

Contents

1	Introduction to Modelling and Validation	1
1.1	Modelling and System Development	1
1.2	Coloured Petri Nets	3
1.3	Abstraction and Visualisation	5
1.4	Formal Modelling and Verification	6
1.5	CPN Tools	8
1.6	Industrial Applications	11
2	Non-hierarchical Coloured Petri Nets	13
2.1	A Simple Example Protocol	13
2.2	Net Structure and Inscriptions	14
2.3	Enabling and Occurrence of Transitions	17
2.4	Second Model of the Protocol	24
2.5	Concurrency and Conflict	29
2.6	Guards	34
2.7	Interactive and Automatic Simulation	35
3	CPN ML Programming	43
3.1	Functional Programming	43
3.2	Colour Sets	45
3.3	Expressions and Types	56
3.4	Functions	60
3.5	Recursion and Lists	65
3.6	Patterns	70
3.7	Computation of Enabled Binding Elements	73
4	Formal Definition of Non-hierarchical Coloured Petri Nets	79
4.1	Multisets	80
4.2	Net Structure and Inscriptions	83
4.3	Enabling and Occurrence of Steps	87

5	Hierarchical Coloured Petri Nets	95
5.1	Modules and Interfaces	95
5.2	Module Instances and Hierarchy	100
5.3	Instance Folding and Module Parameterisation	105
5.4	Model Parameterisation	112
5.5	Fusion Sets	116
5.6	Unfolding Hierarchical CPN Models	124
6	Formal Definition of Hierarchical Coloured Petri Nets	127
6.1	Modules	127
6.2	Module Composition	131
6.3	Instances and Compound Places	136
6.4	Enabling and Occurrence of Steps	141
7	State Spaces and Behavioural Properties	151
7.1	Protocol for State Space Analysis	152
7.2	State Spaces	153
7.3	Strongly-Connected-Component Graphs	160
7.4	Behavioural Properties	163
7.5	Error Diagnostics and Counterexamples	180
7.6	Limitations of State Spaces	185
8	Advanced State Space Methods	189
8.1	State Space Reduction Methods	189
8.2	Sweep-Line Method	191
8.3	Symmetry Method	194
8.4	Equivalence Method	198
9	Formal Definition of State Spaces and Behavioural Properties	203
9.1	Directed Graphs	203
9.2	State Spaces	209
9.3	Reachability Properties	211
9.4	Basic Boundedness Properties	213
9.5	Generalised Boundedness Properties	216
9.6	Home Properties	219
9.7	Liveness Properties	222
9.8	Fairness Properties	227
10	Timed Coloured Petri Nets	231
10.1	First Timed Model of the Protocol	232
10.2	Second Timed Model of the Protocol	243
10.3	State Space Analysis of Timed Models	247
10.4	Time Equivalence Method	252

11	Formal Definition of Timed Coloured Petri Nets	257
11.1	Timed multisets	257
11.2	Net Structure and Inscriptions	264
11.3	Enabling and Occurrence of Steps	265
12	Simulation-based Performance Analysis	273
12.1	Timed Protocol for Performance Analysis	274
12.2	Data Collection from the Occurring Binding Elements	278
12.3	Data Collection from the Markings Reached	281
12.4	Collecting Data from the Final Marking	286
12.5	Simulation Output	287
12.6	Conducting Simulation Experiments	291
12.7	Model Parameters and Configurations	295
13	Behavioural Visualisation	303
13.1	Message Sequence Charts	304
13.2	System-Specific Interaction Graphics	308
14	Examples of Industrial Applications	313
14.1	Protocol Design at Ericsson Telebit	314
14.2	Requirements Engineering at Systematic	329
14.3	Embedded-System Design at Bang and Olufsen	338
14.4	Scheduling Tool for Australian Defence Forces	350
15	Teaching Coloured Petri Nets	363
15.1	Course Context and Aims	363
15.2	Intended Learning Outcomes	364
15.3	Teaching and Assessment Methods	367
15.4	Example of a Student Project from the Course	370
15.5	Experiences from Teaching the CPN Course	372
	References	375
	Index	381

The development of concurrent systems is particularly challenging. A major problem is that these systems possess concurrency and non-determinism which means that the execution of such systems may proceed in many different ways, for example, depending on whether messages are lost during transmission, the scheduling of processes, and the time at which input is received from the environment. Hence, such systems have an astronomical number of possible executions. It is extremely easy for a human designer to miss some crucial interaction patterns when designing such a system, leading to gaps or contradictions in the system design. As a result, concurrent systems are, by nature, complex and difficult to design, test, and debug. Furthermore, for many concurrent systems such as those integrated into nuclear power plants, aircraft control systems, and hospital life support equipment, it is essential that the system works correctly from the very beginning. To cope with the