Formal Methods In Software Engineering Examples

Formal Methods and Software EngineeringLarch: Languages and Tools for Formal SpecificationFormal Methods in Systems EngineeringRigorous Software DevelopmentFormal Methods and Software EngineeringFormal Methods and Software EngineeringZFormal Verification of Object-Oriented SoftwareFrom Software Engineering to Formal Methods and Tools, and BackFormal Methods and Software EngineeringFormal Methods in Software EngineeringFormal Engineering for Industrial Software DevelopmentFormal Methods and Software EngineeringFormal Methods and Software EngineeringFormal Methods and Software EngineeringFormal Methods and Software EngineeringConcise Guide to Formal MethodsAutomated Theorem Proving in Software EngineeringFormal Methods and Software EngineeringFormal Methods in Computer ScienceSoftware Engineering and Formal MethodsFormal Methods and Software EngineeringFormal Methods in ManufacturingPractical Formal Software EngineeringPractical TLA+Formal Methods and Software EngineeringFormal Methods for Software EngineeringFormal Methods and Software EngineeringUnderstanding Formal MethodsTeaching and Learning Formal MethodsObject Modeling with the OCLFormal Methods and Software EngineeringFormal Methods and Software EngineeringFormal Methods and Software EngineeringSoftware Engineering and Formal MethodsJava Software Development with Event BA Practical Theory of ProgrammingFoundations of Algebraic Specification and Formal Software DevelopmentFormal Methods and Software EngineeringFormal Methods Fact File

Formal Methods and Software Engineering

This book presents the thoroughly refereed post-conference proceedings of the International Conference on Formal Verification of Object-Oriented Software, FoVeOOS 2011, held in Turin, Italy, in October 2011 – organised by COST Action IC0701. The 10 revised full papers presented together with 5 invited talks were carefully reviewed and selected from 19 submissions. Formal software verification has outgrown the area of academic case studies, and industry is showing serious interest. The logical next goal is the verification of industrial software products. Most programming languages used in industrial practice are object-oriented, e.g. Java, C++, or C#. FoVeOOS 2011 aimed to foster collaboration and interactions among researchers in this area.

Larch: Languages and Tools for Formal Specification

This invaluable textbook/reference provides an easy-to-read guide to the fundamentals of formal methods, highlighting the rich applications of formal methods across a diverse range of areas of computing. Topics and features: introduces the key concepts in software engineering, software reliability and dependability, formal methods, and discrete mathematics; presents a short history of logic, from Aristotle's syllogistic logic and the logic of the Stoics, through Boole's symbolic logic, to Frege's work on predicate logic; covers propositional and predicate logic, as well as more advanced topics such as fuzzy logic, temporal

logic, intuitionistic logic, undefined values, and the applications of logic to AI; examines the Z specification language, the Vienna Development Method (VDM) and Irish School of VDM, and the unified modelling language (UML); discusses Dijkstra's calculus of weakest preconditions, Hoare's axiomatic semantics of programming languages, and the classical approach of Parnas and his tabular expressions; provides coverage of automata theory, probability and statistics, model checking, and the nature of proof and theorem proving; reviews a selection of tools available to support the formal methodist, and considers the transfer of formal methods to industry; includes review questions and highlights key topics in every chapter, and supplies a helpful glossary at the end of the book. This stimulating guide provides a broad and accessible overview of formal methods for students of computer science and mathematics curious as to how formal methods are applied to the field of computing.

Formal Methods in Systems Engineering

This book constitutes the refereed proceedings of the 13th International Conference on Formal Engineering Methods, ICFEM 2011, held in Durham, UK, October 2011. The 40 revised full papers together with 3 invited talks presented were carefully reviewed and selected from 103 submissions. The papers address all current issues in formal methods and their applications in software engineering. They are organized in topical sections on formal models; model checking and probability; specification and development; security; formal verification; cyber physical systems; event-B; verification, analysis and testing; refinement; as well as theorem proving and rewriting.

Rigorous Software Development

This book constitutes the proceedings of the 21st International Conference on Formal Engineering Methods, ICFEM 2019, held in Shenzhen, China, in November 2019. The 28 full and 8 short papers presented in this volume were carefully reviewed and selected from 94 submissions. They deal with the recent progress in the use and development of formal engineering methods for software and system design and record the latest development in formal engineering methods.

Formal Methods and Software Engineering

As computer systems continue to advance, the positions they hold in human society continue to gain power. Computers now control the flight of aircraft, the cooling systems in chemical plants, and feedback loops in nuclear reactors. Because of the vital roles these systems play, there has been growing concern about the reliability and safety of these advanced computers. Formal methods are now widely recognized as the most successful means of assuring the reliability of complex computer systems. Because formal methods are being mandated in more and more international standards, it is critical that engineers, managers, and industrial project leaders are well trained and conversant in the application of these methods. This book covers a broad range of issues relating to the pedagogy of formal methods. The contributors, all acknowledged experts, have based their contributions on extensive experiences teaching and applying formal methods in

both academia and industry.The two editors, both well known in this area, propose various techniques that can help to dismiss myths that formal methods are difficult to use and hard to learn. Teaching and Learning Formal Methods will be an indispensable text for educators in the fields of computer science, mathematics, software engineering, and electronic engineering as well as to management and product leaders concerned with trainingrecent graduates. Offers proven methods for teaching formal methods, even to students who lack a strong background in mathematics Addresses the important role that formal methods play in society and considers their growing future potential Includes contributions from several pioneers in the area Features a foreword written by Edsger W. Dijkstra

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 9th International Conference on Formal Engineering Methods, ICFEM 2007, held in Boca Raton, Florida, USA, November 14-15, 2007. The 19 revised full papers together with two invited talks presented were carefully reviewed and selected from 38 submissions. The papers address all current issues in formal methods and their applications in software engineering. The papers are organized in topical sections.

Z

Formal Verification of Object-Oriented Software

Formal methods have made signi?cant progress in recent years with succe- ful stories from Microsoft (SLAM project), Intel (i7 processor veri?cation) and NICTA/OK-Lab (formal veri?cation of an OS kernel). The main focus of f-malengineeringmethodsliesinhowformalmethodscanbee?ectivelyintegrated intomainstreamsoftwareengineering.Variousadvancedtheories,techniquesand tools havebeen proposed, developed and applied in the speci?cation, design and veri?cation of software or in the construction of software. The challenge now is how to integrate them into engineering development processes to e?ectively deal with large-scale and complex computer systems for their correct and - ?cient construction and maintenance. This requires us to improve the state of the art by researching e?ective approaches and techniques for integration of f- mal methods into industrial engineering practice, including new and emerging practice. Thisseries,InternationalConferencesonFormalEngineeringMethods,brings together those interested in the application of formal engineering methods to computer systems. This volume contains the papers presented at ICFEM 2010, the 12th International Conference on Formal Engineering Methods, held November 17–19, in Shanghai, China, in conjunction with the Third Inter- tional Symposium on Unifying Theories of Programming (UTP 2010). The Program Committee received 114 submissions from 29 countries and - gions. Each paper was reviewed by at least three program committee members.

From Software Engineering to Formal Methods and Tools, and Back

This book constitutes the refereed proceedings of the 4th International Conference on Formal Engineering methods, ICFEM 2002, held in Shanghai, China, in October 2002. The 43 revised full papers and 16 revised short papers presented together with 5 invited contributions were carefully reviewed and selected from a total of 108 submissions. The papers are organized in topical sections on component engineering and software architecture, method integration, specification techniques and languages, tools and environments, refinement, applications, validation and verification, UML, and semantics.

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 9th International Conference on Software Engineering and Formal Methods, SEFM 2011, held in Montevideo, Uruguay, in November 2011. The 22 revised regular papers presented together with 1 short paper, 2 tool papers, and 4 keynote talks were carefully reviewed and selected from 105 initial abstracts and 85 full submissions. Besides the regular session the conference held a special track devoted to "Modeling for Sustainable Development" with 5 accepted papers - selected from 7 submissions - that are also part of this volume. The aim of SEFM is to advance the state of the art in formal methods, to scale up their application in software industry and to encourage their integration with practical engineering methods.

Formal Methods in Software Engineering

Learn how to design complex, correct programs and fix problems before writing a single line of code. This book is a practical, comprehensive resource on TLA+ programming with rich, complex examples. Practical TLA+ shows you how to use TLA+ to specify a complex system and test the design itself for bugs. You'll learn how even a short TLA+ spec can find critical bugs. Start by getting your feet wet with an example of TLA+ used in a bank transfer system, to see how it helps you design, test, and build a better application. Then, get some fundamentals of TLA+ operators, logic, functions, PlusCal, models, and concurrency. Along the way you will discover how to organize your blueprints and how to specify distributed systems and eventual consistency. Finally, you'll put what you learn into practice with some working case study applications, applying TLA+ to a wide variety of practical problems: from algorithm performance and data structures to business code and MapReduce. After reading and using this book, you'll have what you need to get started with TLA+ and how to use it in your mission-critical applications. What You'll Learn Read and write TLA+ specs Check specs for broken invariants, race conditions, and liveness bugs Design concurrency and distributed systems Learn how TLA+ can help you with your day-to-day production work Who This Book Is For Those with programming experience who are new to design and to TLA+. /div

Formal Engineering for Industrial Software Development

This book constitutes the refereed proceedings of the 10th International Conference on Formal Engineering Methods, ICFEM 2008, held in Kitakyushu-City, Japan, October 2008. The 20 revised full papers together with 3 invited talks

presented were carefully reviewed and selected from 62 submissions. The papers address all current issues in formal methods and their applications in software engineering. They are organized in topical sections on specification and verification; testing; verification; model checking and analysis; tools; application of formal methods; semantics.

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 15th International Conference on Formal Engineering Methods, ICFEM 2013, held in Queenstown, New Zealand, in October/November 2013. The 28 revised full papers together with 2 keynote speeches presented were carefully reviewed and selected from 88 submissions. The topics covered are abstraction and refinement, formal specification and modeling, program analysis, software verification, formal methods for software safety, security, reliability and dependability, tool development, integration and experiments involving verified systems, formal methods used in certifying products under international standards, and formal model-based development and code generation.

Formal Methods and Software Engineering

Based around a theme of the construction of a game engine, this textbook is for final year undergraduate and graduate students, emphasising formal methods in writing robust code quickly. This book takes an unusual, engineering-inspired approach to illuminate the creation and verification of large software systems . Where other textbooks discuss business practices through generic project management techniques or detailed rigid logic systems, this book examines the interaction between code in a physical machine and the logic applied in creating the software. These elements create an informal and rigorous study of logic, algebra, and geometry through software. Assuming prior experience with C, C++, or Java programming languages, chapters introduce UML, OCL, and Z from scratch. Extensive worked examples motivate readers to learn the languages through the technical side of software science.

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 11th International Conference on Formal Engineering Methods, ICFEM 2009, held in Rio de Janeiro, Brazil, December 2009. The 36 revised full papers together with two invited talks presented were carefully reviewed and selected from 121 submissions. The papers address all current issues in formal methods and their applications in software engineering. They are organized in topical sections on Testing, Protocols, verification, model checking, object-orientation, event-b, compilation, process algebra, refinement, algebraic specifications and real-time systems.

Formal Methods and Software Engineering

As computer technology is used to control critical systems to an increasing degree, it is vital that the methods for developing and understanding these systems are

substantially improved. The mathematical and scientific foundations currently used are extremely limited which means that their correctness and reliability cannot be ensured to an acceptable level. Systems engineering needs to become a fully fledged scientific discipline and formal methods, which are characterised by their firm mathematical foundations, are playing a vital role in achieving this transition. This volume is based on the proceedings of the Formal Methods Workshop (FM91), held in Drymen, Scotland, 24-27 September 1991. This was the second workshop sponsored by the Canadian and US governments to address the role of formal methods in the development of digital systems. Traditionally, formal methods have evolved in isolation from more conventional approaches, and one of the aims of this workshop was to emphasise the benefits of integrating the two areas. The workshop concentrated on the themes of quality assurance, design methods and mathematical modelling techniques. Particular emphasis was given to safety and security applications. Among the topics covered in this volume are: what is a formal method?; social research on formal methods; current quality assurance methods and formal methods; a pragmatic approach to validation; integrating methods in practice; composition of descriptions; and topics in large program formal development. Formal Methods in Systems Engineering provides an overview of many of the major approaches to formal methods and the benefits which can result from them. It is relevant to academic and industrial researchers, industrial practitioners and government workers with an interest in certification.

Concise Guide to Formal Methods

This is a graduate-level introduction to formal methods. The first part presents two formal languages: logic, in various forms, and Communicating Sequential Process (CSP) as a process algebra. The second part offers specification and testing methods for formal development of software. Building on the foundations from the first part, the reader is allowed to embrace methods for practical applications. The reader will find the examples cutting across chapters valuable for this purpose. The final section takes the reader further into application domains.

Automated Theorem Proving in Software Engineering

Formal Methods Fact File VDM and Z Andrew Harry Formal methods provide a means of specifying computer systems that is unambiguous,concise and well suited to the development of complex software systems for which accuracy and reliability are critical. Heavily mathematical and seemingly difficult to learn, for many they hold little appeal. Andrew Harry speaks as a programmer who has travelled the difficult route to an understanding of formal methods techniques, and knows why it's worth the effort. He explains, in refreshingly simple terms, what formal methods are, why we need them, what should motivate our choice of methods and how to use them effectively. The book presents a novel view of formal methods, spanning the range of specification techniques. An overview of the different styles of formal notation is followed by detailed chapters on the two most popular languages, VDM and Z, consistent with the latest draft standards. There is a readable account of the underlying maths, a short introduction to semantics for proof, and a survey of tools available. Teaching aids include quick reference appendices on the notation and syntax of VDM and Z; exercises (and their solutions); and a useful glossary of terms. A more populist account than most,

this book's "informal" treatment of the subject will appeal to students and industrial programmers who want to know more but find little on the shelves for the novice. Visit our Web page! http://www.wiley.com/compbooks/

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 7th International Conference on Formal Engineering Methods, ICFEM 2005, held in Manchester, UK in November 2005. The 30 revised full papers presented together with 3 invited contributions were carefully reviewed and selected from 74 submissions. The papers address all current issues in formal methods and their applications in software engineering. They are organized in topical sections on specification, modelling, security, communication, development, testing, verification, and tools.

Formal Methods in Computer Science

Building software often seems harder than it ought to be. It takes longer than expected, the software's functionality and performance are not as wonderful as hoped, and the software is not particularly malleable or easy to maintain. It does not have to be that way. This book is about programming, and the role that formal specifications can play in making programming easier and programs better. The intended audience is practicing programmers and students in undergraduate or basic graduate courses in software engineering or formal methods. To make the book accessible to such an audience, we have not presumed that the reader has formal training in mathematics or computer science. We have, however, presumed some programming experience. The roles of fonnal specifications Designing software is largely a matter of combining, inventing, and planning the implementation of abstractions. The goal of design is to describe a set of modules that interact with one another in simple, well defined ways. If this is achieved, people will be able to work independently on different modules, and yet the modules will fit together to accomplish the larger purpose. In addition, during program maintenance it will be possible to modify a module without affecting many others. Abstractions are intangible. But they must somehow be captured and communicated. That is what specifications are for. Specification gives us a way to say what an abstraction is, independent of any of its implementations.

Software Engineering and Formal Methods

This is an excellent introduction to formal methods which will bring anyone who needs to know about this important topic up to speed. It is comprehensive, giving the reader all the information needed to explore the field of formal methods in more detail. It offers: a guide to the mathematics required; comprehensive but easy-to-understand introductions to various methods; a run-down of how formal methods can help to develop high-quality systems that come in on time, within budget, and according to requirements.

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 5th International Conference

on Formal Engineering Methods, ICFEM 2003, held in Singapore in November 2003. The 34 revised full papers presented together with 3 invited contributions were carefully reviewed and selected from 91 submissions. The papers are organized in topical sections on testing and validation, state diagrams, PVS/HOL, refinement, hybrid systems, Z/Object-Z, Petri nets, timed automata, system modelling and checking, and semantics and synthesis.

Formal Methods in Manufacturing

This book constitutes the refereed proceedings of the 16th International Conference on Formal Engineering Methods, ICFEM 2014, held in Luxembourg, Luxembourg, in November 2014. The 28 revised full papers presented were carefully reviewed and selected from 73 submissions. The papers cover a wide range of topics in the area of formal methods and software engineering and are devoted to advancing the state of the art of applying formal methods in practice. They focus in particular on combinations of conceptual and methodological aspects with their formal foundation and tool support.

Practical Formal Software Engineering

This book constitutes the refereed proceedings of the 14th International Conference on Formal Engineering Methods, ICFEM 2012, held in Kyoto, Japan, November 2012. The 31 revised full papers together with 3 invited talks presented were carefully reviewed and selected from 85 submissions. The papers address all current issues in formal methods and their applications in software engineering. They are organized in topical sections on concurrency, applications of formal methods to new areas, quantity and probability, formal verification, modeling and development methodology, temporal logics, abstraction and refinement, tools, as well as testing and runtime verification.

Practical TLA+

In any serious engineering discipline, it would be unthinkable to construct a large system without having a precise notion of what is to be built and without verifying how the system is expected to function. Software engineering is no different in this respect. Formal methods involve the use of mathematical notation and calculus in software development; such methods are difficult to apply to large-scale systems with practical constraints (e.g., limited developer skills, time and budget restrictions, changing requirements). Here Liu claims that formal engineering methods may bridge this gap. He advocates the incorporation of mathematical notation into the software engineering process, thus substantially improving the rigor, comprehensibility and effectiveness of the methods commonly used in industry. This book provides an introduction to the SOFL (Structured Object-Oriented Formal Language) method that was designed and industry-tested by the author. Written in a style suitable for lecture courses or for use by professionals, there are numerous exercises and a significant real-world case study, so the readers are provided with all the knowledge and examples needed to successfully apply the method in their own projects.

Formal Methods and Software Engineering

This volume was published in honor of Stefania Gnesi's 65th birthday. The Festschrift volume contains 32 papers written by close collaborators and friends of Stefania and was presented to her on October 8, 2019 one-day colloquium held in Porto, Portugal, The Festschrift consists of eight sections, seven of which reflect the main research areas to which Stefania has contributed. Following a survey of Stefania's legacy in research and a homage by her thesis supervisor, these seven sections are ordered according to Stefania's life cycle in research, from software engineering to formal methods and tools, and back: Software Engineering; Formal Methods and Tools; Requirements Engineering; Natural Language Processing; Software Product Lines; Formal Verification; and Applications.

Formal Methods for Software Engineering

Formal engineering methods are changing the way that software systems are - veloped.Withlanguageandtoolsupport,theyarebeingusedforautomaticcode generation, and for the automatic abstraction and checking of implementations. In the future, they will be used at every stage of development: requirements, speci?cation, design, implementation, testing, and documentation. The ICFEM series of conferences aims to bring together those interested in the application of formal engineering methods to computer systems. Researchers and practitioners, from industry, academia, and government, are encouraged to attend,andtohelpadvancethestateoftheart.Authorsarestronglyencouraged to make their ideas as accessible as possible, and there is a clear emphasis upon work that promises to bring practical, tangible bene?t: reports of case studies should have a conceptual message, theory papers should have a clear link to application, and papers describing tools should have an account of results. ICFEM 2004 was the sixth conference in the series, and the ?rst to be held in North America. Previous conferences were held in Singapore, China, UK, A- tralia, and Japan. The Programme Committee received 110 papers and selected 30forpresentation.The?nalversionsofthosepapersareincludedhere,together with 2-page abstracts for the 5 accepted tutorials, and shorter abstracts for the 4 invited talks.

Formal Methods and Software Engineering

This textbook gives students a comprehensive introduction to formal methods and their application in software and hardware specification and verification. It has three parts: The first part introduces some fundamentals in formal methods, including set theory, functions, finite state machines, and regular expressions. The second part focuses on logi

Understanding Formal Methods

This book constitutes the refereed proceedings of the 18th International Conference on Formal Engineering Methods, ICFEM 2016, held in Tokyo, Japan, in November 2016. The 27 revised full papers presented together with three invited talks were carefully reviewed and selected from 64 submissions. The conference

focuses in all areas related to formal engineering meth-ods, such as verification and validation, software engineering, formal specification and modeling, software security, and software reliability.

Teaching and Learning Formal Methods

As part of the UML standard OCL has been adopted by both professionals in industry and by academic researchers and is one of the most widely used languages for expressing object-oriented system properties. This book contains key contributions to the development of OCL. Most papers are developments of work reported at different conferences and workshops. This unique compilation addresses many important issues faced by advanced professionals and researchers in object modeling like e.g. real-time constraints, type checking, and constraint modeling.

Object Modeling with the OCL

This book provides foundations for software specification and formal software development from the perspective of work on algebraic specification, concentrating on developing basic concepts and studying their fundamental properties. These foundations are built on a solid mathematical basis, using elements of universal algebra, category theory and logic, and this mathematical toolbox provides a convenient language for precisely formulating the concepts involved in software specification and development. Once formally defined, these notions become subject to mathematical investigation, and this interplay between mathematics and software engineering yields results that are mathematically interesting, conceptually revealing, and practically useful. The theory presented by the authors has its origins in work on algebraic specifications that started in the early 1970s, and their treatment is comprehensive. This book contains five kinds of material: the requisite mathematical foundations; traditional algebraic specifications; elements of the theory of institutions; formal specification and development; and proof methods. While the book is self-contained, mathematical maturity and familiarity with the problems of software engineering is required; and in the examples that directly relate to programming, the authors assume acquaintance with the concepts of functional programming. The book will be of value to researchers and advanced graduate students in the areas of programming and theoretical computer science.

Formal Methods and Software Engineering

The use of mathematical methods in the development of software is essential when reliable systems are sought; in particular they are now strongly recommended by the official norms adopted in the production of critical software. Program Verification is the area of computer science that studies mathematical methods for checking that a program conforms to its specification. This text is a self-contained introduction to program verification using logic-based methods, presented in the broader context of formal methods for software engineering. The idea of specifying the behaviour of individual software components by attaching contracts to them is now a widely followed approach in program development,

which has given rise notably to the development of a number of behavioural interface specification languages and program verification tools. A foundation for the static verification of programs based on contract-annotated routines is laid out in the book. These can be independently verified, which provides a modular approach to the verification of software. The text assumes only basic knowledge of standard mathematical concepts that should be familiar to any computer science student. It includes a self-contained introduction to propositional logic and first-order reasoning with theories, followed by a study of program verification that combines theoretical and practical aspects - from a program logic (a variant of Hoare logic for programs containing user-provided annotations) to the use of a realistic tool for the verification of C programs (annotated using the ACSL specification language), through the generation of verification conditions and the static verification of runtime errors.

Formal Methods and Software Engineering

There are several theories of programming. The first usable theory, often called "Hoare's Logic", is still probably the most widely known. In it, a specification is a pair of predicates: a precondition and postcondition (these and all technical terms will be defined in due course). Another popular and closely related theory by Dijkstra uses the weakest precondition predicate transformer, which is a function from programs and postconditions to preconditions. Iones's Vienna Development Method has been used to advantage in some industries; in it, a specification is a pair of predicates (as in Hoare's Logic), but the second predicate is a relation. Temporal Logic is yet another formalism that introduces some special operators and quantifiers to describe some aspects of computation. The theory in this book is simpler than any of those just mentioned. In it, a specification is just a boolean expression. Refinement is just ordinary implication. This theory is also more general than those just mentioned, applying to both terminating and nonterminating computation, to both sequential and parallel computation, to both stand-alone and interactive computation. And it includes time bounds, both for algorithm classification and for tightly constrained real-time applications.

Formal Methods and Software Engineering

This book constitutes the refereed proceedings of the 17th International Conference on Software Engineering and Formal Methods, SEFM 2019, held in Oslo, Norway, in September 2019. The 27 full papers presented were carefully reviewed and selected from 89 submissions. The papers cover a large variety of topics, including testing, formal verification, program analysis, runtime verification, malware and attack detection,and software development and evolution and address a wide range of systems, such as cyber-physical systems, UAVs, autonomous robots, and feature-oriented and operating systems. They are organized in the following topical sections: cooperative asynchronous systems; cyber-physical systems; feature-oriented and versioned systems; model-based testing; model inference; ontologies and machine learning; operating systems; program analysis; relating models and implementations; runtime verification; security; and verification.

Software Engineering and Formal Methods

This text is about the formal specification language Z suitable for courses on Z and formal methods at first and second year undergraduate level. The book includes a tutorial introduction covering the basic mathematics of Z and provides four specification case studies.

Java Software Development with Event B

Illustrated with real-life manufacturing examples, Formal Methods in Manufacturing provides state-of-the-art solutions to common problems in manufacturing systems. Assuming some knowledge of discrete event systems theory, the book first delivers a detailed introduction to the most important formalisms used for the modeling, analysis, and control of manufacturing systems (including Petri nets, automata, and max-plus algebra), explaining the advantages of each formal method. It then employs the different formalisms to solve specific problems taken from today's industrial world, such as modeling and simulation, supervisory control (including deadlock prevention) in a distributed and/or decentralized environment, performance evaluation (including scheduling and optimization), fault diagnosis and diagnosability analysis, and reconfiguration. Containing chapters written by leading experts in their respective fields, Formal Methods in Manufacturing helps researchers and application engineers handle fundamental principles and deal with typical quality goals in the design and operation of manufacturing systems.

A Practical Theory of Programming

The cost of fixing software design flaws after the completion of a software product is so high that it is vital to come up with ways to detect software design flaws in the early stages of software development, for instance, during the software requirements, the analysis activity, or during software design, before coding starts. It is not uncommon that software requirements are ambiguous or contradict each other. Ambiguity is exacerbated by the fact that software requirements are typically written in a natural language, which is not tied to any formal semantics. A palliative to the ambiguity of software requirements is to restrict their syntax to boilerplates, textual templates with placeholders. However, as informal requirements do not enjoy any particular semantics, no essential properties about them (or about the system they attempt to describe) can be proven easily. Formal methods are an alternative to address this problem. They offer a range of mathematical techniques and mathematical tools to validate software requirements in the early stages of software development. This book is a living proof of the use of formal methods to develop software. The particular formalisms that we use are EVENT B and refinement calculus. In short: (i) software requirements as written as User Stories; (ii) they are ported to formal specifications; (iii) they are refined as desired; (iv) they are implemented in the form of a prototype; and finally (v) they are tested for inconsistencies. If some unit-test fails, then informal as well as formal specifications of the software system are revisited and evolved. This book presents a case study of software development of a chat system with EVENT B and a case study of formal proof of properties of a social network.

Foundations of Algebraic Specification and Formal Software Development

This book constitutes the refereed proceedings of the 8th International Conference on Formal Engineering Methods, ICFEM 2006, held in Macao, China, in November 2006. The 38 revised full papers presented together with three keynote talks were carefully reviewed and selected from 108 submissions. The papers address all current issues in formal methods and their applications in software engineering.

Formal Methods and Software Engineering

Growing demands for the quality, safety, and security of software can only be satisfied by the rigorous application of formal methods during software design. This book methodically investigates the potential of first-order logic automated theorem provers for applications in software engineering. Illustrated by complete case studies on protocol verification, verification of security protocols, and logic-based software reuse, this book provides techniques for assessing the prover's capabilities and for selecting and developing an appropriate interface architecture.

Formal Methods Fact File

Formal engineering methods are changing the way that software systems are - veloped.Withlanguageandtoolsupport,theyarebeingusedforautomaticcode generation, and for the automatic abstraction and checking of implementations. In the future, they will be used at every stage of development: requirements, speci?cation, design, implementation, testing, and documentation. The ICFEM series of conferences aims to bring together those interested in the application of formal engineering methods to computer systems. Researchers and practitioners, from industry, academia, and government, are encouraged to attend,andtohelpadvancethestateoftheart.Authorsarestronglyencouraged to make their ideas as accessible as possible, and there is a clear emphasis upon work that promises to bring practical, tangible bene?t: reports of case studies should have a conceptual message, theory papers should have a clear link to application, and papers describing tools should have an account of results. ICFEM 2004 was the sixth conference in the series, and the ?rst to be held in North America. Previous conferences were held in Singapore, China, UK, A- tralia, and Japan. The Programme Committee received 110 papers and selected 30forpresentation.The?nalversionsofthosepapersareincludedhere,together with 2-page abstracts for the 5 accepted tutorials, and shorter abstracts for the 4 invited talks.

ROMANCE ACTION & ADVENTURE MYSTERY & THRILLER BIOGRAPHIES & HISTORY CHILDREN'S YOUNG ADULT FANTASY HISTORICAL FICTION HORROR LITERARY FICTION NON-FICTION SCIENCE FICTION