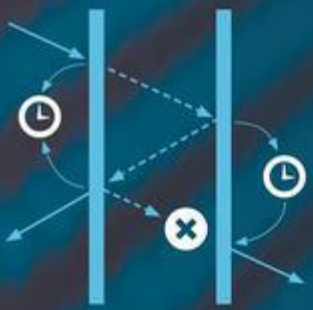

Hartmut König

Protocol Engineering

Springer

Protocol Engineering

Hartmut König

Protocol Engineering

Springer

Hartmut König
Department of Computer Science
Brandenburg University of Technology Cottbus
Cottbus
Germany

Revised English Translation from the German edition:
Protocol Engineering. Prinzip, Beschreibung und Entwicklung von Kommunikationsprotokollen
by Hartmut König

ACM Codes: C.2, B.4

ISBN 978-3-642-29144-9 ISBN 978-3-642-29145-6 (eBook)
DOI 10.1007/978-3-642-29145-6
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012950010

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To Charlotte, Christine, and Sven

Preface

Motivation for the book

Communication protocols – for short protocols – form the basis for the operation of computer networks and telecommunication systems. They are behavior conventions which describe how communication systems interact with each other in computer networks. Protocols define the temporal order of the interactions and the formats of the data units exchanged. Communication protocols comprise a wide range of different functions and mechanisms, such as the sending and receiving of data units, their coding/decoding, error control mechanisms, timer control, flow control, and many others. Protocols essentially determine the efficiency and reliability of computer networks. The processes in protocols, however, may be very complex and sophisticated. Concurrent processes and the nondeterministic appearance of events increase the complexity of protocol behaviors. The diversity of the involved mechanisms is often in conflict with the hoped for efficiency.

Communication systems use defined protocol hierarchies which are based on fixed architectural principles like in the Internet architecture. Protocols provide a specific functionality which is offered in the form of a service to other protocols or to an application. Different principles have been applied for the design of protocol hierarchies. Closed or proprietary architectures are dedicated to the requirements of a certain application or to the products of a company. Open architectures, in contrast, provide unified interaction principles which allow one to set up heterogeneous networks.

Communication protocols can be implemented in either hardware or software. Implementations in software prevail, especially in higher layers. The implementation of protocols is closely connected to the target execution environment, in particular to the given operating system. The manner in which a protocol is implemented influences its efficiency just as strongly as its design.

The experience of almost error-free use of services in the Internet hides the efforts needed for the development of communication protocols. Before protocols can be installed in a network they have to be designed, described, verified, adapted, implemented, and tested. The development of protocols – from design to installation – is a complex, tedious, and error-prone process which is only in part automated nowadays. Protocol development raises similar questions and problems as software engineering. Many of the typical phases and features of software development are also contained in the protocol development process. However, the distributed character of communication protocols raises a number of additional issues which go beyond traditional software development. For that reason, **Protocol**

Engineering[1] has become a sub-discipline in the telecommunication area which comprises the design, validation, and implementation of communication protocols.

The crucial aspect in protocol development is to find an appropriate description of the protocol. Informal descriptions have proved inappropriate due to their ambiguity. Therefore formal descriptions based on formal semantic models are preferred. The use of **formal description techniques (FDTs)** for the design, validation, and implementation of communication protocols is the characteristic feature of *Protocol Engineering*. Formal description techniques guarantee a unique interpretation of the protocol specification. They are the foundation for the systematic development of communication protocols as well as distributed systems as an engineering discipline. They establish the basis for providing tools to support the different development stages and to automate parts of the development phases.

What are the features of communication protocols which justify the establishment of *Protocol Engineering* as an independent sub-discipline? These features relate to a number of particularities which characterize protocol development and distinguish it from traditional software development:

- The protocol notion used commonly comprises, exactly speaking, two concepts: that of the service and that of the protocol. The *service* denotes the result of the interaction between network components, when running a protocol. It can be used by other protocols or an application. The *protocol* describes how the service is provided. It is quasi its "implementation". This has consequences for the description of services and protocols. In contrast to the traditional software specification not only a *What*-specification, which describes the "service", is required, but also a *How*-specification is needed to specify how this service is provided, i.e., its "implementation".
- Protocol entities, which form the communicating partners in a protocol, must be capable of reacting simultaneously to different events and communication requirements, respectively. This can result in concurrent execution threads and the nondeterministic appearance of events.
- Unlike many other areas in computer science, the protocol area is characterized by (international) standards. Many protocols are defined as a standard. This is necessary so that the protocol can be implemented multiple times in different execution environments. The various implementations must be able to work together, i.e., they must be interoperable. Standardization usually cuts off the design stage from subsequent phases, such as verification, implementation, and testing. It also requires specific methods, such as the conformance test, to prove compliance with the standard.
- Protocols are subject to high demands on efficiency and reliability. The complexity of protocols is often contradictory to these requirements. The distributed nature of protocols results in complex state spaces in which design and

[1] The name was first introduced by Piatkowski in 1983 [Piat83].

implementation errors are difficult to find. To prove the correctness of the design and implementations special methods have to be applied.

- The development of protocols is expensive. It binds much manpower over a long period. To make this process more efficient tools are required which implement the protocol-specific development methods.

With regard to the existence of a world-wide communication infrastructure like the Internet, one might assume that the development of new protocols is not required any more. This is not true. Novel technological possibilities and innovative developments put new demands on the communication infrastructure and their protocols. In recent years many new protocols have appeared, in particular for wireless communication or peer-to-peer applications. With the development of new Internet applications and technologies the development of new protocols will continue in the future.

Objective of the book

The book is dedicated to the fundamentals of *Protocol Engineering*. It introduces the reader to the world of protocols, their basic principles, their description, and their development. The book considers both the theoretical and the practical aspects of *Protocol Engineering* and tries to link both parts which are often considered independently. At the same time it aims to point out the possibilities and limitations of the various methods. Last but not least, the book aims to encourage the reader to apply these methods in their practical work.

The book is primarily a book about formal description techniques for communication protocols and related methods. In the introductory part it presents the fundamentals of communication protocols as they are needed for further reading. The book is not a general introduction to computer networks; this is given in the well-known teaching books of Tanenbaum and Whetherall, Stallings, Peterson and Davie, and Kurose and Ross. It deals with a specific, but important area of the development of computer networks and telecommunication systems and thus it supplements the above mentioned books.

For whom is the book written?

This book is written for students and engineers of computer science and communication technology who want to introduce themselves to the field of communication protocols and their development. It also addresses specialists who want to deepen their knowledge of *Protocol Engineering* or look up applied methods. The book may be also of interest to software engineers who work on the development of distributed systems. Many of the presented methods for describing and validating protocols can also be applied to distributed systems.

The book does not require special knowledge of this topic. It is merely assumed that the reader possesses basic knowledge in computer networks and software engineering.

Structure of the book

The book consists of three parts. The first part contains the fundamentals of communication protocols. It describes the working principles of protocols and implicitly also those of computer networks. In this part we introduce the basic concepts *service*, *protocol*, *layer*, and *layered architecture*. Applying analogies from everyday life we try to familiarize the reader with the in-part-complicated procedures in protocols. In parallel, we introduce the basic elements for the description of protocols using a model language. Thereafter we present the most important protocol functions. Finally we give as a case study an overview of the TCP/IP protocol suite.

The second part of the book deals with the description of communication protocols. We give a comprehensive overview of the various methods and techniques for describing protocols. Beginning with the requirements on formal description techniques we first introduce the fundamental description methods which are in part used as semantic models for the formal description techniques. Thereafter we give an example of the various approaches of formal description techniques together with an overview of a representative language. The languages considered in this way are SDL-2000, MSC, LOTOS, cTLA, and ASN.1. We also give an outlook on the use of UML for describing protocols.

The third part presents the protocol life cycle and the most important development stages. We consider the following phases: design, specification, verification, performance evaluation, implementation, and testing and present the most relevant methods applied in these stages. The reader gets acquainted with approaches for a systematic protocol design, with basic techniques for the specification of protocols, with various verification methods, with the main implementation techniques, and with strategies for their testing, in particular with the conformance and the interoperability tests. In the testing chapter we also give an overview of the test description languages TTCN-2 and TTCN-3.

The connection between the three parts of the book is formed by the XDT (*eXample Data Transfer*) protocol. XDT is a simple data transfer example protocol which applies the *go back N* principle. It is used as a reference protocol throughout the book to exemplify the different description techniques as well as to demonstrate important validation and implementation approaches. This is supposed to give the reader the possibility to compare the different techniques and methods. The complete formal descriptions of the XDT protocol in the various formal description techniques are available at the web site of the book (see below).

How should the book be read?

The first part of the book deals with the fundamentals of communication protocols. It introduces the principles and the elements for describing protocols as they are presumed in the following two parts. This part addresses readers who want to familiarize themselves with basic protocol principles. Readers who have a good knowledge about protocols can omit this part. It is recommended, however, to read the short introduction to the XDT protocol in order to understand the reference examples in the later chapters.

The second part deals with the description of protocols. It introduces the basic description methods and the most important formal description techniques. The way protocols are described plays a central role in the protocol development process, since the selection of the description technique, in particular of the associated semantic model, determines the applied design and validation methods. For that reason, we separate the introduction of the formal description techniques from the description of the protocol development phases. We recommend the reader first concentrates on the description methods and the basic concepts of the description techniques he/she is most interested in.

The third part describes the development phases typical for communication protocols. Since communication protocols are seldom developed continuously, i.e., from design to implementation/installation, many protocol engineers specialize in certain phases, e.g., verification. Therefore, it is up to the readers how deeply they immerse themselves in the respective subject.

We provide for almost all chapters exercises which should help the reader to better understand the presented contents and to practice various description techniques.

Website for the book

A website is available to provide online materials and additional information on the subject of the book. These materials are available via the following URL:

http://www.protocol-engineering.tu-cottbus.de

The website contains among others:

- lecture materials related to the book including exercises
- the informal description of the XDT protocol
- an animated simulation of the XDT protocol to visualize the protocol behavior
- formal descriptions of the XDT protocol in the various FDTs presented in the book
- additional information and references.

Cottbus/Dresden, Germany Hartmut König

Autumn 2011

Contents