Sørensen and B. A. Sufrin.


Also available as Technical Report UMCS-93-8-1, Department of Computer Science, University of Manchester, UK, 1993.


Contents: Introduction; Set theory; Logic; Building Z specifications; Relations; Functions; Sequences; Bags; Advanced Z; Case study: a simple banking system.


A description of a \textit{Z} style option ‘\texttt{oz.sty}’, an extended version of Mike Spivey’s ‘\texttt{zed.sty}’ [270], for use with the \texttt{LaTeX} document preparation system [174]. It is particularly useful for printing Object-Z documents [59, 93].


Guest editors’ introduction to a special issue of \textit{IEEE Software} on Safety-Critical Systems. A short section on formal methods mentions several \textit{Z} books on page 18. See also [118].


\textit{Z} specifications may be produced using the document preparation system \texttt{LaTeX} together with a special \texttt{LaTeX} style option. The most widely used style files are \texttt{fuzz.sty} [272], \texttt{zed.sty} [270] and \texttt{oz.sty} [169].


Contents: Chapters introducing object oriented methods, object oriented formal specification and the links between formal and structured object-oriented techniques; seven case studies in particular object oriented formal methods, including:


A glossary, index and bibliography are also included. The contributors are some of the leading figures in the area, including the developers of the above methods and languages: Silvio Meira, Gordon Rose, Roger Duke, Antonio Alencar, Joseph Goguen, Alan Wills, Cassio Souza dos Santos, Ana Cavalcanti.


Contents: Introduction; Sets in Z; Using sets to describe a system – a simple example; Logic: propositional calculus; Example of a Z specification document; Logic: predicate calculus; Relations; Functions; A seat allocation system; Sequences; An example of sequences – the aircraft example again; Extending a specification; Collected notation; Books on formal specification; Hints on creating specifications; Solutions to exercises. Also available in French.

[185] P. A. Lindsay. On transferring VDM verification techniques to Z. In Naftalin et al. [213], pages 190–213.
Also available as Technical Report 94-10, Department of Computer Science, University of Queensland, 1994.


This paper presents a miscellany of observations drawn from experience of using Z, shows a variety of techniques for expressing certain class of idea concisely and clearly, and alerts the reader to certain pitfalls which may trap the unwary.


A specification and top-level design of a steam generating boiler system is presented as an example of the formal development of a real-time system.


A special issue on Z, introduced and edited by Prof. J. A. McDermid. See also [32, 63, 269, 311].


This book contains papers from the 1st Refinement Workshop held at the University of York, UK, 7–8 January 1988. Z-related papers include [218, 171].


This chapter contains a case study in Z, followed by a discussion of the respective trade-offs in specification between Z and VDM.


This book presents a rigorous treatment of most elementary program development techniques, including iteration, recursion, procedures, parameters, modules and data refinement.

This document records some important laws of classical predicate logic. It is designed as a reservoir to be tapped by users of logic, in system development.


This book collects together the work accomplished at Oxford on the refinement calculus: the rigorous development, from state-based assertional specification, of executable imperative code.


The workshop was held at the IBM Laboratories, Hursley Park, UK, 9–11 January 1990. See [258].


The workshop was held at Cambridge, UK, 9–11 January 1991. For Z related papers, see [16, 155, 190, 307, 312, 304].


The 2nd FME Symposium was held at Barcelona, Spain, 24–28 October 1994. Z-related papers include [41, 69, 78, 101, 103, 108, 151, 185]. B-related papers include [80, 249, 285].


A formal specification of the look manager of a dialog system is presented in Z. This deals with the presentation of visual aspects of objects and the editing of those visual aspects.


Proceedings of the Sixth Annual Z User Meeting, York, UK. Published in collaboration with the British Computer Society. For individual papers, see [13, 19, 79, 58, 85, 92, 130, 221, 225, 239, 251, 263, 291, 296, 314, 324].


Contents: Why mathematics; Getting started: sets and logic; Developing ideas:
schemas; Functions; Functions in action; A real problem from start to finish: a
drinks machine; Sequences; Relations; Generating programs from specifications:
refinement; The role of proof; More examples of specifications; Concluding remarks;
Answers to exercises.


Collegiate Crescent, Sheffield S10 2BP, UK.
Video-based Training Course on the Z Specification Language. The course consists
of 5 videos, each of approximately one hour duration, together with supporting texts
and case studies.


A record of the opening address at ZUM’89.

Aerospace, Software Technology Department, Warton PR4 1AX, UK, May 1991.

Z was used to specify 37,000 lines out of 268,000 lines of code in the IBM
CICS/ESA 3.1 release. The initial development benefit from using Z was assessed as
being a 9% improvement in the total development cost of the release, based on the
reduction of programmer days fixing problems.

[236] M. Pilling, A. Burns, and K. Raymond. Formal specifications and proofs of
inheritance protocols for real-time scheduling. IEE/BCS Software Engineering

Engineering Institute, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213,
USA, June 1993.

Bowen and Hall [38], pages 230–249.


[240] F. Polack, M. Whiston, and K. Mander. The SAZ project: Integrating SSADM and
Z. In Woodcock and Larsen [316], pages 541–557.

32

Contents: Formal specification in the context of software engineering; An informal introduction to logic and set theory; A first specification; The Z notation: the mathematical language, relations and functions, schemas and specification structure; A first specification revisited; Formal reasoning: From specification to program: data and operation refinement, operation decomposition; From theory to practice.


The 4th VDM-Europe Symposium was held at Noordwijkerhout, The Netherlands, 21–25 October 1991. Papers with relevance to Z include [12, 25, 74, 89, 115, 147, 299, 306, 323]. See also [243].


Papers with relevance to Z include [4, 313]. See also [242].


This article is part of a special issue on Formal aspects of microprocessor design, edited by H. S. M. Zedan. See also [33].


A general article containing information on the formal development of the T800 floating-point unit for the transputer including the use of Z.


A detailed description of a version of Object-Z similar to (but not identical to) that in [97]. The thesis also includes a formalization of temporal logic history invariants and a fully-abstract model of classes in Object-Z.


Published version of 1985 DPhil thesis.


A description of the Z style option ‘zed.sty’ for use with the \LaTeX \textit{document preparation system} [174].


This case study of an embedded real-time kernel shows that mathematical techniques have an important role to play in documenting systems and avoiding design flaws.


The manual describes a Z type-checker and ‘fuzz.sty’ style option for \LaTeX \textit{documents} [174]. The package is compatible with the book, *The Z Notation: A Reference Manual* by the same author [273].


This is a revised edition of the first widely available reference manual on Z originally published in 1989. The book provides a complete and definitive guide to the use of Z in specifying information systems, writing specifications and designing implementations. See also the draft Z standard [52].

Contents: Tutorial introduction; Background; The Z language; The mathematical tool-kit; Sequential systems; Syntax summary; Changes from the first edition; Glossary.


This is a collection of papers describing various OOZ approaches – Hall, ZERO, MooZ, Object-Z, OOZE, Schuman & Pitt, Z++, ZEST and Fresco (an object-oriented VDM method) – in the main written by the methods’ inventors, and all specifying the same two examples. The collection is a revised and expanded version of a ZIP report distributed at the 1991 Z User Meeting at York.


Also available in a longer form as Technical Report UQCS-255, Department of Computer Science, University of Queensland.


Also available in a longer form as Technical Report UQCS-243, Department of Computer Science, University of Queensland.


This paper presents a Z model of the Unix *make* utility.


A case study on using Z for process modelling.


The workshop was held at City University, London, UK, 5–7 January 1994. See [109, 177].


37


The 1st FME Symposium was held at Odense, Denmark, 19–23 April 1993. Z-related papers include [45, 72, 107, 154, 193, 240].


Work on combining Z and CSP.


This book provides a guide to developing software from specification to code, and is based in part on work done at IBM’s UK Laboratory that won the UK Queen’s Award for Technological Achievement in 1992.

Contents: Introduction; A simple Z specification; Sets and predicates; Relations and functions; Schemas and specifications; Data design; Algorithm design; Specification of an oil terminal control system.


ZTC is a type checker for the Z specification language. ZTC accepts two forms of input: LATEX with zed style option and ZSL, an ASCII version of Z. ZTC can also perform translations between the two input forms. This document is intended to serve as both a user’s guide and a reference manual for ZTC.


Also published as [324].