Preface

The study of real-time systems has been recognized over the past 30 years as a discipline of its own whose research community is firmly established in academia as well as in industry. This book aims at presenting some fundamental problems, methods, and techniques of this domain, as well as questions open for research.

The field is mainly concerned with the control and analysis of dynamically evolving systems for which requirements of timeliness are paramount. Typical examples include systems for the production or transport of goods, materials, energy or information. Frequently, controllers for these systems are "embedded" in the sense that they are physically implemented within the environment with which they interact, such as a computerized controller in a plane or a car. This characteristic imposes strong constraints on space, cost, and energy consumption, which limits the computational power and the available memory for these devices, in contrast with traditional applications of computer science where resources usually grow exponentially according to Moore's law. The design of real-time systems relies on techniques that originate in several disciplines, including control theory, operations research, software engineering, stochastic process analysis, and others.

Software supporting real-time systems needs not only compute the correct value of a given function, but it must also deliver these values at the right moment in order to ensure the safety and the required performance level of the overall system. Usually, this is implemented by imposing constraints (or *deadlines*) on the termination of certain activities. The verification techniques presented in this volume can help to ensure that deadlines are respected.

Chapter written by Stephan MERZ and Nicolas NAVET.

The chapters of this book present basic concepts and established techniques for modeling real-time systems and for verifying their properties. They concentrate on functional and timing requirements; the analysis of non-functional properties such as schedulability and Quality of Service guarantees would be a useful complement, but would require a separate volume. Formal methods of system design are based on mathematical principles and abstractions; they are a cornerstone for a "zero-default" discipline. However, their use for the development of real-world systems requires the use of efficient support tools. The chapters therefore emphasize the presentation of verification techniques and tools associated with the different specification methods, as well as the presentation of case studies that illustrate the application of the formalisms and the tools. The focus lies on model checking approaches, which attempt to provide a "push-button" approach to verification and integrate well into standard development processes. The main obstacle for the use of model checking in industrial-sized developments is the state-explosion problem, and several chapters describe techniques based on abstraction, reduction or compression that stretch the limits of the size of systems that can be handled.

Before verification can be applied, the system must be modeled in a formal description language such as (timed) Petri nets, timed automata or process algebra. The properties expected of a system are typically expressed in temporal logic or using automata as observers. Two main classes of properties are *safety* properties that, intuitively, express that nothing bad ever happens, and *liveness* properties that assert that something good eventually happens. The third step is the application of the verification algorithm itself to decide whether the properties hold over the model of the system or not; in the latter case, model checking generates a counter-example exhibiting a run of the system that violates the property.

Beyond *verification*, which compares two formal objects, the model should also be *validated* to ensure that it faithfully represents the system under development. One approach to validation is to decide healthiness properties of the model (for example, ensure that each component action can occur in a system run), and model checking is again useful here. In general, it is helpful to narrow the gap between the system description and its formal model, for example by writing a model in a high-level executable language or in a notation familiar to designers such as UML. The chapters of this book, written by researchers active in the fields, present different possible approaches to the problems of modeling, verification and validation, as well as open research questions.

Chapter 1, written by Bernard Berthomieu, Florent Peres and François Vernadat, explains the analysis of real-time systems based on timed Petri nets. It illustrates the high expressiveness of that formalism and the different, complementary verification techniques that are implemented in the Tina tool.

In Chapter 2, Camille Constant, Thiery Jéron, Hervé Marchand and Vlad Rusu describe an approach that combines verification and conformance testing (on the actual implementation platform) of input/output symbolic transition systems. Disciplined approaches to testing are indeed a very valuable complement to formal verification for ensuring the correctness of an implementation. This is true in particular when the complexity of the models makes exhaustive verification impossible.

Chapters 3 and 4 are devoted to the presentation of model checking techniques. Starting with the canonical example of a lift controller, Stephan Merz presents the basic concepts and techniques of model checking for discrete state transition systems: temporal logics, principles of model checking algorithms and their complexity, and strategies for mastering the state explosion problem. Patricia Bouyer and François Laroussinie focus on model checking for timed automata, the main semantic formalism for modeling real-time systems. They describe the formalism itself, timed modal and temporal logics, as well as some extensions and subclasses of timed automata. Finally, algorithms and data structures for the representation and verification of timed automata are introduced, and the modeling and verification environment Uppaal is described in some detail.

Using a model of an industrial drilling station as a running example, Radu Mateescu presents in Chapter 5 the functionalities of the CADP toolbox for modeling and verification. CADP is designed to model arbitrary asynchronous systems whose components run in parallel and communicate by message passing. The toolbox accepts models written in different formalisms, including networks of communicating automata or higher-level models written in Lotos. It implements a set of model transformations, simulation and verification algorithms, and offers the possibility to generate conformance tests for the implementation.

Chapter 6, written by Pascal Raymond, is devoted to the verification of programs written in the synchronous language Lustre with the help of the model checker Lesar. Synchronous languages enjoy ever more success for the development of reactive systems, of which real-time systems are a particular instance. Based on mathematical models of concurrency and time, synchronous languages provide a high-level abstraction for the programmer and are well-suited to formal verification.

In Chapter 7, Paul Caspi, Grégoire Hamon and Marc Pouzet go on to describe the language Lucid Synchrone that extends Lustre with constructs borrowed from functional languages, further augmenting expressiveness. The authors start by asking why synchronous languages are relevant for the design of critical systems. They give an account of the development of the Lucid language, and present in detail its primitives and the underlying theoretical concepts, illustrating them by several examples.

One of the most exciting developments over the past 15 years has been the emergence of techniques for the verification of probabilistic systems, intimately coupled with work on stochastic processes carried out in the realms of performance evaluation. Probabilistic models are very useful because they add quantitative information above the non-deterministic representation of the behavior of system components and the environment. They can also be used to determine system parameters such as queue sizes, as a function of the desired failure guarantees. Marta Kwiatkowska, Gethin Norman, David Parker and Jeremy Sproston lay the bases in Chapter 8 by defining probabilistic timed automata and extending the model checking algorithms for ordinary timed automata to handle probabilistic models. The case study of the IEEE FireWire Root Contention Protocol illustrates the application of these techniques. In Chapter 9, Serge Haddad and Patrice Moreaux give an overview of verification techniques for probabilistic systems: discrete and continuous time Markov chains, stochastic Petri nets, Markov decision processes and associated temporal logics. They also cover some of the main tools used in this domain, including GreatSPN, ETMCC, and Prism.

Chapter 10, written by Marius Bozga, Susanne Graf, Laurent Mounier, and Iulian Ober, presents the IF toolbox, a tool environment for modeling and verifying real-time systems centered around a common internal description language based on communicating timed automata. User-level specifications written in languages such as SDL or UML are translated into this internal representation and can be subject to analysis using algorithms of static analysis, reduction and model checking. An extended case study from the aerospace domain, based on joint work with EADS concludes the chapter.

Chapter 11, written by Anne-Marie Déplanche and Sébastien Faucou, is dedicated to the architecture description language AADL, originally designed and standardized for the avionic and aerospace domains, but which is an excellent candidate for arbitrary real-time systems. Architectural descriptions can serve as a reference for all actors involved in system design; they contain the information needed for simulation, formal verification, and testing. The authors examine the specific requirements for describing real-time systems and then present the AADL and its support tools. Their use is illustrated with the help of a case study of a closed-loop control system.

We would like to express our gratitude to all of the authors for the time and energy they have devoted to presenting their topic. We are also grateful to ISTE Ltd. for having accepted to publish this volume and for their assistance during the editorial phase.

We hope that you, the readers of this volume, will find it an interesting source of inspiration for your own research or applications, and that it will serve as a reliable, complete and well-documented source of information on real-time systems.

Stephan MERZ and Nicolas NAVET
INRIA Nancy Grand Est and LORIA
Nancy, France