

Software Engineering 1

Abstraction and Modelling

Bearbeitet von
Dines Bjørner

1. Auflage 2005. Buch. xl, 714 S. Hardcover

ISBN 978 3 540 21149 5

Format (B x L): 15,5 x 23,5 cm

Gewicht: 2710 g

[Weitere Fachgebiete > EDV, Informatik > Programmiersprachen: Methoden > Prozedurorientierte Programmierung](#)

Zu [Leseprobe](#)

schnell und portofrei erhältlich bei

beck-shop.de
DIE FACHBUCHHANDLUNG

Die Online-Fachbuchhandlung beck-shop.de ist spezialisiert auf Fachbücher, insbesondere Recht, Steuern und Wirtschaft. Im Sortiment finden Sie alle Medien (Bücher, Zeitschriften, CDs, eBooks, etc.) aller Verlage. Ergänzt wird das Programm durch Services wie Neuerscheinungsdienst oder Zusammenstellungen von Büchern zu Sonderpreisen. Der Shop führt mehr als 8 Millionen Produkte.

Contents

Preface	VII
Reasons for Writing These Volumes	IX
Shortcomings of These Volumes	X
Methods of Approach	XII
A New Look at Software	XII
Formal Techniques “Light”	XIII
The “Super Programmer”	XIV
What Is Software Engineering?	XV
The Author’s Aspirations	XVI
Role of These Volumes in an SE Education Programme	XVI
Why So Much Material?	XX
How to Use These Volumes in a Course	XX
Brief Guide to the Book	XXI
Guide to This Volume	XXI
Acknowledgments	XXIII

Part I OPENING

1	Introduction	3
1.1	Setting the Stage	3
1.2	A Software Engineering Triptych	7
1.2.1	Software Versus Systems Development	7
1.2.2	Motivating the Triptych	7
1.2.3	Domain Engineering	7
1.2.4	Requirements Engineering	9
1.2.5	Software Design	11
1.2.6	Discussion	12
1.3	Documentation	13

1.3.1	Document Kinds	14
1.3.2	Phase, Stage and Step Documents	14
1.3.3	Informative Documents	15
1.3.4	Descriptive Documents	17
1.3.5	Analytic Documents	22
1.4	Formal Techniques and Formal Tools	25
1.4.1	On Formal Techniques and Languages	25
1.4.2	Formal Techniques in SE Textbooks	26
1.4.3	Some Programming Languages	26
1.4.4	Some Formal Specification Languages	27
1.4.5	Insufficiency of Current, Formal Languages	29
1.4.6	Other Formal Tools	30
1.4.7	Why Formal Techniques and Formal Tools?	30
1.5	Method and Methodology	31
1.5.1	Method	32
1.5.2	Methodology	32
1.5.3	Discussion	32
1.5.4	Meta-methodology	32
1.6	The Very Bases of Software	33
1.6.1	Didactics and Paradigms	34
1.6.2	Pragmatics, Semantics and Syntax	34
1.6.3	On Specification and Programming Paradigms	38
1.6.4	Descriptions, Prescriptions and Specifications	38
1.6.5	Metalanguages	39
1.6.6	Summary	39
1.7	Aims and Objectives	40
1.7.1	Aims	40
1.7.2	Objectives	40
1.7.3	Discussion	41
1.8	Bibliographical Notes	41
1.9	Exercises	41

Part II DISCRETE MATHEMATICS

2	Numbers	45
2.1	Introduction	45
2.2	Numerals and Numbers	46
2.3	Subsets of Numbers	46
2.3.1	Natural Numbers: Nat	46
2.3.2	Integers: Int	48
2.3.3	Real Numbers: Real	49
2.3.4	Irrational Numbers	50
2.3.5	Algebraic Numbers	50
2.3.6	Transcendental Numbers	50

2.3.7	Complex and Imaginary Numbers	51
2.4	Type Definitions: Numbers	51
2.5	Summary	52
2.6	Bibliographical Notes	53
2.7	Exercises	53
3	Sets	55
3.1	Background	56
3.2	Mathematical Sets	56
3.3	Special Sets	58
3.3.1	Axiom of Extension	58
3.3.2	Partitions	58
3.3.3	Power Sets	58
3.4	Sorts and Type Definitions: Sets	58
3.4.1	Set Abstractions	58
3.4.2	Set Type Expressions and Type Definitions	59
3.4.3	Sorts	59
3.5	Sets in RSL	59
3.6	Bibliographical Notes	60
3.7	Exercises	60
4	Cartesians	63
4.1	The Issues	63
4.2	Cartesian-Valued Expressions	64
4.3	Cartesian Types	64
4.4	Cartesian Arity	65
4.5	Cartesian Equality	66
4.6	Some Construed Examples	66
4.7	Sorts and Type Definitions: Cartesians	68
4.7.1	Cartesian Abstractions	68
4.7.2	Cartesian Type Expressions and Type Definitions	68
4.8	Cartesians in RSL	69
4.9	Bibliographical Notes	69
4.10	Exercises	69
5	Types	71
5.1	Values and Types	72
5.2	Phenomena and Concept Types	73
5.2.1	Phenomena and Concepts	73
5.2.2	Entities: Atomic and Composite	73
5.2.3	Attributes and Values	74
5.3	Programming Language Type Concepts	77
5.4	Sorts or Abstract Types	80
5.5	Built-in and Concrete Types	81
5.6	Type Checking	82

5.6.1	Typed Variables and Expressions	82
5.6.2	Type Errors	83
5.6.3	Detection of Type Errors	83
5.7	Types as Sets, Types as Lattices	84
5.8	Summary	84
5.9	Exercises	84
6	Functions	87
6.1	General Overview	89
6.1.1	Special Remarks	89
6.2	The Issues	90
6.2.1	Background	90
6.2.2	Some Concepts of Functions	90
6.3	How Do Functions Come About?	94
6.4	An Aside: On the Concept of Evaluation	96
6.4.1	[E]Valuation, Interpretation and Elaboration	96
6.4.2	Two Evaluation Examples	96
6.4.3	Function Invocation/“Function Call”	98
6.5	Function Algebras	98
6.5.1	Functions	98
6.5.2	Function Types	98
6.5.3	Higher-Order Function Types	99
6.5.4	Nondeterministic Functions	99
6.5.5	Constant Functions	100
6.5.6	Strict Functions	101
6.5.7	Strict Functions and Strict Function Invocation	101
6.5.8	Operations on Functions	101
6.6	Currying and λ -Notation	103
6.6.1	Currying	103
6.6.2	λ -Notation	103
6.6.3	Example of Currying and λ -Notation	104
6.7	Relations and Functions	104
6.7.1	Predicates	105
6.7.2	Function Evaluation by Relation Search	105
6.7.3	Nondeterministic Functions	106
6.8	Type Definitions	106
6.9	Conclusion	106
6.10	Bibliographical Notes	107
6.11	Exercises	107
7	A λ-Calculus	109
7.1	Informal Introduction	110
7.2	A “Pure” λ -Calculus Syntax	110
7.3	A λ -Calculus Pragmatics	112
7.4	A “Pure” λ -Calculus Semantics	112

7.4.1	Free and Bound Variables	113
7.4.2	Binding and Scope	113
7.4.3	Collision and Confusion of Variables	113
7.4.4	Substitution	114
7.4.5	α -Conversion and β -Conversion Rules	115
7.4.6	λ -Conversion	115
7.5	Call-by-Name Versus Call-by-Value	116
7.6	The Church–Rosser Theorems — Informal Version	117
7.7	The RSL λ -Notation	117
7.7.1	Extending λ -Expressions	117
7.7.2	The “let ... in ... end” Construct	118
7.8	Fix Points	119
7.8.1	The Issue	119
7.8.2	Informal Outline	119
7.8.3	The Fix Point Operator \mathbf{Y}	120
7.8.4	Fix Point Evaluation	121
7.9	Discussion	122
7.9.1	General	122
7.9.2	On Minimal, Maximal and All Fix Points	122
7.9.3	Emphasis	122
7.9.4	Principles, Techniques and Tools	122
7.10	Bibliographical Notes	123
7.10.1	References	123
7.10.2	Alonzo Church, 1903–1995	123
7.11	Exercises	123
8	Algebras	127
8.1	Introduction	127
8.2	Formal Definition of the Algebra Concept	128
8.3	How Do Algebras Come About?	129
8.4	Kinds of Algebras	130
8.4.1	Concrete Algebras	130
8.4.2	Abstract Algebras	130
8.4.3	Heterogeneous Algebras	131
8.4.4	Universal Algebras	132
8.5	Specification Algebras	133
8.5.1	Syntactic Means of Expressing Algebras	134
8.5.2	An Example Stack Algebra	134
8.5.3	An Example Queue Algebra	135
8.5.4	Towards Semantic Models of “class” Expressions	136
8.6	RSL Syntax for Algebra Specifications	137
8.6.1	“class” Expressions	137
8.6.2	“scheme” Declarations	138
8.7	Discussion	138
8.7.1	General	138

8.7.2	Principles, Techniques and Tools	139
8.8	Bibliographical Notes	139
8.9	Exercises	139
9	Mathematical Logic	141
9.1	The Issues	142
9.1.1	Language of Boolean Ground Terms	142
9.1.2	Language of Propositional Expressions	143
9.1.3	Language of Predicate Expressions	143
9.1.4	Boolean-Valued Expressions	144
9.1.5	“chaos” — Undefined Expression Evaluations	144
9.1.6	Axiom Systems and Inference Rules	145
9.1.7	Proof Systems	146
9.1.8	A Note on Two Axiom Systems	146
9.1.9	The “if ... then ... else ... end” Connective	147
9.1.10	Discussion	147
9.2	Proof Theory Versus Model Theory	148
9.2.1	Syntax	148
9.2.2	Semantics	148
9.2.3	Syntax Versus Semantics	149
9.2.4	Formal Logics: Syntax and Semantics	149
9.2.5	Issues Related to Proofs	153
9.2.6	Relating Proof Theory to Model Theory	153
9.2.7	Discussion	155
9.3	A Language of Boolean Ground Terms	156
9.3.1	Syntax and Semantics	156
9.3.2	The Connectives: \sim , \wedge , \vee , \Rightarrow , $=$, \neq , \equiv	157
9.3.3	Three-Valued Logic	158
9.3.4	Ground Terms and Their Evaluation	161
9.3.5	“Syntactic” Versus “Semantic Semantics”	164
9.3.6	Discussion	165
9.4	Languages of Propositional Logic	165
9.4.1	Propositional Expressions, PRO	166
9.4.2	Examples	167
9.4.3	Proposition Evaluation, Eval_PRO	168
9.4.4	Two-Valued Propositional Calculi	169
9.4.5	Discussion	171
9.5	Languages of Predicate Logic	171
9.5.1	Motivation	172
9.5.2	Informal Presentation	172
9.5.3	Examples	173
9.5.4	Quantifiers and Quantified Expressions	176
9.5.5	Syntax of Predicate Expressions, PRE	178
9.5.6	A Predicate Calculus	180
9.5.7	Predicate Expression Evaluation	181

9.5.8	First-Order and Higher-Order Logics	184
9.5.9	Validity, Satisfiability and Models	184
9.5.10	Discussion	186
9.6	Axiom Systems	186
9.6.1	General	187
9.6.2	Axioms	187
9.6.3	Axiom System	188
9.6.4	Consistency and Completeness	189
9.6.5	Property-Oriented Specifications	189
9.6.6	Discussion	196
9.7	Summary	196
9.8	Bibliographical Notes	197
9.9	Exercises	197

Part III SIMPLE RSL

General	201
RSL Versus VDM-SL, Z and B	201
What, Syntactically, Constitutes a Specification?	202
Towards an RSL “Standard”	203
RSL Tools	203
10 Atomic Types and Values in RSL	205
10.1 Introduction	205
10.1.1 Mathematical Versus Enterprise Modelling	206
10.1.2 The “Primitive” Model Building Blocks	206
10.2 The RSL Numbers	206
10.2.1 Three Types of Numbers	207
10.2.2 Operations on RSL Numbers	207
10.3 Enumerated Tokens	208
10.3.1 Motivation	208
10.3.2 General Theory	208
10.3.3 Operations on Tokens	209
10.3.4 Enumerated Tokens in Abstract Models	210
10.3.5 Modelling Using Enumerated Tokens	211
10.4 Characters and Texts	212
10.4.1 Motivation	212
10.4.2 The Character and Text Data Types	212
10.5 Identifiers and General Tokens	213
10.5.1 Identifiers	213
10.5.2 Operations on General Tokens	214
10.5.3 General Tokens	215
10.6 Discussion	216
10.6.1 General	216
10.6.2 Modelling Atomic Entities	216

10.7	Exercises	217
11	Function Definitions in RSL	221
11.1	The Function Type	221
11.1.1	Syntax of Function Types	221
11.1.2	Informal Semantics of \rightarrow and \rightsquigarrow	222
11.2	Model-Oriented Explicit Definitions	222
11.3	Model-Oriented Axiomatic Definitions	223
11.4	Model-Oriented pre/post-Condition Definitions	224
11.5	Property-Oriented Axiomatic Definitions	226
11.6	Property-Oriented Algebraic Definitions	227
11.7	Summary of RSL Function Definition Styles	228
11.8	Discussion	229
11.9	Exercises	229
12	Property-Oriented and Model-Oriented Abstraction	231
12.1	Abstraction	232
12.1.1	The Issues	232
12.1.2	Abstraction and Specification	233
12.1.3	An Essay on Abstraction	233
12.2	Property-Oriented Abstractions	235
12.2.1	Pragmatics of Property-Oriented Specifications	235
12.2.2	Syntactics of Property-Oriented Specifications	236
12.2.3	Semantics of Property-Oriented Specifications	240
12.2.4	Discussion	240
12.3	Model Versus Property Abstractions	241
12.3.1	Representation and Operation Abstraction	241
12.3.2	Property- Versus Model-Oriented Abstractions	241
12.3.3	Definitions	242
12.3.4	Representation Abstraction Examples	243
12.3.5	Operation Abstraction Examples	246
12.3.6	Discussion	248
12.4	Model-Oriented Abstractions	250
12.4.1	Ultrashort Overview of the Next Six Chapters	250
12.4.2	Models and Models	250
12.4.3	Underspecification	251
12.4.4	Determinism and Nondeterminism	252
12.4.5	Why Loose Specifications?	253
12.4.6	Discussion	253
12.5	Principles, Techniques and Tools	254
12.5.1	Property- Versus Model-Oriented Specification?	254
12.5.2	Property-Oriented Specification Style	255
12.5.3	Model-Oriented Specification Style	256
12.5.4	Implicit and Explicit Functions	257
12.5.5	No Confusion, Please!	257

12.5.6	A Note on Observer Functions	258
12.6	Exercises	260
13	Sets in RSL	263
13.1	Sets: The Issues	264
13.2	The Set Data Type	265
13.2.1	Set Types: Definitions and Expressions	265
13.2.2	Set Value Expressions	266
13.2.3	Set Binding Patterns and Matching	271
13.2.4	Nondeterminism	272
13.3	Examples of Set-Based Abstractions	273
13.3.1	Representation I	273
13.3.2	File Systems I	273
13.3.3	Representation II	275
13.4	Abstracting and Modelling With Sets	276
13.4.1	Modelling Networks	276
13.4.2	Modelling Pseudo-hierarchies	277
13.4.3	Modelling a Telephone System	280
13.5	Inductive Set Definitions	284
13.5.1	Inductive Set Type Definitions	284
13.5.2	Inductive Set Value Definitions	285
13.6	A Comment on Varying Sets	287
13.7	Principles, Techniques and Tools	288
13.8	Discussion	289
13.9	Bibliographical Notes	289
13.10	Exercises	289
14	Cartesians in RSL	295
14.1	Cartesians: The Issues	295
14.2	The Cartesian Data Type	296
14.2.1	Cartesian Types and Type Expressions	296
14.2.2	Cartesian Value Expressions	298
14.2.3	Cartesian Operations, I	299
14.2.4	Cartesian Binding Patterns and Matching	299
14.2.5	Cartesian Operations, II	300
14.3	Examples of Cartesian Abstractions	300
14.3.1	File Systems II	300
14.3.2	Kuratowski: Pairs as Sets	301
14.4	Abstracting and Modelling with Cartesians	303
14.4.1	Modelling Syntactic Structures	303
14.4.2	Cartesian “ <i>let ... in ... end</i> ” Bindings	308
14.4.3	Modelling Semantic Structures	308
14.4.4	Cartesians: A First Discussion	312
14.5	Inductive Cartesian Definitions	312
14.5.1	Inductive Cartesian Type Definitions	312

14.6	14.5.2 Inductive Cartesian Value Definitions	313
	14.6 Discussion	315
	14.6.1 General	315
	14.6.2 Principles, Techniques and Tools	315
14.7	14.7 Exercises	316
15	15 Lists in RSL	321
15.1	15.1 Issues Related to Lists	322
15.2	15.2 The List Data Type	322
	15.2.1 List Types	322
	15.2.2 List Value Expressions	323
	15.2.3 List Binding-Patterns and Matching	327
	15.2.4 Lists: Determinism and Nondeterminism Revisited ..	328
15.3	15.3 Small Examples of List-Based Abstractions	328
	15.3.1 Representations	328
	15.3.2 Stacks and Queues	329
	15.3.3 File Systems III	330
	15.3.4 Sorting Algorithms	332
15.4	15.4 Abstracting and Modelling with Lists	333
	15.4.1 Modelling Books Using Lists	334
	15.4.2 Modelling “KeyWord-In-Context, KWIC”	335
15.5	15.5 Inductive List Definitions	340
	15.5.1 Inductive List Type Definitions	340
	15.5.2 Inductive List Value Definitions	341
15.6	15.6 A Review of List Abstractions and Models	342
15.7	15.7 Lists: A Discussion	343
15.8	15.8 Exercises	343
16	16 Maps in RSL	349
16.1	16.1 The Issues	350
16.2	16.2 The Map Data Type	350
	16.2.1 Map Types: Definitions and Expressions	350
	16.2.2 Map Value Expressions	351
	16.2.3 Map Binding Patterns and Matching	355
	16.2.4 Nondeterminism	356
16.3	16.3 Examples of Map-Based Abstractions	356
	16.3.1 Sorting	356
	16.3.2 Equivalence Relations	357
16.4	16.4 Abstracting and Modelling with Maps	358
	16.4.1 Graphs	358
	16.4.2 Structured Tables	360
	16.4.3 Hierarchies	362
	16.4.4 Relational File Systems (IV) and Databases	366
	16.4.5 Complex Pointer Data Structures	369
	16.4.6 Well-formedness of Data Structures	378

16.4.7	Discussion	382
16.5	Inductive Map Definitions	383
16.5.1	Inductive Map Type Definitions	383
16.5.2	Inductive Map Value Definitions	384
16.6	A Review of Map Abstractions and Models	386
16.7	Maps: A Discussion	388
16.8	Exercises	388
17	Higher-Order Functions in RSL	393
17.1	Functions: The Issues	393
17.2	Examples Using Function-Based Abstractions	394
17.2.1	Functionals	394
17.2.2	Discussion	395
17.3	Abstracting and Modelling With Functions	395
17.3.1	Concepts as Functions	396
17.3.2	Operator Lifting	399
17.4	Inductive Function Definitions	406
17.4.1	Inductive Function Type Definitions	406
17.4.2	Inductive Function Value Definitions	407
17.5	Review of Function Abstractions and Models	407
17.6	Discussion	408
17.7	Exercises	408

Part IV SPECIFICATION TYPES

18	Types in RSL	413
18.1	The Issues	413
18.2	Type Categories	415
18.2.1	Abstract Types: Sorts	415
18.2.2	Concrete Types	415
18.2.3	Discussion	416
18.3	Enumerated Token Types Revisited	416
18.4	Records: Constructors and Destructors	417
18.4.1	General	417
18.4.2	Variant Record Value Induction Axioms	418
18.4.3	An Example	419
18.5	Union Type Definitions	420
18.6	Short Record Type Definitions	421
18.7	Type Expressions, Revisited	421
18.8	Subtypes	422
18.9	Type Definitions, Revisited	422
18.10	On Recursive Type Definitions	423
18.11	Discussion	423
18.11.1	General	423

18.11.2	Principles, Techniques and Tools	423
18.12	Bibliographical Notes	424
18.13	Exercises	424

Part V SPECIFICATION PROGRAMMING

On Specification Programming	427	
On Problems and Exercises	428	
19	Applicative Specification Programming	429
19.1	Scope and Binding	430
19.1.1	Binding Patterns — An Informal Exposition	430
19.1.2	“let” Construct Scope and Binding [1]	432
19.1.3	Function Definition Scope and Binding [2]	432
19.1.4	“case” Construct Scope and Binding [3]	433
19.1.5	Comprehensions: Scope and Binding [4]	434
19.1.6	Quantifications: Scope and Binding [5]	435
19.2	Intuition	435
19.2.1	Simple “let $a = \mathcal{E}_d$ in $\mathcal{E}_b(a)$ end”	435
19.2.2	Recursive “let $f(a) = \mathcal{E}_d(f)$ in $\mathcal{E}_b(f,a)$ end”	436
19.2.3	Predicative “let $a:A \bullet \mathcal{P}(a)$ in $\mathcal{E}(a)$ end”	436
19.2.4	Multiple “let $a_i = \mathcal{E}_{d_i}$ in $\mathcal{E}_b(a_i)$ end”	436
19.2.5	Literals and Identifiers	437
19.3	Operator/Operand Expressions	438
19.4	Enumerated and Comprehended Expressions	438
19.5	Conditional Expressions	439
19.6	Bindings, Typings, Patterns and Matching	440
19.6.1	The Issues	441
19.6.2	An Essence of Bindings and Patterns	441
19.6.3	Binding Patterns	443
19.6.4	Typings	448
19.6.5	Choice Patterns and Bindings	448
19.6.6	Summary	454
19.7	Review and Discussion	455
19.7.1	General	455
19.7.2	Principles and Techniques	455
19.8	Bibliographical Notes	455
19.9	Exercises	456
20	Imperative Specification Programming	467
20.1	Intuition	468
20.2	Imperative Combinators: A λ -Calculus	468
20.2.1	[0] “variable” Declarations	468
20.2.2	[1] Assignments: “var := expression”	470
20.2.3	[9] State Expressions	471

20.2.4	[2] “skip”: No-Action	471
20.2.5	[3] Statement Sequencing (;)	471
20.2.6	[4] “if ... then ... else ... end”	472
20.2.7	[5–6] “while ... do ... end”, and “do ... until ... end” ..	472
20.2.8	[7] “case ... of ... end”	472
20.2.9	[8] “for... in ... do... end”	473
20.3	Variable References: Pointers	473
20.3.1	A Discourse on Simple References	473
20.3.2	Dynamic Allocation and Referencing	474
20.3.3	Discussion: Semantics First, Then Syntax	479
20.3.4	Discussion: Type Homomorphisms	480
20.3.5	The Notion of State	480
20.4	Function Definitions and Expressions	480
20.4.1	The Unit Type Expression, I	481
20.4.2	Imperative Functions	481
20.4.3	Read/Write Access Descriptions	481
20.4.4	Local Variables	482
20.4.5	The Unit Type Expression, II	482
20.4.6	Pure Expressions	483
20.4.7	Read-Only Expressions	483
20.4.8	Equivalence (\equiv) and Equality ($=$)	484
20.5	Translations: Applicative to Imperative	486
20.5.1	Applicative to Imperative Translations	486
20.5.2	Recursive to Iterative Translations	487
20.5.3	Applicative to Imperative Schemas	488
20.5.4	Correctness, Principles, Techniques and Tools	495
20.6	Styles of Configuration Modelling	495
20.6.1	Applicative Contexts and States	495
20.6.2	Applicative Contexts and Imperative States	499
20.6.3	Imperative Contexts and States	502
20.6.4	Summary of Sequential Models	505
20.7	Review and Discussion	505
20.7.1	Review	505
20.7.2	Discussion	506
20.8	Bibliographical Notes	506
20.8.1	Theory of Computation	506
20.8.2	A Type Theory for the λ -Calculus	506
20.8.3	Source Program Transformation Works	507
20.8.4	Laws of Imperative Programming	507
20.9	Exercises	508

21 Concurrent Specification Programming	511
21.1 Behaviour and Process Abstractions	512
21.1.1 Introduction	513
21.1.2 On Process and Other Abstractions	513
21.2 Intuition	514
21.2.1 Illustrative Rendezvous Scenarios	514
21.2.2 Diagram and Notation Summary	518
21.2.3 On a Trace Semantics	518
21.2.4 Some Characterisations: Processes, Etcetera	520
21.2.5 Principle of Process Modelling	521
21.2.6 Informal Examples	522
21.2.7 Some Modelling Comments — An Aside	527
21.2.8 Examples Continued	528
21.2.9 Some System Channel Configurations	529
21.2.10 Concurrency Concepts — A Summary	530
21.3 Communicating Sequential Processes, CSP	532
21.3.1 Preliminaries: Processes and Events	532
21.3.2 Process Combinators, Etcetera	533
21.3.3 Discussion	536
21.4 The RSL/CSP Process Combinators	537
21.4.1 RSL-like Channels	537
21.4.2 RSL Communication Clauses	538
21.4.3 RSL Processes	539
21.4.4 Parallel Process Combinator	541
21.4.5 Nondeterministic External Choice	541
21.4.6 Nondeterministic Internal Choice	542
21.4.7 Interlock Combinator	542
21.4.8 Summary	542
21.4.9 A Note of Caution	543
21.5 Translation Schemas	543
21.5.1 Stage I: An Applicative Schema	543
21.5.2 Stage II: A Simple Reformulation	544
21.5.3 Stage III: Introducing Parallelism	544
21.5.4 Stage IV: A Simple Reformulation	545
21.5.5 Stage Relations	546
21.5.6 Stage V: An Imperative Reformulation	547
21.5.7 Some Remarks	547
21.6 Parallelism and Concurrency: A Discussion	547
21.6.1 CSP and RSL/CSP	547
21.6.2 Modelling Techniques	548
21.7 Bibliographical Notes	548
21.8 Exercises	548

Part VI AND SO ON!

22 Etcetera!	557
22.1 What Have We Covered?	557
22.2 What Is Next?	557
22.3 What Is Next-Next?	558
22.4 A Caveat	559
22.5 Formal Methods “Lite”	559
22.6 Bibliographical Notes	560

Part VII APPENDIXES

A Common Exercise Topics	563
A.1 Transportation Nets	563
A.2 Container Logistics	564
A.3 Financial Service Industry	564
A.4 Summary References to Exercises	566
B Glossary	567
B.1 Categories of Reference Lists	568
B.1.1 Glossary	568
B.1.2 Dictionary	568
B.1.3 Encyclopædia	569
B.1.4 Ontology	569
B.1.5 Taxonomy	569
B.1.6 Terminology	569
B.1.7 Thesaurus	569
B.2 Typography and Spelling	569
B.3 The Glosses	570
C Indexes	649
C.1 Symbols Index	650
C.1.1 Operators	650
C.1.2 Constructors	652
C.1.3 Constant Value Literals	653
C.1.4 Combinators	653
C.1.5 Calculi	654
C.1.6 Abbreviations	654
C.2 Concepts Index	656
C.3 Characterisations and Definitions Index	680
C.4 Authors Index	682
References	687