

Contents

1	Introduction	1
1.1	Goals of the Book and Contours of its Method	3
1.1.1	Stepwise Refinable Abstract Operational Modeling	3
1.1.2	Abstract Virtual Machine Notation	5
1.1.3	Practical Benefits	6
1.1.4	Harness Pseudo-Code by Abstraction and Refinement	8
1.1.5	Adding Abstraction and Rigor to UML Models	9
1.2	Synopsis of the Book	10
2	ASM Design and Analysis Method	13
2.1	Principles of Hierarchical System Design	13
2.1.1	Ground Model Construction (Requirements Capture)	16
2.1.2	Stepwise Refinement (Incremental Design)	20
2.1.3	Integration into Software Practice	26
2.2	Working Definition	27
2.2.1	Basic ASMs	28
2.2.2	Definition	28
2.2.3	Classification of Locations and Updates	33
2.2.4	ASM Modules	36
2.2.5	Illustration by Small Examples	37
2.2.6	Control State ASMs	44
2.2.7	Exercises	53
2.3	Explanation by Example: Correct Lift Control	54
2.3.1	Exercises	62
2.4	Detailed Definition (Math. Foundation)	63
2.4.1	Abstract States and Update Sets	63
2.4.2	Mathematical Logic	67
2.4.3	Transition Rules and Runs of ASMs	71
2.4.4	The Reserve of ASMs	76
2.4.5	Exercises	82
2.5	Notational Conventions	85

3	Basic ASMs	87
3.1	Requirements Capture by Ground Models	87
3.1.1	Fundamental Questions to be Asked	88
3.1.2	Illustration by Small Use Case Models	92
3.1.3	Exercises	109
3.2	Incremental Design by Refinements	110
3.2.1	Refinement Scheme and its Specializations	111
3.2.2	Two Refinement Verification Case Studies	117
3.2.3	Decomposing Refinement Verifications	133
3.2.4	Exercises	134
3.3	Microprocessor Design Case Study	137
3.3.1	Ground Model DLX^{seq}	138
3.3.2	Parallel Model DLX^{par} Resolving Structural Hazards .	140
3.3.3	Verifying Resolution of Structural Hazards (DLX^{par}) .	143
3.3.4	Resolving Data Hazards (Refinement DLX^{data})	148
3.3.5	Exercises	156
4	Structured ASMs (Composition Techniques)	159
4.1	Turbo ASMs (seq, iterate, submachines, recursion)	160
4.1.1	Seq and Iterate (Structured Programming)	160
4.1.2	Submachines and Recursion (Encapsulation and Hiding)	167
4.1.3	Analysis of Turbo ASM Steps	174
4.1.4	Exercises	178
4.2	Abstract State Processes (Interleaving)	180
5	Synchronous Multi-Agent ASMs	187
5.1	Robot Controller Case Study	188
5.1.1	Production Cell Ground Model	188
5.1.2	Refinement of the Production Cell Component ASMs .	193
5.1.3	Exercises	196
5.2	Real-Time Controller (Railroad Crossing Case Study)	198
5.2.1	Real-Time Process Control Systems	198
5.2.2	Railroad Crossing Case Study	201
5.2.3	Exercises	205
6	Asynchronous Multi-Agent ASMs	207
6.1	Async ASMs: Definition and Network Examples	208
6.1.1	Mutual Exclusion	210
6.1.2	Master-Slave Agreement	212
6.1.3	Network Consensus	214
6.1.4	Load Balance	215
6.1.5	Leader Election and Shortest Path	216
6.1.6	Broadcast Acknowledgment (Echo)	218
6.1.7	Phase Synchronization	220
6.1.8	Routing Layer Protocol for Mobile Ad Hoc Networks .	223

6.1.9	Exercises	228
6.2	Embedded System Case Study	229
6.2.1	Light Control Ground Model	229
6.2.2	Signature (Agents and Their State)	231
6.2.3	User Interaction (Manual Control)	231
6.2.4	Automatic Control	236
6.2.5	Failure and Service	237
6.2.6	Component Structure	239
6.2.7	Exercises	240
6.3	Time-Constrained Async ASMs	240
6.3.1	Kermit Case Study (Alternating Bit/Sliding Window)	241
6.3.2	Processor-Group-Membership Protocol Case Study ...	252
6.3.3	Exercises	259
6.4	Async ASMs with Durative Actions	260
6.4.1	Protocol Verification using Atomic Actions	261
6.4.2	Refining Atomic to Durative Actions	268
6.4.3	Exercises	271
6.5	Event-Driven ASMs	271
6.5.1	UML Diagrams for Dynamics	274
6.5.2	Exercises	282
7	Universal Design and Computation Model	283
7.1	Integrating Computation and Specification Models	283
7.1.1	Classical Computation Models	285
7.1.2	System Design Models	293
7.1.3	Exercises	300
7.2	Sequential ASM Thesis (A Proof from Postulates)	301
7.2.1	Gurevich's Postulates for Sequential Algorithms	302
7.2.2	Bounded-Choice Non-Determinism	307
7.2.3	Critical Terms for ASMs	307
7.2.4	Exercises	311
8	Tool Support for ASMs	313
8.1	Verification of ASMs	313
8.1.1	Logic for ASMs	314
8.1.2	Formalizing the Consistency of ASMs	315
8.1.3	Basic Axioms and Proof Rules of the Logic	317
8.1.4	Why Deterministic Transition Rules?	326
8.1.5	Completeness for Hierarchical ASMs	328
8.1.6	The Henkin Model Construction	330
8.1.7	An Extension with Explicit Step Information	334
8.1.8	Exercises	336
8.2	Model Checking of ASMs	338
8.3	Execution of ASMs	340

9	History and Survey of ASM Research	343
9.1	The Idea of Sharpening Turing's Thesis	344
9.2	Recognizing the Practical Relevance of ASMs	345
9.3	Testing the Practicability of ASMs	349
9.3.1	Architecture Design and Virtual Machines	349
9.3.2	Protocols	351
9.3.3	Why use ASMs for Hw/Sw Engineering?	352
9.4	Making ASMs Fit for their Industrial Deployment	354
9.4.1	Practical Case Studies	354
9.4.2	Industrial Pilot Projects and Further Applications	356
9.4.3	Tool Integration	362
9.5	Conclusion and Outlook	365
 References		369
 List of Problems		429
 List of Figures		431
 List of Tables		433
 Index		435

Abstract State Machines

A Method for High-Level System Design and Analysis

Börger, E.; Stärk, R.

2003, X, 438 p. With online files/update., Hardcover

ISBN: 978-3-540-00702-9