Preface

Time has become a part of computing, where it has claimed its own models. This book summarizes several decades of research on developing, analyzing, and applying time models to computing and, more generally, engineering.

Every researcher contributing to such a burgeoning field is forced to consider several available approaches, to understand what they have in common and how they differ. This continued effort prompts a systematic work of comparison, classification, and assessment of the diverse models and methods. This book picks up this call and tries to put it into practice.

Our survey paper with the same title[1] is a precursor work: while paper and book share the same fundamental themes, this book is aimed at a wider public than computer science researchers – including engineers, students, and practitioners – and is more ambitious in scope and number of topics and examples.

Scope and Topics

Models including a notion of time are ubiquitous in the natural sciences and engineering, and they have also received the attention of philosophers, linguists, and other scholars. It is in computing, however, that the abstractions provided by the traditional models of time in the physical sciences can be inadequate. To bridge this hiatus between abstractions, computer science research has been quite prolific in spawning novel models of time. This book is a thorough presentation of the state of the art in this area, including some historical perspective and not omitting the occasional controversial aspects.

This book is *not* meant to be an encyclopedic catalog of models of time, nor to be a reference handbook that collects independent chapters sharing a common theme.

[1] Furia, C.A., Mandrioli, D., Morzenti, A., Rossi, M.: Modeling time in computing: a taxonomy and a comparative survey. ACM Comput. Surv. **42**(2), 1–59 (2010).

Rather, it is an organized presentation of the issues related with timing modeling and analysis that recur in computer science and other branches of engineering and science, and of the most significant available solutions. The connections among the themes of different chapters and the usage of a taxonomy of fundamental issues help readers develop a solid understanding of the problems and their solutions, which is applicable outside the necessarily limited array of topics covered in the book, in their concrete research and practice. For each chapter, we selected examples and approaches that cover some fundamental aspects of time modeling, and presented them in a way that abstracts away the inessential low-level details and focuses on the novel conceptual contributions and their relations to other approaches. Each chapter is supplemented by a comprehensive collection of bibliographic remarks, where readers interested in knowing all the details about a specific approach, or looking for variants of a fundamental idea, can find up-to-date references to the scientific literature.

Audience

The book addresses an ambitiously broad audience. Graduate students and researchers in computer science – or with a computer science background – are the primary intended readership. Researchers and practitioners in other scientific and engineering disciplines interested in time modeling with a computational flavor will also find the book interesting and approachable: the comparative and conceptual approach of the book makes it a valuable introduction for non-experts, and a useful preparatory reading to more specialized technical references.

A large number of engineering disciplines target software-intensive systems whose correctness crucially depends on the timing behavior, for example, embedded control systems, complex manufacturing systems, avionics systems, railway signaling systems, and so on. The concepts, models, techniques, and methods developed by computer scientists to describe timing behavior have increasingly been applied beyond their original application domains to model and analyze such software-intensive systems. This requires a mutual knowledge exchange between traditionally distinct engineering areas: domain experts are usually unfamiliar with the concepts and notations specialized by computer science to model time, whereas computer scientists may be unprepared to apply their methods to heterogeneous systems operating in unfamiliar domains. This book may contribute to forming a shared vocabulary: the presentation uses a rigorous, yet not overly technical, style, which is approachable by readers with heterogeneous specialized backgrounds.

Addressing a wide public has two opposite risks: repeating material already known and making the presentation not self-contained. We structured the book to minimize these risks, by providing a flexible reading approach as we explain next.

Book Structure

The book has two parts and a total of 12 chapters. Chapters 1–3, as well as the concluding Chap. 12, belong to the frame; Part I includes Chaps. 4–6, Part II the remaining Chaps. 7–11.

Chapter 1 introduces the book, its contents, topics, structure, and goals. Chapter 2 introduces the notions of formalism and model in general terms, and some of their fundamental classification criteria; while doing so, it presents the fundamentals of propositional and predicate logic.

Chapter 3 is a cornerstone of the whole book: it introduces some essential issues that arise when modeling time across every type of system. The presentation of the numerous formalisms in the rest of the book constantly refers to these "dimensions" to put the different models on a common ground.

Part I is a concise summary of the models of time that are traditional in engineering and the natural sciences, including fundamental computer science. Chapter 4 targets dynamical systems and control theory; Chap. 5 takes the point of view of hardware design, and Chap. 6 of software algorithmic and complexity analysis. Part I is meant to provide heterogeneous readers with a homogeneous background; hence the reading will focus on the topics that are less familiar. For example, readers with expertise in control, electrical, or mechanical engineering will probably skim through Chap. 4 and spend more time on Chaps. 5 and 6 to acquaint themselves with some basic computer science models, in preparation for Part II. Conversely, computer scientists may find the content of Chap. 4 especially useful, whereas they will probably only use Chaps. 5 and 6 as references or reminders.

Part II covers advanced and specialized formalisms developed for dealing with specific issues of time modeling in heterogeneous software-intensive systems. Chapter 7 presents formalisms that all share finite state machines as common "ancestors"; Chap. 8 discusses Petri nets in many variants; Chap. 9 targets notations based on mathematical logic, such as temporal logic; Chap. 10 is about process algebras – widely used to model concurrency, but less prominent in timing analysis; Chap. 11 presents some "dual-language approaches" that combine two notations with different characteristics to model and verify complex systems (model checking frameworks are the most popular applications of the dual-language approach).

Chapter 12 concludes the book with summarizing remarks and hints towards future developments and open challenges.

Prerequisites, Dependencies, and Teaching

The book assumes a basic knowledge of the standard topics of engineering and science undergraduate curricula: calculus, probability theory, algorithms, and programming.

A more advanced knowledge of computer science topics, in particular automata, formal languages, and mathematical logic, is not required but will make the book easier to read and understand in full. Readers without a computer science background should, in any case, read Chaps. 2, 5, and 6 with attention and make sure they understand the basic notions presented there. Mathematical logic is a particularly critical topic, in that Chap. 2 cannot replace a detailed introduction to such a technical topic; consequently, readers totally unfamiliar with propositional and predicate logic may have to skip the more advanced topics covered in Chaps. 9 and 11.

Chapter 4 is in contrast required reading for readers with little scientific knowledge outside "traditional" computer science. More generally, many of the examples in the book refer to systems from various engineering domains; hence it is important to develop an interdisciplinary view of problems and solutions as the reading progresses.

This book can be used as a textbook in multiple ways. The typical usage could be with a one-semester graduate course, where instructors can focus on the topics and models from Part II that are closest to the course topics and goals, according to the students' background. In particular, every chapter of Part II starts by presenting formalisms in their basic form (e.g., transition systems in Chap. 7, and basic Petri nets in Chap. 8) and then continues with richer extensions (e.g., Statecharts in Chap. 7, and timed Petri nets in Chap. 8). The teaching may stop at the basic levels, or tackle advanced extensions only of certain formalisms. An aspect that is mostly independent of the others is modeling with probabilities; readers can safely skip these parts if uninterested or when lacking a background in probability theory.

Exercises

Exercises are an important device for taking full advantage of the book's contents. While they are optional in a first reading, some of them become necessary for assimilating every technical concept and for understanding the most subtle aspects of the presentation. This does not necessarily mean writing out every solution in full detail: checking that the requests are clear and making the first steps toward a solution may be enough to take advantage of most exercises.

The exercises appear close to the topics they target. They have varying difficulty and scope; those requiring a stronger technical background are marked with a "♦"; their headings may specifically indicate which notions are assumed (e.g., the theory of computation, or probability theory).

A few exercises, marked with a "♣", have a distinctively conceptual flavor, in that their main purpose is to be thought provoking and they may admit different legitimate solutions, possibly informal. This type of exercise mostly occurs in Chap. 3, given its foundational nature.

Acknowledgments

The people who supported, directly or indirectly, the development of the book are too many to acknowledge; hence we start by apologizing for the inevitable omissions.

We thank the colleagues of many years who worked with us in our research on time modeling and contributed to broadening and shaping our views. This includes people from academia – the "historical fellows" Carlo Ghezzi and Pierluigi San Pietro, as well as the other major coworkers Connie Heitmeyer, Dick Kemmerer, Alberto Coen-Porisini, Miguel Felder, Sandro Morasca, Mauro Pezzè, Angelo Gargantini, Matteo Pradella, Paul Kolano, Paola Spoletini, Marcello Bersani, and Luca Ferrucci – and also practitioners from industry, in particular Piergiorgio Mirandola and the whole ENEL-CRA group, which provided technical cooperation besides financial support. We also acknowledge the fruitful collaboration with Selenia-Alenia, Elsag, Agusta-Westland, and Parvis.

The presentation has benefitted from the suggestions made indirectly by Bertrand Meyer while working with the first author on different topics; the first author also thanks Scott West for the English tips he gave – knowingly and unknowingly – and Sebastian Nanz for suggesting bibliographic references for Chap. 10 and for putting up with lengthy and frequent phone meetings with the other authors.

Finally, we are very grateful to the editor Grzegorz Rozenberg, to the anonymous reviewer, and to the Springer staff for their support and careful reading of various drafts of the book. Uncaught errors and deficiencies are, of course, our responsibility.

Zürich, Switzerland
Milan, Italy

Carlo A. Furia
Dino Mandrioli
Angelo Morzenti
Matteo Rossi

http://www.springer.com/978-3-642-32331-7

Modeling Time in Computing
Furia, C.A.; Mandrioli, D.; Morzenti, A.; Rossi, M.
2012, XVI, 424 p., Hardcover
ISBN: 978-3-642-32331-7