Understanding Petri Nets

Wolfgang Reisig

Understanding Petri Nets

Modeling Techniques, Analysis Methods, Case Studies

Springer

Wolfgang Reisig
Humboldt-Universität zu Berlin
Berlin, Germany

ISBN 978-3-642-33277-7 ISBN 978-3-642-33278-4 (eBook)
DOI 10.1007/978-3-642-33278-4

Library of Congress Control Number: 2013937989

ACM Computing Classification (1998): D.2, F.1, I.6

Translation from the German language edition: Petrinetze: Modellierungstechnik, Analysemethoden, Fallstudien by Wolfgang Reisig

© Vieweg + Teubner Verlag 2010. All rights reserved

© Springer-Verlag Berlin Heidelberg 2013
This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Printed on acid-free paper.

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

Zuerst einmal kommt es darauf an,
verstanden zu werden.

Theodor Fontane

Put it before them briefly so they will read it,
clearly so they will appreciate it,
picturesquely so they will remember it
and, above all, accurately so they will be
guided by its light.

Joseph Pulitzer

Progress is possible only if
we train ourselves to think about programs
without thinking of them
as pieces of executable code.

E.W. Dijkstra

Computer Science
is no more about computers
than astronomy
is about telescopes.

E.W. Dijkstra

Carl Adam Petri (1926 – 2010)

1988	Officer's Cross of the Order of Merit of the Federal Republic of Germany
1988	Honorary professor at the University of Hamburg, Germany
1989	Member of the Academia Europaea, London
1993	Konrad Zuse Medal of the German Informatics Society (Gesellschaft für Informatik) for excellent achievements in computer science
1997	Werner von Siemens Ring for distinguished achievements in technology and science
1997	Member of the New York Academy of Sciences
1999	Honorary doctorate of the University of Zaragoza
2003	Commander of the Order of the Netherlands Lion
2007	Gold Medal of Honor of the Academy of Transdisciplinary Learning and Advanced Studies
2009	IEEE Computer Pioneer Award

Foreword

Dear reader:
The currently common graphic representation for nets as well as the methodology of their refinement and abstraction was invented in August 1939 to visualize my knowledge about chemical processes. I also devised a plan to analyze catalytic processes.

This plan failed due to the beginning of the Second World War, because the necessary equipment was no longer available. So I had no other alternative but to study my invention on paper.

In 1941, I was so excited about my father's report on Konrad Zuse's programmable computer that I could not stop thinking about it. However, it was not until the late 1950s that it became evident that the development of computer technology would greatly influence our society.

It seemed to me that a theoretical framework was needed to formulate and solve the basic problems that arose for the structure and the sensible use of this computer technology – independent from the current state of the technology, but in accordance with the laws of physics. I decided to lay the foundations of this framework. Indeed, I remembered my toy from 1939 and applied it to signals and bits, but formulated my results predominantly in mathematical form to achieve a higher acceptance of my dissertation on "Communication with Automata" (original German: "Kommunikation mit Automaten"). (Only after my exam, did I reintroduce the net graphics.)

My arguments were:

- The theory of computability is based on the Turing machine, consisting of an abstract automaton and an infinite tape. While the automaton can be implemented by means of a computer, the infinite tape cannot.

- To avoid a contradiction to the laws of physics (finite, invariant speed of light and the uncertainty principle), the infinite ribbon has to be replaced with two short shifting registers. Attached to the end of each register is a register cell factory that works at least as fast as it takes to access the register.
- The cells of the register are only allowed to communicate with their two neighbors, just as each transition can only affect its immediate neighbors.
- That means that the computability theory current at that time, with its global state changes, cannot be applied to the kinds of distributed systems that the technology is developing towards. Trying to conceive the Internet as a single automaton is simply not useful.

Since then, many others have helped to develop the net theory of distributed systems into an established field of knowledge, with many computer tools for the study of system properties hidden within the graphical notation, like behavioral invariants or the potential disassembly into components.

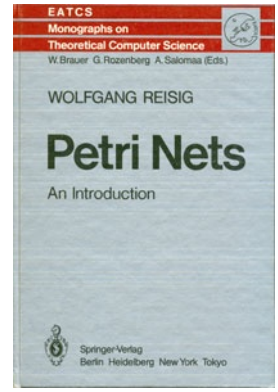
This book chooses the most important, most original, most successful and most basic concepts and presents them comprehensively. In doing so, it offers students and practitioners, in some cases even specialists of the field, an approach to or a new view on distributed systems.

I wish this book a wide distribution and a long usage.

St. Augustin, Germany, January 2008

Carl Adam Petri

Preface



Petri Nets – An Introduction [63] is a slim book that was published almost 30 years ago and was quickly translated into six languages and sold several thousand copies. For a long time, it was considered the standard reference for Petri nets.

Now it is time for a new introduction. Petri nets have been further developed in unbelievably diverse directions. "First model, then program" is a principle that increasingly dominates software engineering, and Petri nets are a popular modeling technique.

The biggest problem in writing an introduction to a topic is the selection of the content. Today we have a better understanding of which modeling techniques and analysis methods are truly central than we had 30 years ago. Those techniques and methods are presented here. Their usefulness is illustrated with the help of several examples, particularly the case studies in the last part of this book. The software engineer can use these case studies as a guide. At the same time, the theorist will find the classic results of Petri net theory, complemented with a few new concepts and a new taxonomy.

The formal arguments in this book are reduced to a minimum. Examples often have enough detail to sufficiently demonstrate the characteristics of the content. Furthermore, individual chapters can often be read independently from one another. For instance, the basics introduced in the first three chapters of Part I are enough to understand the case studies in Part III. Only their analysis requires knowledge of the methods presented in Part II.

This book was not written for a specific audience such as students, teachers, theoretical computer scientists or software engineers. Instead, it addresses a broad audience by compiling the central developments of the last 50 years of net theory and

practice and presenting it in a comprehensible way. The selection of aspects presented here is, therefore, highly subjective.

As a substantial extension, the English version also includes a concise compilation of the formal framework that is used throughout this book. For easy navigation and cross-reference, the relevant terms are highlighted in the margin of each chapter as well as in the formal framework itself.

Acknowledgments

This book was originally published in German. The German readers provided very valuable feedback and pointed out some inconsistencies in the terminology as well as various typos. Thanks to all of them.

From the late 1970s, Carl Adam Petri, in numerous conversations, explained to me his proposals for a basic theory of information processing systems. Without these conversations, this book would not have been possible. Jörg Desel contributed important arguments for the selection of the content. His advice influenced some chapters decisively. Since the early drafting phase, Rüdiger Valk pointed out numerous typing errors, inconsistencies, ambiguities and interesting citations to me. Before submission to the publisher, some colleagues proposed changes to the manuscript, which I gladly took up. In particular, Katharina Görlach, Jana Köhler, Bernd Krämer, Ernst Mayr, Agathe Merceron, Birger Peil, Christian Stahl, Hagen Völzer and Walter Vogler made valuable suggestions. I also thank Daniel Janusz, Andreas Hoheisel, Olaf Hochmuth, Rolf Hoffmann and Dominic Wist for pointing out inconsistencies. Annika Flemming made large contributions to the development of the exercises.

My special thanks to Birgit Heene. Patiently and tirelessly, she typed and arranged the numerous draft versions. Dirk Fahland, Katharina Görlach and Simon Heiden assisted her with great commitment.

Michael Rücker then took on the burden of translating the German text into English. I think he did a great job (also in putting up with my continuous criticism). My very special thanks goes to Tim Denvir, whose relentless scrutiny of the draft version found many mistakes and spawned various discussions on some amazing subtleties of the English language. With Ronan Nugent at Springer-Verlag, I shared numerous (and quite

humorous) conversations about the book's shape and layout. I am grateful that he accepted most of my wishes. He also organized yet another proof reading process and engaged himself personally.

Wolfgang Reisig, Berlin, February 2013.

Introduction

What Are Petri Nets and What Should They Be?

A central challenge of computer science is the appropriate construction of *systems* that contain IT-based components and that are embedded in an automated or organizational environment. Such systems are *modeled* so that clients, manufacturers, users, etc., can, with the help of the model, better understand what a system is supposed to do and how it can be implemented, used, varied and improved.

In the past 50 years, a multitude of modeling techniques has been proposed for such systems. They all have their strengths and weaknesses, as well as their preferred fields of application, user groups and tools. Petri nets are among the oldest modeling techniques of computer science.

For decades now, interesting theoretical questions about Petri nets have been posed and solved. Special subclasses have been studied and tools have been developed. Case studies have been conducted and successful projects have been implemented. In contrast to some other methods, which were favored for a short time and then forgotten, Petri nets have kept their place as one of the well-established modeling techniques. Initially, there was an interest in integrating Petri nets with other methods. However, over time, this has given way to a tendency to keep them separate.

With a lively, steadily growing group of interested researchers and users as well as ever-new software tools, for decades Petri nets have been in the best position to shape

some foreseeable developing lines of computer science and to contribute to novel concepts like "model-based," "ubiquitous," "pervasive" or "disappearing" software engineering. The Petri net portal www.informatik.uni-hamburg.de/TGI/PetriNets/index.html offers rich material on this.

Carl Adam Petri himself originally designed his approach even more broadly. He looked for a theory of information processing in accord with the laws of physics, with deep-seated invariants in the traditions of the natural sciences and with formalisms that would allow for the description of a human-oriented, pragmatic handling of the technical possibilities of computer science.

What Does This Book Cover?

The book is divided into three parts. The first part describes how to use Petri nets for modeling. All concepts are explained with the help of examples; at first, with different variants of a cookie vending machine. We use the most powerful, most generic type of Petri nets for this (with real-world objects as "tokens"). It is not until Chapter 3 that we will use the more specialized and technically simpler type of *elementary* system nets with abstract "black" tokens. I chose this order so as to start with an intuitively very convincing and realistic model. From there on, I always introduce derived concepts like *sequential* and *distributed* runs, *scenarios* and additional notations for elementary system nets first, because they are more comprehensible this way. The examples are meant to show that scenarios, in particular, decisively deepen our understanding of a system and deserve special attention. The synthesis problem, i.e., deriving non-sequential behavior from a description of sequential behavior, can be solved very convincingly with Petri nets. It is therefore covered in a separate chapter.

The second part introduces techniques with which important properties of system nets can be formulated as well as algorithms with which one can prove or disprove their validity. For easier understanding, some concepts are only introduced for elementary system nets at first. This second part covers all essential analysis methods that are specific to Petri nets, particularly traps, place invariants, transition invariants, covering graphs and special techniques for free-choice nets. Some read-

ers may be unfamiliar with the combination of the analytical power of traps and place invariants, the consistent distinction between "hot" and "cold" transitions, and the discussion of run properties. Temporal logic and particularly model checking offer generic analysis techniques, which are also successfully applied to Petri nets. This book will not explicitly cover such techniques, because they are not specific to Petri nets. However, ideas from temporal logic frequently influence this text implicitly, particularly the distinction between state and run properties.

The third part presents a selection of three case studies from very different fields. They show how diversely applicable Petri net models are. In addition, each case study introduces new, more generic concepts, properties and analysis techniques, which are also useful for very different modeling tasks. Among them are fairness properties, the combination of region theory and state properties and symbolic schemata for Petri nets.

In summary, the three parts of this book explain:

- Petri nets as a modeling technique,
- the theoretically well-founded analysis of Petri nets,
- case studies from very different areas of application.

Each chapter ends with exercises and recommendations for further reading. Particularly challenging exercises are marked with an asterisk. Texts with a colored background focus on historic or exemplary aspects of the main text. Finally, the appendix includes a compilation of the formal framework used throughout this book.

Conclusive Threads

The previous section has already mentioned the special case of *elementary* nets (with "black tokens"). Readers who only wish to study this type of net can follow the *elementary strand* of this book. Figure 1 shows which chapters and sections belong to this strand.

Readers who only wish to study the modeling (but not the analysis) of systems can follow the *modeling strand*. This strand contains all the models of real systems appearing in this book. Figure 2 outlines the chapters and sections of this strand.

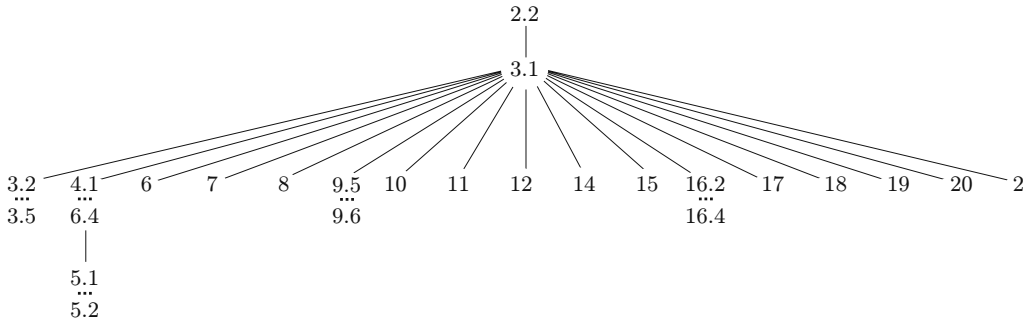


Figure 1 The elementary strand

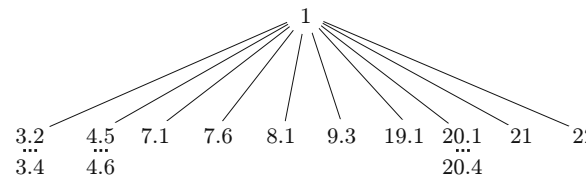


Figure 2 The modeling strand

Readers who are interested in technically simple, but challenging examples and case studies can choose the elementary models of the modeling strand, shown in Fig. 3.

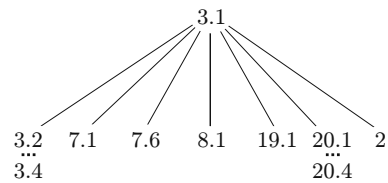


Figure 3 The elementary modeling strand

How Is the Content Presented?

Pertinent examples and intuitive descriptions often explain the content sufficiently well. In places where formal arguments are used, only basic standard notations are employed, for instance, for sets, functions, graphs, vectors, matrices, systems of equa-

tions and propositional logic. Everything beyond that (for instance, the handling of multisets) is explained in detail. The area of Petri nets has grown so rapidly that there is not yet any overall consistent and intuitively appealing terminology. This text constantly looks for compromises between consistency (identical names for identical things, different names for different things), intuitive comprehensibility and linguistic correctness of terms and notations. Some (few) terms are newly introduced here or are used in a new way. Readers who want to avoid formalism can still acquire a sufficient understanding of the content.

The page layout is designed to break up the text. The broad margin contains examples, annotations and alternative formulations. In detail, this means:

- The margin illustrates the text: the examples and annotations in the margin explain and illustrate the concepts in the text.
- The text explains the margin: readers who are (somewhat) familiar with Petri nets can often get along by only looking at the examples and annotations in the margin. The text then serves as confirmation and generalization.

Even without an analysis, Petri nets are a very interesting modeling technique. Chapters 1 and 3 are sufficient to understand the case studies in Part III.

When looking for a particular passage in the text, the reader can find help in the list of Examples and Case Studies in the Frontmatter or in the Index at the end of this book. Further information, feedback from readers and material for classes can be found at: https://u.hu-berlin.de/understandingpetrinets.

Contents

List of Examples and Case Studies xxvii

Part I Modeling Techniques

1 An Example 3
1.1 A Cookie Vending Machine 3
1.2 A Look Inside 4
1.3 The Interface 5
1.4 Hot and Cold Transitions 6
1.5 Runs 6
1.6 Alternatives 7
1.7 Fine Tuning 8
1.8 Diverse Components 10
Exercises 11
Further Reading 11

2 The Basic Concepts 13
2.1 A Variant of the Cookie Vending Machine 13
2.2 Components of a Net 14
2.3 The Data Structure for Petri Nets: Multisets 16
2.4 Markings as Multisets 18
2.5 Steps with Constant Arc Labelings 18
2.6 Steps with Variable Arc Labelings 19
2.7 System Nets 21
2.8 Marking Graph 22
2.9 Final Markings 23
Exercises 23
Further Reading 23

3 Common Special Case: Elementary System Nets 25
3.1 Elementary System Nets 25

3.2 An Abstract Model of the Cookie Vending Machine . . . 26
3.3 Mutual Exclusion . . . 27
3.4 The Crosstalk Algorithm . . . 28
3.5 1-Bounded Elementary System Nets . . . 32
Exercises . . . 32
Further Reading . . . 33

4 Sequential and Distributed Runs . . . 35
4.1 Sequential Runs . . . 35
4.2 Tokens as Labeled Places . . . 37
4.3 Actions . . . 37
4.4 Distributed Runs . . . 39
4.5 Example: A Bell Clock . . . 42
4.6 The Kindergarten Game . . . 43
4.7 Causal Order . . . 45
4.8 The Composition of Distributed Runs . . . 46
Exercises . . . 47
Further Reading . . . 48

5 Scenarios . . . 51
5.1 Defining Scenarios . . . 51
5.2 The Scenarios of the Crosstalk Algorithm . . . 53
5.3 The Scenarios of the Cookie Vending Machine . . . 54
Exercises . . . 55
Further Reading . . . 55

6 Further Notations for Elementary System Nets . . . 57
6.1 Place Capacities . . . 57
6.2 Arc Weights . . . 58
6.3 Real Extensions . . . 61
Exercises . . . 62
Further Reading . . . 62

7 The Synthesis Problem . . . 65
7.1 Example: The Light/Fan System . . . 65
7.2 The General Question of the Synthesis Problem . . . 67
7.3 Regions of State Automata . . . 68
7.4 The System Net of a State Automaton . . . 69
7.5 The Solution to the Synthesis Problem . . . 70
7.6 The Synthesis Problem of the Light/Fan State Automaton . . . 70
Exercises . . . 71
Further Reading . . . 72

8 Composition of Nets . . . 75
8.1 Nets with Interfaces . . . 75
8.2 Communicating Nets . . . 78
8.3 Unambiguous Decomposition into Open Nets . . . 78
Exercises . . . 80
Further Reading . . . 80

Part II Analysis Methods

9 State Properties . . . 83
9.1 Equations and Inequalities of the Cookie Vending Machine . . . 83
9.2 Valid Equations . . . 86
9.3 Example: Dining Philosophers . . . 86
9.4 Valid Inequalities . . . 88
9.5 Equations and Inequalities of Elementary System Nets . . . 88
9.6 Modulo Equations . . . 90
9.7 Propositional State Properties . . . 91
Exercises . . . 93
Further Reading . . . 94

10 Traps and Cotraps of Elementary System Nets . . . 97
10.1 Traps of Elementary System Nets . . . 97
10.2 Cotraps . . . 98
10.3 The Trap/Cotrap Property . . . 99
Exercises . . . 100
Further Reading . . . 101

11 Place Invariants of Elementary System Nets . . . 103
11.1 Vector Representation for Elementary System Nets . . . 103
11.2 The Matrix $\underline{N}$ . . . 104
11.3 Place Invariants . . . 104
11.4 Positive Place Invariants . . . 107
Exercises . . . 108
Further Reading . . . 109

12 Combining Traps and Place Invariants of Elementary System Nets . . . 111
12.1 Calculating with Equations and Inequalities . . . 111
12.2 State Properties of the Mutual Exclusion System . . . 113
12.3 State Properties of the Crosstalk Algorithm . . . 113
12.4 Unstable Properties . . . 114
Exercises . . . 116

13 Traps and Place Invariants of Generic System Nets 119
13.1 Traps of a System Net 119
13.2 Sum Expressions 120
13.3 Multiplying Sum Expressions 120
13.4 Applying a Sum Expression to a Multiset 121
13.5 The Matrix $\underline{N}$ of a System Net N 121
13.6 The Place Invariants of a System Net 122
13.7 The Constant of a Place Invariant 122
13.8 The Equation of a Place Invariant 124
13.9 Properties of the Philosophers System 124
13.10 Properties of the Kindergarten Game 126
Exercises 127
Further Reading 127

14 Marking and Covering Graphs 131
14.1 Deriving Properties from the Marking Graph 131
14.2 The Idea of the Covering Graph 132
14.3 ω-Markings 133
14.4 The Construction of the Covering Graph 134
14.5 The Finiteness of the Covering Graph 135
14.6 The Covering of Sequential Runs 136
14.7 Simultaneously Unbounded Places 136
14.8 Dead Transitions 138
14.9 Covering Graphs of Generic System Nets 138
Exercises 138
Further Reading 139

15 Reachability in Elementary System Nets 141
15.1 Corollaries of Place Invariants 141
15.2 The Marking Equation 142
15.3 Transition Invariants 144
Exercises 145
Further Reading 145

16 Run Properties 147
16.1 Intuitive Question 147
16.2 Defining Run Properties 148
16.3 The Deduction Rule 150
16.4 Proof Graphs 153
Exercises 155
Further Reading 155

17 Free-Choice Nets . . . 157
17.1 Defining Free-Choice Nets . . . 157
17.2 The Trap/Cotrap Theorem for Free-Choice Nets . . . 158
17.3 Clusters . . . 158
17.4 The Rank Theorem . . . 160
Exercises . . . 160
Further Reading . . . 161

18 Marked Graphs . . . 163
18.1 Defining Marked Graphs . . . 163
18.2 Liveness of Marked Graphs . . . 164
18.3 1-Bounded Marked Graphs . . . 165
18.4 Liveness of 1-Bounded Marked Graphs . . . 165
Exercises . . . 166
Further Reading . . . 166

19 Well-Formed System Nets . . . 169
19.1 Example: Models of Business Processes . . . 169
19.2 Well-Formed Elementary System Nets . . . 170
19.3 Deciding Well-Formedness . . . 170
Exercises . . . 171
Further Reading . . . 171

Part III Case Studies

20 Mutual Exclusion . . . 175
20.1 The Problem . . . 175
20.2 Realizability . . . 176
20.3 Fairness Assumptions . . . 177
20.4 Mutex with Autonomous Fairness . . . 178
20.5 The Scenarios of the Algorithm . . . 179
20.6 Correctness of the Algorithm . . . 181
Exercises . . . 183
Further Reading . . . 183

21 Asynchronous Hardware . . . 185
21.1 The Counterflow Pipeline Processor (CFPP): The Problem . . . 185
21.2 The Solution Idea . . . 186
21.3 The Synthesis Problem for the CFPP . . . 187
21.4 Structural Simplification of a Module . . . 188
21.5 The Model of the CFPP . . . 189
21.6 Analysis of the Model . . . 190

Exercises 191
Further Reading 191

22 Network Algorithms 193
22.1 Some Conventions for the Representation of Network Algorithms 194
22.2 The Echo Algorithm 194
22.3 Synchronization in Acyclic Networks 198
22.4 Consensus in the Network 201
Exercises 204
Further Reading 204

Part IV Conclusion

23 Closing Remarks 207
23.1 A Brief History of Petri Nets 207
23.2 Properties of the Elementary Formalisms of Petri Nets 207
23.3 Speculative Questions 209
23.4 Petri Nets in Software Engineering 209
23.5 Reference to Other System Models and Analysis Techniques 210
23.6 Other Introductory Texts 210

Formal Framework 213

Bibliography 225

Index 229

List of Examples and Case Studies

Simple Cookie Vending Machine – Sect. 1.7, Fig. 1.10 9
Cookie Vending Machine With Two Kinds of Packets – Sect. 2.1, Fig. 2.1 13
The Net Structure of the Cookie Vending Machine – Sect. 2.1, Fig. 2.2 15
Abstract Cookie Vending Machine – Sect. 3.2, Fig. 3.1 26
Mutual Exclusion – Sect. 3.3, Fig. 3.2 27
Crosstalk Algorithm – Sect. 3.4, Fig. 3.7 31
Minute Clock – Sect. 4.5, Fig. 4.9 42
Bell Clock – Sect. 4.5, Fig. 4.10 43
Kindergarten Game – Sect. 4.6, Fig. 4.13 44
Refilled Simple Cookie Vending Machine – Sect. 6.2, Fig. 6.2 59
Light/Fan System – Sect. 7.1, Fig. 7.2 66
Customer, Salesperson and Stock as Open Nets – Sect. 8.1, Fig. 20.1 76
Five Philosophers – Sect. 9.3, Fig. 9.1 87
Crosstalk With Counters – Sect. 12.3, Fig. 12.2 114
Business Process: Develop an Offer – Sect. 19.1, Fig. 19.1 170
Message-Based Mutex – Sect. 20.4, Fig. 20.4 179
Counterflow Pipeline Processor CFPP – Sect. 21.6, Fig. 21.7 190
Echo Algorithm – Sect. 22.2, Fig. 22.6 197
Synchronization in Acyclic Networks – Sect. 22.3, Fig. 22.10 200
Consensus Algorithm – Sect. 22.4, Fig. 22.13 203