Design Methods For Reactive Systems Yourdon Statemate And The Uml

Providing in-depth guidance on how to design and rate emergency pressure relief systems, Guidelines for Pressure Relief and Effluent Handling Systems incorporates the current best designs from the Design Institute for Emergency Relief Systems as well as American Petroleum Institute (API) standards. Presenting a methodology that helps properly size all the components in a pressure relief system, the book includes software with the CCFlow suite of design tools and the new Superchems for DIERS Lite software, making this an essential resource for engineers designing chemical plants, refineries, and similar facilities. Access to Software Access the Guidelines for Pressure Relief and Effluent Handling Software and documents using a web browser at: http://www.aiche.org/ccps/PRTools Each folder will have a readme file and installation instructions for the program. After downloading SuperChems™ for DIERS Lite the purchaser of this book must contact the AIChE Customer Service with the numeric code supplied within the book. The purchaser will then be supplied with a license code to be able to install and run SuperChems™ for DIERS Lite. Only one license per purchaser will be issued.

Design Methods for Reactive Systems describes methods and techniques for the design of software systems-particularly reactive software systems that engage in stimulus-response behavior. Such systems, which include information systems, workflow management systems, systems for e-commerce, production control systems, and embedded software, increasingly embody design aspects previously considered alone-such as complex information processing, non-trivial behavior, and communication between different components-aspects traditionally treated separately by classic software design methodologies. But, as this book illustrates, the software designer is better served by the ability to intelligently pick and choose from among a variety of techniques according to the particular demands and properties of the system under development. Design Methods for Reactive Systems helps the software designer meet today's increasingly complex challenges by bringing together specification techniques and guidelines proven useful in the design of a wide range of software systems, allowing the designer to evaluate and adapt different techniques for different projects. Written in an exceptionally clear and insightful style, Design Methods for Reactive Systems is a book that students, engineers, teachers, and researchers will undoubtedly find of great value. * Shows how the techniques and design approaches of the three most popular design methods can be combined in a flexible, problem-driven manner. * Pedagogical features include summaries, rehearsal questions, exercises, discussion questions, and numerous case studies, with additional examples on the companion Web site.

This book contains papers presented at the 11th Symposium of Computer Aided Process Engineering (ESCAPE-11), held in Kolding, Denmark, from May 27-30, 2001. The objective of ESCAPE-11 is to highlight the use of computers and information technology tools, that is, the traditional CAPE topics as well as the new CAPE topics of current and future interests. The main theme for ESCAPE-11 is process and tools integration with emphasis on hybrid processing, cleaner and efficient technologies (process integration), computer aided systems for modelling, design, synthesis, control (tools integration) and industrial case studies (application of integrated strategies). The papers are arranged in terms of the following themes: computer aided control/operations, computer aided manufacturing, process and tools integration, and new frontiers in CAPE. A total of 188 papers, consisting of 5 keynote and 183 contributed papers are included in this book.

This title serves as an introduction ans reference for the field, with the papers that have shaped the hardware/software co-design since its inception in the early 90s.

This book presents revised tutorial lectures given by invited speakers at the First International Symposium on Formal Methods for Components and Objects, FMCO 2002, held in Leiden, The Netherlands, in November 2002. The 21 revised lectures by leading researchers present a comprehensive account of the potential of formal methods applied to complex software systems such as components and object systems. The book makes a unique contribution to bridging the gap between theory and practice in software engineering.

Learn to apply modeling and parameter estimation tools and strategies to chemicalprocesses using your personal computer This book introduces readers to powerful parameter estimation and computational methods for modeling complex chemical reactions and reaction processes. It presents useful mathematical models, numerical methods for solving them, and statistical methods for testing and discriminating candidate models with experimental data. Topics covered include: Chemical reaction models Chemical reactor models Probability and statistics Bayesian estimation Process modeling with single-response data Process modeling with multi-response data Computer software (Athena Visual Studio) is available via a related Web site http://www.athenavisual.com enabling readers to carry out parameter estimation based on their data and to carry out process modeling using these parameters. As an aid to the reader, an appendix of example problems and solutions is provided. Computer-Aided Modeling of Reactive Systems is an ideal supplemental text for advanced undergraduates and graduate students in chemical engineering courses, while it also serves as a valuable resource for practitioners in industry who want to keep up to date on the most current tools and strategies available.

This book brings together a selection of the best papers from the twenty-first edition of the Forum on specification and Design Languages Conference (FDL), which took place on September 10-12, 2018, in Munich, Germany. FDL is a well-established international forum devoted to dissemination of research results, practical experiences and new ideas in the application of specification, design and verification languages to the design, modeling and verification of integrated circuits, complex hardware/software embedded systems, and mixed-technology systems. Covers Assertion Based Design, Verification & Debug; Includes language-based modeling and design techniques for embedded systems; Covers design, modeling and verification of mixed physical domain and mixed signal systems that include significant analog parts in electrical and non-electrical domains; Includes formal and semi-formal system level design methods for complex embedded systems based on the Unified Modelling Language (UML) and Model Driven Engineering (MDE).

This edited book presents scientific results of the 12th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2013) which was held on June 16-20, 2013 in Toki Messe, Niigata, Japan. The aim of this conference was to bring together scientists, engineers, computer users, and students to share their experiences and exchange new ideas, research results about all aspects (theory, applications and tools) of computer and information science, and to discuss the practical challenges encountered along the way and the solutions adopted to solve them The conference organizers selected the best 20 papers from those papers accepted for presentation at the conference. The papers were chosen based on review scores submitted by members of the program committee, and underwent further rigorous rounds of review.

Formal methods is the term used to describe the specification and verification of software and software systems using mathematical logic. Various methodologies have been developed and incorporated into software tools. An important subclass is distributed systems. There are many books that look at particular methodologies for such systems, e.g. CSP, process algebra. This book offers a more balanced introduction for graduate students that describes the various approaches, their strengths and weaknesses, and when they are best used. Milner's CCS and its operational semantics are introduced, together with notions of behavioural equivalence based on bisimulation techniques and with variants of Hennessy-Milner modal logics. Later in the book, the presented theories are extended to take timing issues into account. The book has arisen from various courses

taught in Iceland and Denmark and is designed to give students a broad introduction to the area, with exercises throughout.
A man may imagine he understands something, but still not understand anything in the way that he ought to. (Paul of Tarsus, 1 Corinthians 8:2) Calling this a ‘practical theory’ may require some explanation. Theory and practice are often thought of as two di?erent worlds, governed bydi?erentideals,principles, andlaws.DavidLorgeParnas, forinstance,who hascontributedmuchtoourtheoreticalunderstandingofsoftwareengineering and also to sound use of theory in the practice of it, likes to point out that ‘theoretically’ is synonymous to ‘not really’. In applied mathematics the goal is to discover useful connections between these two worlds. My thesis is that in software engineering this two-world view is inadequate, and a more intimate interplay is required between theory and practice. That is, both theoretical and practical components should be integrated into a practical theory. It should beclearfrom theabovethattheintended readership of this book is not theoreticians. They would probably have di?culties in appreciating a book on theory where the presentation does not proceed in a logical sequence from basic de?nitions to theorems and mathematical proofs, followed by - plication examples. In fact, all this would not constitute what I understand by a practical theory in this context.
This book provides a framework for software design that shows where the techniques and approaches of design methods for software systems fit in. It discusses three methods in detail and demonstrates how to pick techniques from each of them. It also shows how to follow problem-solving steps that focus on the design problem rather than on the method.
This book constitutes the thoroughly refereed post-conference proceedings of the 8th International Workshop on Model-Based Methodologies for Pervasive and Embedded Software, MOMPES 2012, held in Essen, Germany, in September 2012. The 7 revised full papers presented together with 1 short papers were carefully reviewed and selected from 16 submissions. The papers cover a large spectrum of topics including model-driven engineering, model analysis, runtime verification, modeling of reactive systems, variability modeling, and domain-specific languages.
The IFIP TC-10 Working Conference on Distributed and Parallel Embedded Systems (DIPES 2004) brings together experts from industry and academia to discuss recent developments in this important and growing field in the splendid city of Toulouse, France. The ever decreasing price/performance ratio of microcontrollers makes it economically attractive to replace more and more conventional mechanical or electronic control systems within many products by embedded real-time computer systems. An embedded real-time computer system is always part of a well-specified larger system, which we call an intelligent product. Although most intelligent products start out as stand-alone units, many of them are required to interact with other systems at a later stage. At present, many industries are in the middle of this transition from stand-alone products to networked embedded systems. This transition requires reflection and architecting: The complexity of the evolving distributed artifact can only be controlled, if careful planning and principled design methods replace the - hoc engineering of the first version of many standalone embedded products.
Design Methods for Reactive Systems describes methods and techniques for the design of software systems—particularly reactive software systems that engage in stimulus-response behavior. Such systems, which include information systems, workflow management systems, systems for e-commerce, production control systems, and embedded software, increasingly embody design aspects previously considered alone—such as complex information processing, non-trivial behavior, and communication between different components—aspects traditionally treated separately by classic software design methodologies. But, as this book illustrates, the software designer is better served by the ability to intelligently pick and choose from among a variety of techniques according to the particular demands and properties of the system under development. Design Methods for Reactive Systems helps the software designer meet today's increasingly complex challenges by bringing together specification techniques and guidelines proven useful in the design of a wide range of software systems, allowing the designer to evaluate and adapt different techniques for different projects. Written in an exceptionally clear and insightful style, Design Methods for Reactive Systems is a book that students, engineers, teachers, and researchers will undoubtedly find of great value. Shows how the techniques and design approaches of the three most popular design methods can be combined in a flexible, problem-driven manner. Pedagogical features include summaries, rehearsal questions, exercises, discussion questions, and numerous case studies.
Composed of over 50 papers, "Enterprise Interoperability" ranges from academic research through case studies to industrial and administrative experience of interoperability. The international nature of the authorship continues to broaden. Many of the papers have examples and illustrations calculated to deepen understanding and generate new ideas. This is a concise reference to the state-of-the-art in software interoperability.
Since the late 1980s, the CAiSE conferences have provided a forum for the p- sentation and exchange of research results and practical experiences within the ?eld of Information Systems Engineering. CAiSE 2001 was the 13th conference in this series and was held from 4th to 8th June 2001 in the resort of Int- laken located near the three famous Swiss mountains – the Eiger, M ? onch, and Jungfrau. The ?rst two days consisted of pre-conference workshops and tutorials. The workshop themes included requirements engineering, evaluation of modeling methods, data integration over the Web, agent-oriented information systems, and the design and management of data warehouses. Continuing the tradition of recent CAiSE conferences, there was also a doctoral consortium. The p- conference tutorials were on the themes of e-business models and XML appli-tion development. The main conference program included three invited speakers, two tuto- als, and a panel discussion in addition to presentations of the papers in these proceedings. We also included a special ‘practice and experience’ session to give presentersanopportunitytoreportonanddiscussexperiencesandinvestigations on the use of

methods and technologies in practice. Weextendourthankstothemembersoftheprogramcommitteeandallother referees without whom such conferences would not be possible. The program committee, whose members came from 20 di?erent countries, selected 27 hi- quality research papers and 3 experience reports from a total of 97 submissions. The topics of these papers span the wide-range of topics relevant to information systems engineering – from requirements and design through to implementation and operation of complex and dynamic systems.

This volume contains a selection of papers presented at the third European Computer Aided Systems Theory workshop, EUROCAST '93, held in Spain in February 1993. The workshop emphasizes interdisciplinarity with the specificgoal of creating a synergy between fields such as systems theory, computer science, systems engineering and related areas. The contributions in this volume are strongly related to current problems in CAST research. They emphasize an engineering point of view concerning systems theory. Since the computer is an essential instrument in CAST research, there are close relations to specific topics incomputer science. The papers are grouped into parts on systems theory and systems technology, specific methods, and applications.

Conceptual modeling represents a recent approach to creating knowledge. It has emerged in response to the computer revolution, which started in the middle of the 20th century. Computers, in the meantime, have become a major knowledge media. Conceptual modeling provides an answer to the difficulties experienced throughout the development of computer applications and aims at creating effective, reasonably priced, and sharable knowledge about using computers in business. Moreover, it has become evident that conceptual modeling has the potential to exceed the boundaries of business and computer usage. This state-of-the-art survey originates from the International Seminar on the Evolution of Conceptual Modeling, held in Dagstuhl Castle, Germany, in April 2008. The major objective of this seminar was to look into conceptual modeling from a historical perspective with a view towards the future of conceptual modeling and to achieve a better understanding of conceptual modeling issues in several different domains of discourse, going beyond individual (modeling) projects. The book contains 14 chapters. These were carefully selected during two rounds of reviewing and improvement from 26 presentations at the seminar and are preceded by a detailed preface providing general insights into the field of conceptual modeling that are not necessarily discussed in any of the chapters but nevertheless aid in conceptualizing the inner structure and coherence of the field. The chapters are grouped into the following three thematic sections: the evolution of conceptual modeling techniques; the extension of conceptual modeling to a service-oriented, peer-to-peer, or Web context; and new directions for conceptual modeling.

This book constitutes the refereed proceedings of the 4th International Conference on Service-Oriented Computing, ICSOC 2006, held in Chicago, IL, USA, December 2006. Coverage in this volume includes service mediation, grid services and scheduling, mobile and P2P services, adaptive services, data intensive services, XML processing, service modeling, service assembly, experience with deployed SOA, and early adoption of SOA technology.

This book constitutes the refereed proceedings of the Second Pacific Rim International Workshop on Multi-Agents, PRIMA'99, held in Kyoto, Japan in December 1999. The 17 revised full papers presented were carefully reviewed and selected from a total of 43 submissions. The papers are organized in sections on agent cooperation, agent mobility, learning in multiagent systems, interface agents, and agent system design.

Design Methods for Reactive SystemsYourdan, Statemate, and the UMLMorgan Kaufmann

This book constitutes the refereed proceedings of the 15th International Conference on Advanced Information Systems Engineering, CaiSE 2003, held in Klagenfurt, Austria in June 2003. The 45 revised full papers presented together with 3 invited contributions were carefully reviewed and selected from 219 submissions. The papers are organized in topical sections on XML, methods and models for information systems, UML, Internet business and social modeling, peer-to-peer systems, ontology-based methods, advanced design of information systems, knowledge, knowledge management, Web services, data warehouses, electronic agreements and workflow, requirements engineering, metrics and method engineering, and agent technologies and advanced environments.

An enterprise architecture tries to describe and control an organisation’s structure, processes, applications, systems and techniques in an integrated way. The unambiguous specification and description of components and their relationships in such an architecture requires a coherent architecture modelling language. Lankhorst and his co?authors present such an enterprise modelling language that captures the complexity of architectural domains and their relations and allows the construction of integrated enterprise architecture models. They provide architects with concrete instruments that improve their architectural practice. As this is not enough, they additionally present techniques and heuristics for communicating with all relevant stakeholders about these architectures. Since an architecture model is useful not only for providing insight into the current or future situation but can also be used to evaluate the transition from ‘as?is’ to ‘to?be’, the authors also describe analysis methods for assessing both the qualitative impact of changes to an architecture and the quantitative aspects of architectures, such as performance and cost issues. The modelling language presented has been proven in practice in many real?life case studies and has been adopted by The Open Group as an international standard. So this book is an ideal companion for enterprise IT or business architects in industry as well as for computer or management science students studying the field of enterprise architecture.

This book brings together a selection of the best papers from the twentiethedition of the Forum on specification and Design Languages Conference (FDL), which took place on September 18-20, 2017, in Verona, Italy. FDL is a well-established international forum devoted to dissemination of research results, practical experiences and new ideas in the

application of specification, design and verification languages to the design, modeling and verification of integrated circuits, complex hardware/software embedded systems, and mixed-technology systems. Covers modeling and verification methodologies targeting digital and analog systems; Addresses firmware development and validation; Targets both functional and non-functional properties; Includes descriptions of methods for reliable system design.

This three-volume set of books highlights major advances in the development of concepts and techniques in the area of new technologies and architectures of contemporary information systems. Further, it helps readers solve specific research and analytical problems and glean useful knowledge and business value from the data. Each chapter provides an analysis of a specific technical problem, followed by a numerical analysis, simulation and implementation of the solution to the real-life problem. Managing an organisation, especially in today's rapidly changing circumstances, is a very complex process. Increased competition in the marketplace, especially as a result of the massive and successful entry of foreign businesses into domestic markets, changes in consumer behaviour, and broader access to new technologies and information, calls for organisational restructuring and the introduction and modification of management methods using the latest advances in science. This situation has prompted many decision-making bodies to introduce computer modelling of organisation management systems. The three books present the peer-reviewed proceedings of the 39th International Conference "Information Systems Architecture and Technology" (ISAT), held on September 16–18, 2018 in Nysa, Poland. The conference was organised by the Computer Science and Management Systems Departments, Faculty of Computer Science and Management, Wroclaw University of Technology and Sciences and University of Applied Sciences in Nysa, Poland. The papers have been grouped into three major parts: Part I—discusses topics including but not limited to Artificial Intelligence Methods, Knowledge Discovery and Data Mining, Big Data, Knowledge Based Management, Internet of Things, Cloud Computing and High Performance Computing, Distributed Computer Systems, Content Delivery Networks, and Service Oriented Computing. Part II—addresses topics including but not limited to System Modelling for Control, Recognition and Decision Support, Mathematical Modelling in Computer System Design, Service Oriented Systems and Cloud Computing, and Complex Process Modelling. Part III—focuses on topics including but not limited to Knowledge Based Management, Modelling of Financial and Investment Decisions, Modelling of Managerial Decisions, Production Systems Management and Maintenance, Risk Management, Small Business Management, and Theories and Models of Innovation.

A reactive system is one that is in continual interaction with its environment and executes at a pace determined by that environment. Examples of reactive systems are network protocols, air-traffic control systems, industrial-process control systems etc. Reactive systems are ubiquitous and represent an important class of systems. Due to their complex nature, such systems are extremely difficult to specify and implement. Many reactive systems are employed in highly-critical applications, making it crucial that one considers issues such as reliability and safety while designing such systems. The design of reactive systems is considered to be problematic, and p.oses one of the greatest challenges in the field of system design and development. In this paper, we discuss specification-modeling methodologies for reactive systems. Specification modeling is an important stage in reactive system design where the designer specifies the desired properties of the reactive system in the form of a specification model. This specification model acts as the guidance and source for the implementation. To develop the specification model of complex systems in an organized manner, designers resort to specification modeling methodologies. In the context of reactive systems, we can call such methodologies reactive-system specification modeling methodologies.

This book is a solid foundation of the most important formalisms used for specification and verification of reactive systems. In particular, the text presents all important results on m-calculus, w-automata, and temporal logics, shows the relationships between these formalisms and describes state-of-the-art verification procedures for them. It also discusses advantages and disadvantages of these formalisms, and shows up their strengths and weaknesses. Most results are given with detailed proofs, so that the presentation is almost self-contained. Includes all definitions without relying on other material Proves all theorems in detail Presents detailed algorithms in pseudo-code for verification as well as translations to other formalisms

Reactive systems and event-driven architecture are becoming essential to application design--and companies are taking note. Reactive systems ensure applications are responsive, resilient, and elastic no matter what failures, latency, or other errors may be occurring, while event-driven architecture offers a flexible and composable option for distributed systems. This practical resource helps you bring these approaches together using Quarkus, a Java framework that greatly simplifies the work developers must undertake for cloud deployments. This book covers how Quarkus 2.0 reactive features allow the smooth development of reactive systems. Clement Escoffier and Ken Finnigan from Red Hat show you how to take advantage of event-driven and reactive principles to build more robust distributed systems, reducing latency and increasing throughput, particularly in your microservices and serverless applications. Java developers will also get a foundation in Quarkus, enabling you to create truly Kubernetes-native applications for the cloud. Understand the fundamentals of reactive systems and event-driven architecture Learn how to use Quarkus to build reactive applications Combine Quarkus with Apache Kafka or AMQP to build reactive systems Develop microservices that utilize messages with Quarkus for use in event-driven architectures

This book provides guidelines for practicing design science in the fields of information systems and software engineering research. A design process usually iterates over two activities: first designing an artifact that improves something for stakeholders and subsequently empirically investigating the performance of that artifact in its context. This "validation in context" is a key feature of the book - since an artifact is designed for a context, it should also be validated in this context. The book is divided into five parts. Part I discusses the fundamental nature of design science and its artifacts, as well as related design research questions and goals. Part II deals with the design cycle, i.e. the creation,

design and validation of artifacts based on requirements and stakeholder goals. To elaborate this further, Part III presents the role of conceptual frameworks and theories in design science. Part IV continues with the empirical cycle to investigate artifacts in context, and presents the different elements of research problem analysis, research setup and data analysis. Finally, Part V deals with the practical application of the empirical cycle by presenting in detail various research methods, including observational case studies, case-based and sample-based experiments and technical action research. These main sections are complemented by two generic checklists, one for the design cycle and one for the empirical cycle. The book is written for students as well as academic and industrial researchers in software engineering or information systems. It provides guidelines on how to effectively structure research goals, how to analyze research problems concerning design goals and knowledge questions, how to validate artifact designs and how to empirically investigate artifacts in context – and finally how to present the results of the design cycle as a whole.

AMAST’s goal is to advance awareness of algebraic and logical methodology as part of the fundamental basis of software technology. Ten years and seven conferences after the start of the AMAST movement, I believe we are attaining this. The movement has propagated throughout the world, assembling many enthusiastic specialists who have participated not only in the conferences, which are now annual, but also in the innumerable other activities that AMAST promotes and supports. We are now facing the Seventh International Conference on Algebraic Methodology and Software Technology (AMAST’98). The previous meetings were held in Iowa City, USA (1989 and 1991), in Enschede, The Netherlands (1993), in Montreal, Canada (1995), in Munich, Germany (1996), and in Sydney, Australia (1997). This time it is Brazil’s turn, in a very special part of this colorful country – Amazonia. Thus, “if we have done more it is by standing on the shoulders of giants.” The effort started by Teodor Rus, Arthur Fleck, and William A. Kirk at AMAST’89 was consolidated in AMAST'91 by Teodor Rus, Maurice Nivat, Charles Rattray, and Giuseppe Scollo. Then came modular construction of the building, wonderfully carried out by Giuseppe Scollo, Vangalur Alagar, Martin Wirsing, and Michael Johnson, as Program Chairs of the AMAST conferences held between 1993 and 1997.

Covers central topics in information systems modeling and architectures. Includes the latest developments in information systems modeling, methods, and best practices.

This book brings together a selection of the best papers from the seventeenth edition of the Forum on specification and Design Languages Conference (FDL), which took place on October 14-16, 2014, in Munich, Germany. FDL is a well-established international forum devoted to dissemination of research results, practical experiences and new ideas in the application of specification, design and verification languages to the design, modeling and verification of integrated circuits, complex hardware/software embedded systems, and mixed-technology systems.

This book will attempt to give a first synthesis of recent works con cerning reactive system design. The term "reactive system" has been introduced in order to at'oid the ambiguities often associated with by the term "real-time system," which, although best known and more sugges tive, has been given so many different meanings that it is almost in evitably misunderstood. Industrial process control systems, transporta tion control and supervision systems, signal-processing systems, are ex amples of the systems we have in mind. Although these systems are more and more computerized, it is sur prising to notice that the problem of time in computer science has been studied only recently by "pure" computer scientists. Until the early 1980s, time problems were regarded as the concern of performance evalu ation, or of some (unjustly scorned) "industrial computer engineering," or, at best, of operating systems. A second surprising fact, in contrast, is the growth of research con cerning timed systems during the last decade. The handling of time has suddenly become a fundamental goal for most models of concurrency. In particular, Robin Alilner 's pioneering works about synchronous process algebras gave rise to a school of thought adopting the following abstract point of view: As soon as one admits that a system can instantaneously react to events, i. e.

Enterprise modeling (EM) has gained substantial popularity both in the academic community and among practitioners. A variety of EM methods, approaches, and tools are developed and offered on the market. In practice they are used for various p- poses such as business strategy development, process restructuring, as well as business and IT architecture alignment and governance. PoEM 2008, the First IFIP WG 8. 1 Working Conference on The Practice of Ent- prise Modeling, took place in Stockholm, Sweden. It is the first conference aiming to establish a dedicated forum where the use of EM in practice is addressed by bringing together researchers, users, and practitioners. The goals of PoEM 2008 were to - velop a better understanding of the practice of EM, to contribute to improved EM practice, as well as to share knowledge and experiences. The theme of PoEM 2008 was EM in different application contexts, e. g. , software development, including agile development, as well as business development, gove- ance, and change.

The Sixth Refinement Workshop took place at City University in London from 5th to 7th January 1994. The present volume includes all of the papers which were submitted and accepted for presentation, together with two papers by invited speakers. The workshops in the series have generally occurred at one year intervals but in this last case a two year period had elapsed. These workshops have established themselves as an important event in the calendar for all those who are interested in progress in the underlying theory of refinement and in the take-up by industry of the methods supported by that theory. One of the proposed themes of the sixth workshop was the reporting of successful adoption in industry of rigorous software development methods. The programme committee was perhaps slightly disappointed by the response from industry to the call in this respect. However, the recent period could be characterised as one of consolidation, when those companies which have made the decision that formal development methods are important to their business have been adopting them where appropriate and finding them to be worthwhile. On the other hand,. the difficult economic climate which exists in most parts of the developed world is perhaps not the context within which companies still dubious about the benefits are goil'\g to opt for making major changes in their working practices.

This book is based upon work done under the project "Correct Software through Formal Methods" supported by the German Ministry of Research and Technology. As a case-study report on the practice of formal software development, this book systematically presents and compares 18 different approaches to the control of a real-world production cell. Mathematically precise, formal methods play an increasingly important role in software development, particularly in areas where failure of software would result in injury to people or, at best, significant loss of money.

By analyzing the benefits and explaining the use and limitations of formal methods on a sample basis, this book provides a roadmap for the selection and application of appropriate approaches and thus helps in putting formal methods into industrial use.

Daily life relies more and more on safety critical systems, e.g. in areas such as power plant control, traffic management, flight control, and many more. MOVEP is a school devoted to the broad subject of modeling and verifying software and hardware systems. This volume contains tutorials and annotated bibliographies covering the main subjects addressed at MOVEP 2000. The four tutorials deal with Model Checking, Theorem Proving, Composition and Abstraction Techniques, and Timed Systems. Three research papers give detailed views of High-Level Message Sequence Charts, Industrial Applications of Model Checking, and the use of Formal Methods in Security. Finally, four annotated bibliographies give an overview of Infinite State Space Systems, Testing Transition Systems, Fault-Model-Driven Test Derivation, and Mobile Processes.

This book brings together a selection of the best papers from the nineteenth edition of the Forum on specification and Design Languages Conference (FDL), which took place on September 14-16, 2016, in Bremen, Germany. FDL is a well-established international forum devoted to dissemination of research results, practical experiences and new ideas in the application of specification, design and verification languages to the design, modeling and verification of integrated circuits, complex hardware/software embedded systems, and mixed-technology systems.

Summary Reactive Design Patterns is a clearly written guide for building message-driven distributed systems that are resilient, responsive, and elastic. In this book you'll find patterns for messaging, flow control, resource management, and concurrency, along with practical issues like test-friendly designs. All patterns include concrete examples using Scala and Akka. Foreword by Jonas Bonér. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Modern web applications serve potentially vast numbers of users - and they need to keep working as servers fail and new ones come online, users overwhelm limited resources, and information is distributed globally. A Reactive application adjusts to partial failures and varying loads, remaining responsive in an ever-changing distributed environment. The secret is message-driven architecture - and design patterns to organize it. About the Book Reactive Design Patterns presents the principles, patterns, and best practices of Reactive application design. You'll learn how to keep one slow component from bogging down others with the Circuit Breaker pattern, how to shepherd a many-staged transaction to completion with the Saga pattern, how to divide datasets by Sharding, and more. You'll even see how to keep your source code readable and the system testable despite many potential interactions and points of failure. What's Inside The definitive guide to the Reactive Manifesto Patterns for flow control, delimited consistency, fault tolerance, and much more Hard-won lessons about what doesn't work Architectures that scale under tremendous load About the Reader Most examples use Scala, Java, and Akka. Readers should be familiar with distributed systems. About the Author Dr. Roland Kuhn led the Akka team at Lightbend and coauthored the Reactive Manifesto. Brian Hanafee and Jamie Allen are experienced distributed systems architects. Table of Contents PART 1 - INTRODUCTION Why Reactive? A walk-through of the Reactive Manifesto Tools of the trade PART 2 - THE PHILOSOPHY IN A NUTSHELL Message passing Location transparency Divide and conquer Principled failure handling Delimited consistency Nondeterminism by need Message flow PART 3 - PATTERNS Testing reactive applications Fault tolerance and recovery patterns Replication patterns Resource-management patterns Message flow patterns Flow control patterns State management and persistence patterns