

大学计算机教育国外著名教材系列 (影印版)

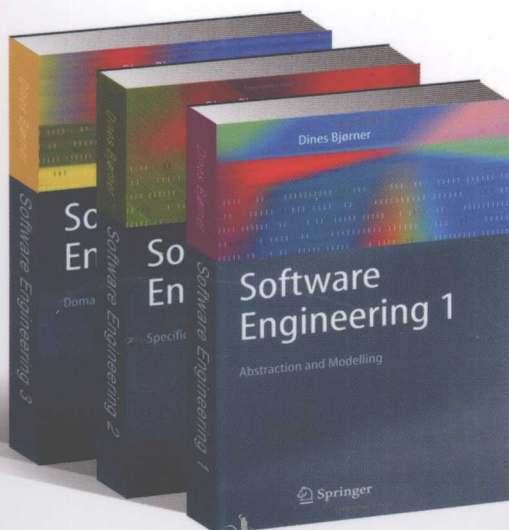
SOFTWARE ENGINEERING 2

SPECIFICATION OF SYSTEMS AND LANGUAGES

软件工程 系统与语言规约

卷2

Dines Bjørner 著



清华大学出版社

大学计算机教育国外著名教材系列（影印版）

Software Engineering 2

Specification of Systems and Languages

软件工程 卷2

系统与语言规约

English reprint edition copyright © 2007 by Springer-Verlag and TSINGHUA UNIVERSITY PRESS.

Original English language title: Software Engineering 2: Specification of Systems and Languages by Dines Bjørner, Copyright © 2006 Springer Science + Business Media, Inc.

All Rights Reserved.

This edition has been authorized by Springer-Verlag (Berlin/Heidelberg/New York) for sale in the People's Republic of China only and not for export therefrom.

本书影印版由 Springer-Verlag 授权给清华大学出版社出版发行。

北京市版权局著作权合同登记号 图字 01-2007-0350 号

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

软件工程卷 2: 系统与语言规约=Software Engineering 2: Specification of Systems and Languages: 英文/(丹) 比约纳 (Bjørner, D.) 著. —影印本. —北京: 清华大学出版社, 2007. 7

(大学计算机教育国外著名教材系列)

ISBN 978-7-302-15432-7

I. 软… II. 比… III. 软件工程—高等学校—教材—英文 IV. TP311.5

中国版本图书馆 CIP 数据核字 (2007) 第 086863 号

责任印制: 李红英

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

c-service@tup.tsinghua.edu.cn

社 总 机: 010-62770175

投稿咨询: 010-62772015

地 址: 北京清华大学学研大厦

邮 编: 100084

邮购热线: 010-62786544

客户服务: 010-62776969

印 刷 者: 清华大学印刷厂

装 订 者: 三河市漂源装订厂

发 行 者: 全国新华书店

开 本: 185×230 印张: 50.5

版 次: 2007 年 7 月第 1 版 2007 年 7 月第 1 次印刷

印 数: 1~3000

定 价: 79.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题, 请与清华大学出版社出版部联系调换。联系电话: 010-62770177 转 3103 产品编号: 023296-01

出版说明

进入 21 世纪, 世界各国的经济、科技以及综合国力的竞争将更加激烈。竞争的中心无疑是对人才的竞争。谁拥有大量高素质的人才, 谁就能在竞争中取得优势。高等教育, 作为培养高素质人才的事业, 必然受到高度重视。目前我国高等教育的教材更新较慢, 为了加快教材的更新频率, 教育部正在大力促进我国高校采用国外原版教材。

清华大学出版社从 1996 年开始, 与国外著名出版公司合作, 影印出版了“大学计算机教育丛书(影印版)”等一系列引进图书, 受到国内读者的欢迎和支持。跨入 21 世纪, 我们本着为我国高等教育教材建设服务的初衷, 在已有的基础上, 进一步扩大选题内容, 改变图书开本尺寸, 一如既往地请有关专家挑选适用于我国高校本科及研究生计算机教育的国外经典教材或著名教材, 组成本套“大学计算机教育国外著名教材系列(影印版)”, 以飨读者。深切期盼读者及时将使用本系列教材的效果和意见反馈给我们。更希望国内专家、教授积极向我们推荐国外计算机教育的优秀教材, 以利我们把“大学计算机教育国外著名教材系列(影印版)”做得更好, 更适合高校师生的需要。

清华大学出版社

中 文 序

信息化社会的发展越来越依赖于软件。如何在期望的时限内、以可承受的开销、开发具有满意质量的软件成为人们特别是软件开发人员所关心的问题。自 1969 年 NATO 会议首次提出“软件工程”一词以来,软件工程已成为了一门学科。IEEE 的 SWEBOK 和 SEEK,以及 ACM/IEEE 的计算教程 2005 相继发布,标志着软件工程的学科内涵及其教育逐步走向规范与成熟。

软件形式化开发方法是软件工程的重要组成,荟萃了软件工程在原理和科学上的许多精华,被认为是开发可靠、安全软件的重要途径。Dines Bjørner 教授所著的《软件工程》(卷 1~3)系统地介绍了软件形式化开发方法,包括数学基础、形式化模型与建模、形式化规约、领域与需求分析以及软件设计等内容。该书对如何运用形式化方法进行软件开发进行了比较全面的阐述。Dines 是软件工程领域的著名专家,是著名的 VDM 方法和 RAISE 方法的创建者之一,因其在形式化软件开发方法及其在工业界的应用的突出贡献和倡导作用而获选 ACM Fellow 和 IEEE Fellow。这本书是他长期研究和实践的总结和提炼,内容丰富,想必会令中国的软件工程人员开卷有益。

很高兴得知《软件工程》(卷 1~3)的英文影印版和中文翻译版即将出版,应老友之邀,提笔作序,以资庆贺。

陈火旺
于梦泽园

Preface

The present volume is the second of three volumes on the engineering principles and techniques of software engineering. We refer to the Preface of Vol. 1, as well as to Chap. 1 of that same volume, for a proper preface and overall introduction to all volumes in this series. We assume that the reader has studied Vol. 1.

Overview

The present volume focuses on principles and techniques for specifying languages and systems. It uses the abstraction and modelling principles, techniques and tools covered in Vol. 1, and it supplements those principles, techniques and tools with additional ones. In particular the present volume emphasises the following four aspects:

- advanced specification facets:
 - ★ hierarchies and composition Chap. 2
 - ★ denotations and computations Chap. 3
 - ★ configurations: contexts and states Chap. 4
 - ★ time, space and space/time Chap. 5
 - ★ modularisation and Chap. 10
 - ★ automata and machines Chap. 11
- linguistics:
 - ★ pragmatics Chap. 6
 - ★ semantics Chap. 7
 - ★ syntax and Chap. 8
 - ★ semiotics Chap. 9
- concurrency and temporality:
 - ★ Petri nets Chap. 12
 - ★ message sequence charts and live sequence charts Chap. 13
 - ★ statecharts and Chap. 14

★ quantitative models of time	Chap. 15
• interpreter and compiler definitions:	
★ applicative programming languages	Chap. 16
★ imperative programming languages	Chap. 17
★ modular programming languages and	Chap. 18
★ parallel programming languages	Chap. 19

“UML”-ising Formal Techniques

Some notable features should be emphasised here. The concurrency aspect, Chaps. 12–14, also illustrates diagrammatic specifications, as does Sect. 10.3 (UML class diagrams). Together this material illustrates that popular features of the Unified Modeling Language (UML [59, 237, 382, 440]) can simply and elegantly be included, i.e., used, with RSL. Christian Krog Madsen is the main author of Chaps. 12–14.

The RAISE Specification Language: RSL

As in Vol. 1, we use RSL extensively in the present volume. Hence we insert, in Chap. 1, an RSL Primer — and otherwise refer to the RAISE URL: <http://www.iist.unu.edu/raise/>.

Acknowledgments

The preface of Vol. 1 contained an extensive acknowledgment section.

Combining RSL with Petri nets (Chap. 12), with message or live sequence charts (Chap. 13), and with statecharts (Chap. 14) is due primarily to Christian Krog Madsen [316, 317]. Very many and dear thanks are therefore extended to Christian. Combining RSL with UML-like class diagrams (Sect. 10.3) is due primarily to Steffen Holmslykke [9, 10]. Similar thanks are therefore graciously extended to Steffen. Martin Pěnička is likewise dearly acknowledged for having provided Examples 12.8 (Sect. 12.3.4), 14.7 (Sect. 14.4.1) and 14.8 (Sect. 14.4.2).

Colleagues at the National University of Singapore, Andrei Stefan and Yang ShaoFa, studied and proofread Chaps. 12–15. It was with Yang ShaoFa, as the leading person, that I decided to work out the model of CTP (Communicating Transaction Processes) of Sect. 13.6. Their comments and work are much appreciated.

A main source of academic joy has been the 30 years I have been at the Technical University of Denmark, 1976 till now.

Last, but not least, I acknowledge the tremendous support received from the Springer editor, Ronan Nugent.

Brief Guide to Volume 2

This volume has several chapters. The chapters are grouped into parts. Figure 2 abstracts a precedence relation between chapters. It is one that approximates suggested sequences of studying this volume.

- Chapter 1 is considered a prerequisite for the study of any chapter.
- We group some parts into the dash-circled groups *A* – *E*.
- Group *A* consists of Chaps. 2–5 that can be studied in any order.
- Group *B* consists of Chaps. 6–9 that should be studied in order 6, 7, 8 and 9.
- Group *C* consists of Chaps. 10–11 that can be studied in any order.
- Group *D* consists of Chaps. 12–15 that can be studied in almost any order. Chap. 14 does contain an example which requires having studied Chap. 13 first.
- Group *E* consists of Chaps. 16–19 that should be studied in order 16, 17 and 18. Chap. 19 can be studied in-between, before, or after. Preferably after.
- Groups *A* – *E* can be studied in any order. But it might be useful to have studied Chap. 5 before Chap. 15, and Chap. 10 before Chap. 18, and to have studied Group *B* before Group *E*.
- It is no harm to study Chap. 20.
- Appendix A contains an overview of our naming convention.

Within most chapters many sections can be skipped. Typically those with larger examples or towards the end of the chapters.

In this way a teacher or a reader can compose a number of suitable courses and studies. Some such are suggested in Fig. 1.

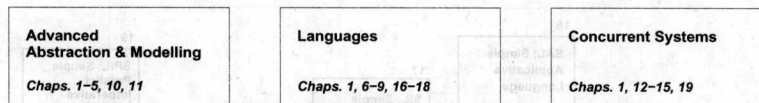


Fig. 1 Alternative courses based solely on Vol. 2

Dines Bjørner
 Technical University of Denmark, 2005–2006

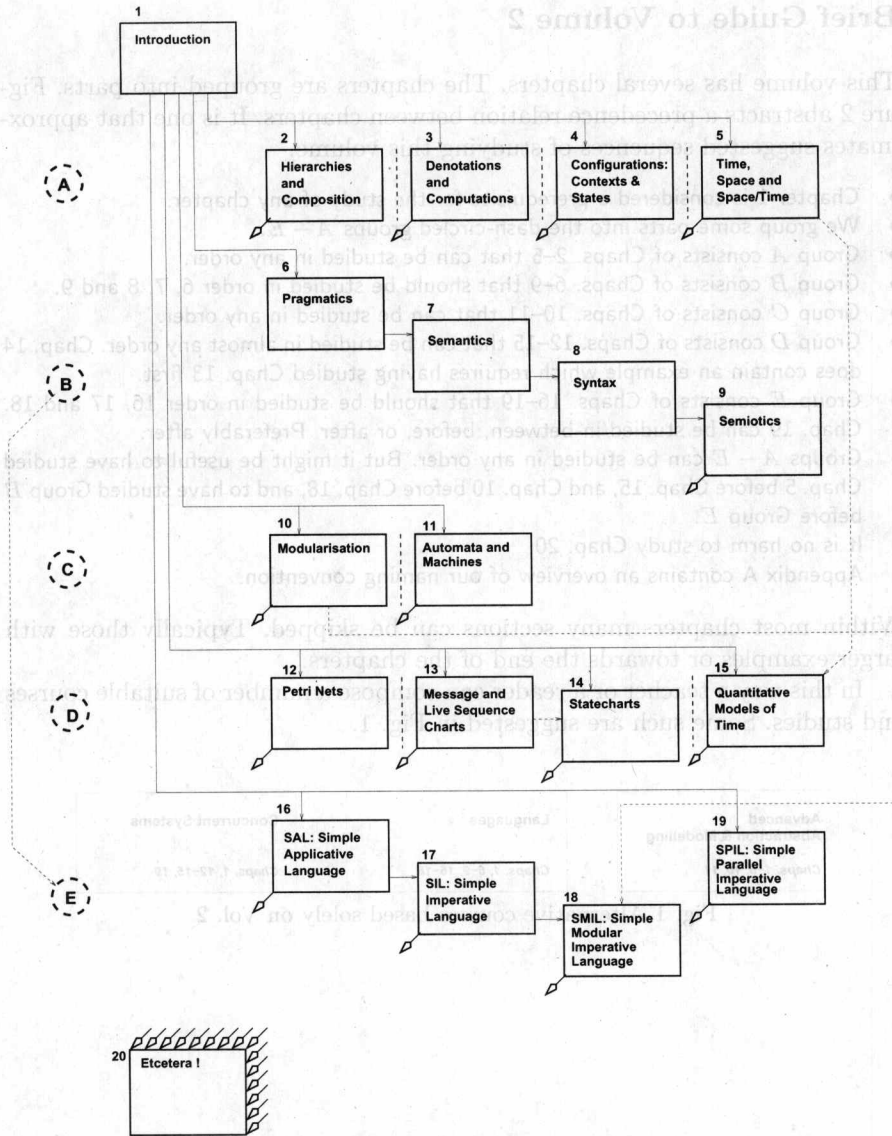


Fig. 2 Chapter precedence graph

Part II SPECIFICATION FACETS

Contents

PREFACE	VII
Overview	VII
“UML”-ising Formal Techniques	VIII
The RAISE Specification Language: RSL	VIII
Acknowledgments	VIII
Brief Guide to Volume 2	IX

Part I OPENING

1 Introduction	3
1.1 Introduction	3
1.1.1 Why This Volume?	3
1.1.2 Why Master These Principles, Techniques and Tools?	4
1.1.3 What Does This Volume “Contain”?	4
1.1.4 How Does This Volume “Deliver”?	5
1.2 Formal Techniques “Lite”	6
1.3 An RSL Primer	8
1.3.1 Types	8
1.3.2 The RSL Predicate Calculus	11
1.3.3 Concrete RSL Types	12
1.3.4 λ -Calculus+Functions	21
1.3.5 Other Applicative Expressions	24
1.3.6 Imperative Constructs	26
1.3.7 Process Constructs	27
1.3.8 Simple RSL Specifications	29
1.4 Bibliographical Notes	29

Part II SPECIFICATION FACETS

2	Hierarchies and Compositions	35
2.1	The Issues	35
2.1.1	Informal Illustrations	36
2.1.2	Formal Illustrations	36
2.2	Initial Methodological Consequences	37
2.2.1	Some Definitions	37
2.2.2	Principles and Techniques	38
2.3	The Main Example	40
2.3.1	A Hierarchical, Narrative Presentation	40
2.3.2	A Hierarchical, Formal Presentation	42
2.3.3	A Compositional, Narrative Presentation	45
2.3.4	A Compositional, Formal Presentation	47
2.4	Discussion	49
2.5	Bibliographical Notes: Stanisław Leshniewski	49
2.6	Exercises	50
3	Denotations and Computations	55
3.1	Introduction	55
3.1.1	Computations and Denotations	56
3.1.2	Syntax and Semantics	56
3.1.3	Characterisations	56
3.2	Denotational Semantics	57
3.2.1	A Simple Example: Numerals	57
3.2.2	The Denotational Principle	58
3.2.3	Expression Denotations	58
3.2.4	GOTO Continuations	62
3.2.5	Discussion of Denotational Semantics	72
3.3	Computational Semantics	74
3.3.1	The Issues	74
3.3.2	Two Examples	74
3.3.3	Expression Computations	74
3.3.4	Computational Semantics of GOTO Programs	78
3.3.5	Computational Semantics of Coroutine Programs	83
3.3.6	Discussion	85
3.4	Review: Denotations and Computations	86
3.5	Some Pioneers of Semantics	86
3.5.1	John McCarthy	86
3.5.2	Peter Landin	88
3.6	Exercises	90

4	Configurations: Contexts and States	93
4.1	Introduction	94
4.2	The Issues	97
4.3	"Real-World" Contexts and States	98
4.3.1	A Physical System: Context and State	99
4.3.2	Configurations of Contexts and States	99
4.3.3	Nonphysical System: Context and State	100
4.3.4	Discussion, I	101
4.3.5	Discussion, II	102
4.4	First Summary: Contexts and States	102
4.4.1	General	102
4.4.2	Development Principles and Techniques	103
4.5	Programming Language Configurations	104
4.6	Concurrent Process Configurations	104
4.6.1	The Example	104
4.6.2	Summary	110
4.7	Second Summary: Contexts and States	111
4.8	Information States and Behaviour States	112
4.8.1	Program Flowcharts as State Machine Data	112
4.8.2	Flowcharts $\equiv$ Machines	113
4.8.3	Flowchart Machines	113
4.8.4	Observations	114
4.8.5	Conclusion	114
4.9	Final Summary: Contexts and States	115
4.10	Exercises	116

Part III A CRUCIAL DOMAIN AND COMPUTING FACET

5	Time, Space and Space/Time	121
5.1	Time	122
5.1.1	Time — The Basics	122
5.1.2	Time — General Issues	124
5.1.3	"A-Series" and "B-Series" Models of Time	125
5.1.4	A Continuum Theory of Time	125
5.1.5	Temporal Events	126
5.1.6	Temporal Behaviour	127
5.1.7	Representation of Time	127
5.1.8	Operations "on" Time	128
5.2	Space	129
5.2.1	Space — The Basics	129
5.2.2	Location-Varying Entities	129
5.2.3	Locations and Dynamicity	131
5.2.4	Space — General Issues	132
5.3	Space/Time	135

XIV Contents

5.3.1	A Guiding Example	135
5.3.2	Representation of Space/Time	135
5.3.3	Blizard's Theory of Time-Space	136
5.4	Discussion	137
5.5	Bibliographical Notes	137
5.6	Exercises	137

Part IV LINGUISTICS

6	Pragmatics	145
6.1	Introduction	145
6.2	Everyday Pragmatics	146
6.3	"Formal" Pragmatics	146
6.4	Discussion	147
6.4.1	General	147
6.4.2	Principles and Techniques	148
6.5	Bibliographical Note	148
6.6	Exercises	149
7	Semantics	151
7.1	Introduction	151
7.2	Concrete Semantics	152
7.3	"Abstract" Semantics	152
7.4	Preliminary Semantics Concepts	152
7.4.1	Syntactic and Semantic Types	153
7.4.2	Contexts	153
7.4.3	States	154
7.4.4	Configurations	154
7.4.5	Interpretation, Evaluation and Elaboration	154
7.5	Denotational Semantics	155
7.5.1	Simple Case	156
7.5.2	Composite Case	156
7.6	Macro-expansion Semantics	157
7.6.1	Rewriting	157
7.6.2	Macro-expansion	158
7.6.3	Inductive Rewritings	158
7.6.4	Fix Point Evaluation	161
7.7	Operational and Computational Semantics	161
7.7.1	Stack Semantics	162
7.7.2	Attribute Grammar Semantics	162
7.8	Proof Rule Semantics	166
7.9	Discussion	169
7.9.1	General	169
7.9.2	Principles, Techniques and Tools	169

7.10	Bibliographical Notes	170
7.11	Exercises	170
8	Syntax	173
8.1	The Issues	174
8.1.1	Form and Content: Syntax and Semantics	174
8.1.2	Structure and Contents of This Chapter	175
8.2	Sentential Versus Semantical Structures	175
8.2.1	General	175
8.2.2	Examples of Sentential Structures	176
8.2.3	Examples of Semantical Structures	178
8.3	The First Abstract Syntax, John McCarthy	181
8.3.1	Analytic Grammars: Observers and Selectors	182
8.3.2	Synthetic Grammars: Generators	182
8.4	BNF Grammars $\approx$ Concrete Syntax	183
8.4.1	BNF Grammars	183
8.4.2	BNF $\leftrightarrow$ RSL Parse Trees Relations	184
8.5	Structure Generators and Recognisers	186
8.5.1	Context-Free Grammars and Languages	186
8.5.2	Parse Trees	188
8.5.3	Regular Expressions and Languages	189
8.5.4	Language Recognisers	190
8.6	XML: Extensible Markup Language	190
8.6.1	An Example	191
8.6.2	Discussion	192
8.6.3	Historical Background	192
8.6.4	The Current XML "Craze"	193
8.6.5	XML Expressions	193
8.6.6	XML Schemas	195
8.6.7	References	197
8.7	Abstract Syntaxes	197
8.7.1	Abstract Syntax of a Storage Model	197
8.7.2	Abstract Syntaxes of Other Storage Models	200
8.8	Converting RSL Types to BNF	202
8.8.1	The Problem	202
8.8.2	A Possible Solution	202
8.9	Discussion of Informal and Formal Syntax	203
8.9.1	General	203
8.9.2	Principles, Techniques and Tools	204
8.10	Bibliographical Notes	204
8.11	Exercises	205

9 Semiotics	213
9.1 Semiotics = Syntax $\oplus$ Semantics $\oplus$ Pragmatics	213
9.2 Semiotics	214
9.3 Language Components	215
9.4 Linguistics	216
9.5 Languages and Systems	217
9.5.1 Professional Languages	218
9.5.2 Metalanguages	219
9.5.3 Systems	219
9.5.4 System Diagram Languages	232
9.5.5 Discussion of System Concepts	232
9.5.6 Systems as Languages	233
9.6 Discussion	233
9.6.1 General	233
9.6.2 Principles, Techniques and Tools	234
9.7 Charles Sanders Peirce	234
9.8 Bibliographical Notes	234
9.9 Exercises	235

Part V FURTHER SPECIFICATION TECHNIQUES

10 Modularisation	243
10.1 Introduction	244
10.1.1 Some Examples	244
10.1.2 Preparatory Discussion	249
10.1.3 Structure of Chapter	252
10.2 RSL Classes, Objects and Schemes	253
10.2.1 Introducing the RSL "class" Concept	253
10.2.2 The RSL "class" Concept	257
10.2.3 The RSL "object" Concept	257
10.2.4 The RSL "scheme" Concept	257
10.2.5 RSL "scheme" Parameterisation	263
10.2.6 A "Large-Scale" Example	265
10.2.7 Definitions: Class, Scheme and Object	270
10.3 UML and RSL	271
10.3.1 Overview of UML Diagrams	271
10.3.2 Class Diagrams	272
10.3.3 Class Diagrams	273
10.3.4 Example: Railway Nets	276
10.3.5 Comparison of UML and RSL OO Constructs	278
10.3.6 References	279
10.3.7 Class Diagram Limitations	280
10.4 Discussion	280
10.4.1 Modularity Issues	280

10.4.2	Principles, Techniques and Tools	281
10.5	Bibliographical Notes	282
10.6	Exercises	282
11	Automata and Machines	285
11.1	Discrete State Automata	286
11.1.1	Intuition	287
11.1.2	Motivation	288
11.1.3	Pragmatics	288
11.2	Discrete State Machines	290
11.3	Finite State Automata	291
11.3.1	Regular Expression Language Recognisers	292
11.3.2	Regular Expressions	293
11.3.3	Formal Languages and Automata	294
11.3.4	Automaton Completion	295
11.3.5	Nondeterministic Automata	295
11.3.6	Minimal State Finite Automata	296
11.3.7	Finite State Automata Formalisation, I	297
11.3.8	Finite State Automata Realisation, I	297
11.3.9	Finite State Automaton Formalisation, II	298
11.3.10	Finite State Automata Realisation, II	299
11.3.11	Finite State Automata — A Summary	299
11.4	Finite State Machines	300
11.4.1	Finite State Machine Controllers	300
11.4.2	Finite State Machine Parsers	303
11.4.3	Finite State Machine Formalisation	304
11.4.4	Finite State Machine Realisation	305
11.4.5	Finite State Machines — A Summary	306
11.5	Pushdown Stack Devices	307
11.5.1	Pushdown Stack Automata and Machines	307
11.5.2	Formalisation of Pushdown Stack Machines	309
11.5.3	Pushdown Stack Device Summary	310
11.6	Bibliographical Notes: Automata and Machines	311
11.7	Exercises	311

Part VI CONCURRENCY AND TEMPORALITY

12	Petri Nets	315
Christian Krog Madsen is chief author of this chapter		
12.1	The Issues	315
12.2	Condition Event Nets (CENs)	316
12.2.1	Description	316
12.2.2	Small CEN Examples	317
12.2.3	An RSL Model of Condition Event Nets	320

12.3	Place Transition Nets (PTNs)	323
12.3.1	Description	323
12.3.2	Small PTN Examples	324
12.3.3	An RSL Model of Place Transition Nets	324
12.3.4	Railway Domain Petri Net Examples	328
12.4	Coloured Petri Nets (CPNs)	333
12.4.1	Description	333
12.4.2	A CPN Example	336
12.4.3	An RSL Model of Coloured Petri Nets	336
12.4.4	Timed Coloured Petri Nets	341
12.5	CEN Example: Work Flow System	342
12.5.1	Project Planning	342
12.5.2	Project Activities	346
12.5.3	Project Generation	353
12.6	CPN and RSL Examples: Superscalar Processor	356
12.6.1	Description	356
12.6.2	Coloured Petri Net Model	357
12.6.3	RSL Model: Superscalar Processor	362
12.7	Discussion	371
12.8	Bibliographical Notes	372
12.9	Exercises	372
13	Message and Live Sequence Charts	375
	Christian Krog Madsen is chief author of this chapter	
13.1	Message Sequence Charts	376
13.1.1	The Issues	376
13.1.2	Basic MSCs (BMSCs)	376
13.1.3	High-Level MSCs (HMSCs)	383
13.1.4	An RSL Model of HMSC Syntax	385
13.1.5	MSCs Are HMSCs	385
13.1.6	Syntactic Well-formedness of MSCs	386
13.1.7	An Example: IEEE 802.11 Wireless Network	391
13.1.8	Semantics of Basic Message Sequence Charts	400
13.1.9	Semantics of High-Level Message Sequence Charts	401
13.2	Live Sequence Charts: Informal Presentation	402
13.2.1	Live Sequence Chart Syntax	402
13.2.2	A Live Sequence Chart Example, I	408
13.3	Process Algebra	409
13.3.1	The Process Algebra PA_ϵ	410
13.3.2	Semantics of PA_ϵ	416
13.3.3	The Process Algebra PAC_ϵ	420
13.3.4	Semantics for PAC_ϵ	423
13.4	Algebraic Semantics of Live Sequence Charts	427
13.4.1	Textual Syntax of Live Sequence Charts	427
13.4.2	Semantics of Live Sequence Charts	428