
Contents

Part I Prologue

1	Components of a Theory	3
1.1	The Role of Theory	4
1.1.1	Theory and Practice	4
1.1.2	What Is Theory?	5
1.1.3	Reality and Abstractions	5
1.1.4	Role of Theory in Engineering	6
1.1.5	Emerging Need for a Theory	7
	Review Questions	7
1.2	Parts of a Comprehensive Theory	8
1.2.1	Spectrum of Abstractions	8
1.2.2	Development of Abstractions	9
1.2.3	Execution Models	10
1.2.4	Language Concepts	10
1.2.5	Underlying Philosophy	11
1.2.6	Programming Logic	14
1.2.7	Design Methodology	15
	Review Questions	16
1.3	The Structure of the Book	17
1.3.1	Part II: Fundamentals	17
1.3.2	Part III: Building a Practical Theory	18
1.3.3	Part IV: Distributed and Real-Time Systems	19
	Bibliographic Notes	20

Part II Fundamentals

2	Towards an Action Language	25
2.1	Modeling of System State	26
2.1.1	An Introductory Problem: Gas Burner	26
2.1.2	State Variables	26
2.1.3	Components and Interfaces	27
2.1.4	Input and Output Variables	28
	Review Questions	29
	Exercises	29
2.2	Executions and Actions	29
2.2.1	Executions	30
2.2.2	Actions	30
2.2.3	Nondeterminism	31
2.2.4	Absence of Probabilities	32
2.2.5	Atomicity of Actions	32
2.2.6	Interleaving	33
2.2.7	Absence of Processes	34
2.2.8	Actions as Syntactic Entities	35
2.2.9	Example: Gas-burner Actions	36
	Review Questions	39
	Exercises	39
2.3	Fairness as an Execution Force	40
2.3.1	Need to Control Nondeterminism	40
2.3.2	Fairness	41
2.3.3	Fairness in the Action Language	41
2.3.4	The Effect of Fairness	42
2.3.5	Example: Gas Burner	43
2.3.6	Fundamental Liveness	43
2.3.7	Example: Fair Scheduling of Processes	44
2.3.8	Theoretical Power of Fairness	44
2.3.9	Probabilistic Implementation	45
2.3.10	Critique and Defence of Fairness	45
	Review Questions	46
	Exercises	46
2.4	Implementation of Action Systems	48
2.4.1	Operational Specifications vs. Programs	48
2.4.2	Example: Gas Burner	49
2.4.3	Validation: Satisfaction of Specification	53
2.4.4	Validation: Non-formalized Requirements	54
	Review Questions	55
	Exercises	55
	Bibliographic Notes	55

3	Formal Properties of Behaviors	57
3.1	States and State Functions	58
3.1.1	State Variables	58
3.1.2	States	58
3.1.3	State Functions	59
3.1.4	State Predicates	59
3.1.5	Predicate Expressions and Laws	61
	Review Questions	61
	Exercises	62
3.2	Properties of Behaviors	62
3.2.1	Behaviors	62
3.2.2	Properties and Characteristic Sets	63
3.2.3	Safety Properties	63
3.2.4	Liveness Properties	64
3.2.5	Mixed Properties	64
3.2.6	Limitations of Potential Infinity	65
3.2.7	The Closure of a Property	65
3.2.8	Formal Characterization of Safety and Liveness	66
	Review Questions	67
	Exercises	68
3.3	Temporal Expressions	69
3.3.1	Semantic Interpretation	69
3.3.2	Extending Predicate Logic	69
3.3.3	Temporal ‘Always’ Operator	70
3.3.4	State Invariants	71
3.3.5	Actions	71
3.3.6	Stuttering	72
3.3.7	Enabling of Actions	73
3.3.8	TLA: Insensitivity to Stuttering	74
3.3.9	Step Invariants	75
3.3.10	Stability Predicates	76
3.3.11	Quantification of State Variables	77
	Review Questions	78
	Exercises	78
3.4	Expressing Liveness Properties	79
3.4.1	Eventualities	79
3.4.2	Arity of Operators	79
3.4.3	Duality of Operators	80
3.4.4	Duality Principle	80
3.4.5	Derived Operators for Temporal Relations	81
3.4.6	Combined Temporal Operators	82
3.4.7	Stuttering Steps and Fairness	82
3.4.8	Stutter-excluding Part of Actions	83
3.4.9	Weak Fairness	83
3.4.10	Strong Fairness	84

	Review Questions	84
	Exercises	84
3.5	TLA-based Specifications	86
3.5.1	Behaviors and Executions	86
3.5.2	Operational Safety Specifications in TLA	86
3.5.3	Feasible Liveness Conditions	87
3.5.4	Canonical TLA Expressions	88
3.5.5	Action Systems and TLA	89
3.5.6	Erroneous Action Systems	91
3.5.7	Action Identity	93
3.5.8	Specification vs. Modeling	94
3.5.9	Non-canonical Requirements	95
3.5.10	Example: Mutual Exclusion	96
3.5.11	Enforcing Causal Relations	97
3.5.12	Possibility Properties	98
3.5.13	Parameterization	99
	Review Questions	99
	Exercises	100
	Bibliographic Notes	101
4	Proving Behavioral Properties	103
4.1	Introduction	104
4.1.1	Logical Proofs in System Design	104
4.1.2	Logical Deductions	104
4.1.3	Non-temporal Basis	105
	Review Questions	106
	Exercises	106
4.2	Deduction of Invariants	107
4.2.1	State Invariants	107
4.2.2	Step Invariants	108
4.2.3	Simple Examples	108
	Exercises	110
4.3	Deduction of Eventualities	111
4.3.1	Temporal Weakening of Eventualities	111
4.3.2	Deduction of Fairness Properties	112
4.3.3	Utilizing Weak Fairness	112
4.3.4	Utilizing Strong Fairness	113
4.3.5	Well-founded Ordering	115
4.3.6	Example	116
	Review Questions	118
	Exercises	119
	Bibliographic Notes	121

Part III Building a Practical Theory

5	Basic Language Facilities	125
5.1	Finite-state Structures	126
5.1.1	Motivation for Finite-state Structures	126
5.1.2	Mutually Exclusive States	126
5.1.3	Example: Gas-burner States	127
5.1.4	Parallel State Structures	128
5.1.5	Nested State Structures	128
5.1.6	Example	129
5.1.7	On the Role of Graphical Illustrations	130
5.1.8	Interpreting Substate Structures in TLA	131
	Review Questions	132
5.2	Typing of State Variables	132
5.2.1	Types as Sets of Values	132
5.2.2	Type Invariants	133
5.2.3	Scopes of Variables	133
5.2.4	Initial and Default Values	134
5.2.5	Discussion	135
	Review Questions	136
	Exercises	136
5.3	Objects and Relations	136
5.3.1	The State of an Object	136
5.3.2	Classes	137
5.3.3	The Size of a Class	137
5.3.4	Relations	138
5.3.5	Special Classes of Relations	139
	Review Questions	142
5.4	Parameterized and Multi-object Actions	142
5.4.1	Action Parameters	142
5.4.2	Action Participants	143
5.4.3	Intuition for Action Execution	143
5.4.4	Multi-object Actions in Specifications	144
5.4.5	Absence of Object Names	145
5.4.6	Quantification and Relations in Actions	145
	Review Questions	146
	Exercises	146
5.5	Formalization of Multi-object Actions	147
5.5.1	Quantified Action Expressions	147
5.5.2	Mapping Between Actions	147
5.5.3	Fairness for Multi-object Actions	148
5.5.4	The Effect of Infinite Classes	149
5.5.5	Expressing Fairness Requirements	149
	Review Questions	150

	Exercises	150
5.6	Dealing with Quantification	151
	5.6.1 Free Variables	151
	5.6.2 Quantifiers	151
	5.6.3 Quantified Operators	152
	5.6.4 Proof Rules for Quantification	153
	5.6.5 Example: Simple Exchange Sort	156
	Review Questions	158
	Exercises	159
	Bibliographic Notes	159
6	Fundamentals of Design Methodology	161
6.1	Refinement by Superposition	162
	6.1.1 Refinement of Actions	162
	6.1.2 Superposition	164
	6.1.3 Preservation of Properties	164
	6.1.4 Supporting Superposition	165
	6.1.5 Example: Refining a Stack	166
	6.1.6 Example on Liveness Preservation	167
	6.1.7 Proof Obligations for Liveness Preservation	169
	6.1.8 Liveness Preservation: Special Case	169
	6.1.9 Liveness Preservation: General Case	171
	Review Questions	171
	Exercises	172
6.2	Composition and Layered Specifications	172
	6.2.1 Layered Specifications	173
	6.2.2 Composition of Independent Layers	175
	6.2.3 Unification of Action Instances	176
	6.2.4 Composition of Layers: General Case	178
	6.2.5 Example: Simple Component Composition	178
	Review Questions	179
	Exercises	179
6.3	Superposition-based Design	179
	6.3.1 Closed-system Modularity	180
	6.3.2 Development Strategies	180
	6.3.3 Note on Variables and Associated Actions	182
	6.3.4 Nondeterminism and Refinement Steps	183
	6.3.5 Notes on Encapsulation	183
	Review Questions	184
6.4	Beyond Superposition	185
	6.4.1 Simplification of Actions	185
	6.4.2 Data Refinement	186
	6.4.3 Refinement of Atomicity	186
	6.4.4 Disjunctive Actions	187
	6.4.5 Concatenated Actions	187

	Review Questions	189
	Exercises	189
6.5	Example: Pocket Calculator	190
	6.5.1 Overall Plan	190
	6.5.2 Operand Stack	191
	6.5.3 Entering Numbers	191
	6.5.4 Operations	192
	6.5.5 Overflow	193
	6.5.6 Composition	194
	6.5.7 Discussion	194
	Exercises	195
6.6	Example: Resource Allocation	196
	6.6.1 Overall Plan	196
	6.6.2 Basis	197
	6.6.3 Simple User Processes	198
	6.6.4 Distributed Users	200
	6.6.5 Allocator	201
	6.6.6 Messages	203
	6.6.7 Composed System	204
	6.6.8 Simplification for Distributed Implementation	208
	6.6.9 Discussion	211
	Exercises	212
	Bibliographic Notes	212
7	Object Orientation Elaborated	215
7.1	General Principles	216
	7.1.1 Specification vs. Implementation Concerns	216
	7.1.2 Intuitive Meaning of Subclasses	216
	7.1.3 Formal Requirements for Subclasses	217
	7.1.4 Classes vs. Layers as Basic Units	217
	7.1.5 Specifications as Patterns	218
	7.1.6 Incremental Definition of Classes	219
	7.1.7 Effects of the Execution Model	219
	Review Questions	220
7.2	Class System	221
	7.2.1 Subclasses	221
	7.2.2 Local Variables of Subclasses	223
	7.2.3 Explicit Subclasses	223
	7.2.4 Actions for a Subclass	224
	7.2.5 Composition in the Presence of Subclasses	226
	7.2.6 Subclasses and Preservation of Properties	227
	Review Questions	229
	Exercises	229
7.3	Example: Doctors' Office	229
	7.3.1 First Layer: Illness	230

7.3.2	An Independent Initial Layer: Work	230
7.3.3	Combining the Two Views.....	231
7.3.4	Next Layer: Doctors	232
7.3.5	Final Layer: Receptionists	234
	Exercises	237
7.4	Composite Objects and Subobject Classes	237
7.4.1	Aggregate Classes	237
7.4.2	Identity of Aggregates and Subobjects	238
7.4.3	Subobject Classes	239
7.4.4	Subobject Relations	239
7.4.5	Example: Putting Buffers Together	240
7.4.6	Data Refinement by Aggregation	241
	Review Questions	243
	Exercises	243
7.5	Aggregation vs. Inheritance vs. Copying	243
7.5.1	Similarity of Aggregation and Inheritance	243
7.5.2	Non-equivalence of Aggregation and Inheritance	244
7.5.3	Copying and the ‘Uses’ Relation.....	246
7.5.4	Simple Use of Multiple Inheritance	247
7.5.5	Multiple Inheritance with a Common Superclass	247
7.5.6	Combining Explicit Subclasses and Aggregation	249
7.5.7	Recursive Aggregates	250
7.5.8	Example: Nested Stacks	252
7.5.9	Synchronous Aggregates.....	256
	Review Questions	258
	Exercises	259
	Bibliographic Notes	259
8	Components and Interfaces	263
8.1	Components in a Closed System.....	264
8.1.1	Partitioning of State	264
8.1.2	Partitioned Action Systems	265
8.1.3	Interface Variables.....	266
8.1.4	Interface Actions	267
8.1.5	Communication Through Action Parameters	267
8.1.6	Components and Refinement	268
8.1.7	Modularity by Components	269
	Review Questions	271
8.2	Non-interfering Component Refinements.....	271
8.2.1	Non-interference Conditions	271
8.2.2	Design by Non-interfering Refinements	272
8.2.3	Dealing with Environment Errors.....	274
8.2.4	Example: Database Transactions	275
	Review Questions	278
	Exercises	279

8.3	Robust Component Refinements	279
8.3.1	Need for Relaxing Non-interference	279
8.3.2	Independence of Interput Parameters	281
8.3.3	General Assumptions	281
8.3.4	Robustness Conditions	282
	Review Questions	285
	Exercises	285
8.4	Interface Refinement	286
8.4.1	Goals for Interface Refinement	286
8.4.2	Changes in Responsibilities	286
8.4.3	Example: Simplifying an Interface Action	288
8.4.4	Example: Refinement of Communication	289
8.4.5	Loosening of Component Synchronization	290
	Review Questions	291
	Exercises	291
8.5	Example: Reliable Channel	291
8.5.1	Simple Buffer as an Abstract Channel	292
8.5.2	Adding the Two Ends	293
8.5.3	Alternating Bits	295
8.5.4	Asynchronous Communication	296
	Exercises	299
	Bibliographic Notes	300

Part IV Distributed and Real-Time Systems

9	Distributed Systems	305
9.1	Interleaved Modeling of Concurrency	306
9.1.1	Theory and Reality	306
9.1.2	Interleaved Computations	307
9.1.3	Distributed Execution Model	308
9.1.4	Fairness Paradox	311
9.1.5	Observations	313
9.1.6	Correctness of the Interleaving Model	315
	Review Questions	315
	Exercises	316
9.2	Modeling of Practical Mechanisms	316
9.2.1	Refinements vs. High-level Mechanisms	317
9.2.2	Decentralized Evaluation of Guards	317
9.2.3	Modeling of Implementations	319
9.2.4	Constraints for Implementable Actions	323
9.2.5	Example: Distributed Exchange Sort	324
9.2.6	Getting Rid of Global Guards	325
9.2.7	Dealing with Non-separable Guards	326
9.2.8	Imposing a Policy	327

9.2.9	Asymmetry in Invoking Actions	328
9.2.10	Decoupling of Effects	329
9.2.11	Discussion	331
	Review Questions	331
	Exercises	332
9.3	Uniform Fairness Assumptions	333
9.3.1	Enforcement of Fairness Properties	333
9.3.2	Basic Actions and Action Fairness	334
9.3.3	Example: Dining Philosophers	334
9.3.4	Insufficiency of Fundamental Liveness	336
9.3.5	Weak Interaction Fairness	337
9.3.6	Outline for Enforcing WIF	338
9.3.7	Philosophers with Counters on the Forks	339
9.3.8	Monitoring of Actions Reconsidered	340
9.3.9	WIF and Beyond	340
	Review Questions	341
	Exercises	342
9.4	Generic Example of Coordination	343
9.4.1	Cyclic Stages	343
9.4.2	Critical Moments and Actions	344
9.4.3	Global Guards for Critical Actions	345
9.4.4	Example: Repeated Exchange Sort	346
	Review Questions	348
	Exercises	348
	Bibliographic Notes	348
10	Real Time	353
10.1	Modeling of Real-Time Properties	353
10.1.1	Introducing a Clock	354
10.1.2	Timing of Actions	354
10.1.3	Real-Time Properties	355
10.1.4	Enforcing Real-Time Properties	356
10.1.5	Note on Implementing Deadlines	358
10.1.6	Finite Variability	358
10.1.7	Zeno Behaviors	359
10.1.8	Relative Safety Properties	360
10.1.9	Example: Gas-burner Revisited	361
	Review Questions	362
	Exercises	363
10.2	Periodic and Aperiodic Events	363
10.2.1	Periodic Events	363
10.2.2	Aperiodic Events	364
10.2.3	Example: Toy Car	365
	Exercises	371
10.3	Hybrid Systems	372

10.3.1 Real-Time Functions 372
 10.3.2 Approximation of Physical Quantities 375
 10.3.3 Real-Time Invariants 375
 10.3.4 Monotonic Time-dependent State Functions 376
 10.3.5 Regular Real-Time Predicates 377
 10.3.6 Example: Gas Burner as a Hybrid System 377
 Review Questions 381
 Exercises 382
 Bibliographic Notes 382

Part V Epilogue

11 Reexamining the Theory 387
 11.1 Basic Principles 388
 11.1.1 Dynamic Behaviors 388
 11.1.2 Closed-system Modeling 388
 11.1.3 Action Orientation 389
 11.1.4 Abstractions 389
 11.1.5 Preexistence of Variables 390
 11.1.6 Connection Between Variables and Actions 391
 11.1.7 Composition of Closed Systems 391
 11.2 Two Dimensions of System Architecture 392
 11.2.1 Vertical Architectures 392
 11.2.2 Horizontal Architectures 393
 11.2.3 Orthogonality of the Two Dimensions 393
 11.2.4 Notes on Current Practice 394
 11.2.5 Comparing the Two Kinds of Modularity 395
 11.2.6 Combining the Two Dimensions 396
References 397
Index 409