Compared with other engineering disciplines, software engineering is still a relatively young discipline. In particular, its formal foundation is still under development, and the importance of theoretical results is often underestimated due to prejudices that cast doubt on their general practicability.

For this reason, Kurki-Suonio’s book aims at developing a practical theory for the development of distributed reactive systems. In more detail, descriptions and methods are considered for property-preserving incremental design steps of operational action-oriented models. Instead of a precise formal foundation, the author emphasizes the practical usefulness of the presented approach. Consequently, he does not spend much effort on formal proofs or theorems. Instead, many convincing examples to demonstrate the application of the discussed concepts are given.

The book is divided into 11 chapters, which are grouped into five parts. Chapter 1, the first part, discusses the role and need of theories (formal methods) in engineering in general and introduces some basic definitions. Part 2 establishes the formal foundation used in the approach: in three chapters, it introduces a low-level action language with a fair execution scheme as well as a temporal logic of actions (TLA) as a foundation of the action language together with some deduction rules to verify properties of the considered models.

Part 3, the core of the book, aims at building a practical theory on top of the fundamentals given in the second part. First, chapter 5 shows how basic language facilities such as types, finite-state structures, objects and classes, relations between objects as well as multi-object actions can be implemented on top of the primitive execution model introduced in part 2. The next chapter is devoted to design methods: in particular, superposition and composition of incrementally refined, layered specifications are considered. It is crucial that the incremental refinement process maintains certain properties of the system such as safety properties. Chapter 7 introduces object-oriented concepts on top of the so-far developed language such as classes and objects as well as means for aggregation and inheritance. In particular, object-oriented concepts are compared with layered specifications where inheritance is viewed to establish an implication relation between different classes. Finally, chapter 8 focuses on components and interfaces and their corresponding refinement.

Part 4 integrates distributed (chapter 9) and real-time systems (chapter 10) in the developed approach. Chapter 9 considers how action models can be implemented with distributed systems and develops relevant execution policies. Chapter 10 aims at modelling real-time systems by simply adding a notion of time to the action language and even incorporates hybrid systems by handling continuous state functions. It is not difficult to see how real time can be added to the action language as a simple variable. However, many problems that have to be solved for the design of real-time systems cannot be addressed in this approach since the provided models do not consider the hardware architecture.

The approach presented in the book is the result of many years of research. The design method developed by the author has been successfully used for the construction of many distributed parallel systems. A temporal logic of actions (TLA) is used as a foundation in that the discussed concepts are defined in terms of TLA. The relationship to other action-oriented languages, such as Chandy and Misra’s Unity, is however closer.

Potential audience of this book is definitely found among software engineers who want to put their software development on a more formal basis without getting lost in difficult formalisms. For this audience, the book provides a valuable method for the incremental construction of distributed software together with property-preserving refinement techniques that go beyond object-orientation. The book is also useful for researchers working in the mentioned area. However, researchers interested in formal verification may miss a more formal treatment. Moreover, problems occurring with real-time systems are only briefly sketched in the book.

KLAUS SCHNEIDER
University of Kaiserslautern Germany
Email: Klaus.Schneider@informatik.uni-kl.de