



# Petri Nets

*Fundamental Models, Verification and Applications*

Edited by  
Michel Diaz

ISTE

 WILEY

This page intentionally left blank

## Petri Nets

This page intentionally left blank

# Petri Nets

*Fundamental Models, Verification and Applications*

Edited by  
Michel Diaz

ISTE

 WILEY

First published in France in 2001 and 2003 by Hermes Science/Lavoisier entitled *Les réseaux de Petri* and *Vérification et mise en œuvre des réseaux de Petri* © Hermes Science Ltd, 2001 © LAVOISIER 2003  
First published in Great Britain and the United States in 2009 by ISTE Ltd and John Wiley & Sons, Inc.

Apart from any fair dealing for the purposes of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act 1988, this publication may only be reproduced, stored or transmitted, in any form or by any means, with the prior permission in writing of the publishers, or in the case of reprographic reproduction in accordance with the terms and licenses issued by the CLA. Enquiries concerning reproduction outside these terms should be sent to the publishers at the undermentioned address:

ISTE Ltd  
27-37 St George's Road  
London SW19 4EU  
UK

John Wiley & Sons, Inc.  
111 River Street  
Hoboken, NJ 07030  
USA

[www.iste.co.uk](http://www.iste.co.uk)

[www.wiley.com](http://www.wiley.com)

© ISTE Ltd, 2009

The rights of Michel Diaz to be identified as the author of this work have been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

---

Library of Congress Cataloging-in-Publication Data

Réseaux de Petri and vérification et mise en œuvre des réseaux de Petri. English  
Petri nets : fundamental models, verification, and applications / edited by Michel Diaz.  
p. cm.

Includes bibliographical references and index.

ISBN 978-1-84821-079-0

1. Electronic data processing--Distributed processing.
  2. Parallel processing (Electronic computers)
  3. System design.
  4. Petri nets. I. Diaz, Michel, 1945- II. Title.
- QA76.9.D5P4813 2009  
511.3'5--dc22

2009017412

---

British Library Cataloguing-in-Publication Data

A CIP record for this book is available from the British Library

ISBN: 978-1-84821-079-0

---

Printed and bound in Great Britain by CPI Antony Rowe, Chippenham and Eastbourne.



**Mixed Sources**  
Product group from well-managed  
forests and other controlled sources

Cert no. SGS-COC-2953  
[www.fsc.org](http://www.fsc.org)  
© 1996 Forest Stewardship Council

# Table of Contents

<b>Preface</b> . . . . .	xv
<b>Introduction</b> . . . . .	xvii
<b>PART 1. FUNDAMENTAL MODELS</b> . . . . .	1
<b>Chapter 1. Basic Semantics</b> . . . . .	3
Michel DIAZ	
1.1. Automata or state machines . . . . .	3
1.1.1. Automata and state machine models. . . . .	3
1.1.2. Tasks and processes . . . . .	5
1.1.3. Some models . . . . .	6
1.2. State machines and Petri nets (PN) . . . . .	8
1.2.1. Composing state machines . . . . .	8
1.2.2. Composition and synchronization . . . . .	10
1.3. Concepts and definitions . . . . .	11
1.3.1. Local states and enabling . . . . .	12
1.3.2. Definition of the semantics of parallelism . . . . .	12
1.3.3. Firing transitions . . . . .	13
1.4. Accessibility graph or marking graph . . . . .	15
1.5. Some basic models . . . . .	18
1.5.1. Co-begin (parallel start) and co-end (synchronized termination) . . . . .	18
1.5.2. Synchronization by a signal. . . . .	18
1.5.3. Mutual exclusion . . . . .	19
1.5.4. The reader and writer mechanisms. . . . .	21
1.5.5. Bounded buffers . . . . .	23
1.6. Conclusion . . . . .	24
1.7. Bibliography . . . . .	25

<b>Chapter 2. Application of Petri Nets to Communication Protocols . . . . .</b>	<b>27</b>
Michel DIAZ	
2.1. Basic models . . . . .	27
2.2. A simple establishment of a connection . . . . .	29
2.2.1. Different global semantics . . . . .	29
2.2.2. Conclusion . . . . .	30
2.3. The alternating bit protocol (ABP): model and verification . . . . .	32
2.3.1. Loss of messages . . . . .	33
2.3.2. Modeling losses . . . . .	34
2.4. Communicating state machines and PN's . . . . .	36
2.5. Conclusion . . . . .	37
2.6. Bibliography . . . . .	38
<b>Chapter 3. Analysis Methods for Petri . . . . .</b>	<b>41</b>
Serge HADDAD and François VERNADAT	
3.1. Introduction . . . . .	41
3.2. Behavioral analysis of Petri nets . . . . .	44
3.2.1. Semantics of a net . . . . .	44
3.2.2. Usual properties . . . . .	45
3.3. Analysis of nets by linear invariants . . . . .	52
3.3.1. Definitions and first applications . . . . .	52
3.3.2. Flow computations . . . . .	58
3.3.3. Semiflow computation . . . . .	60
3.3.4. Application of invariants to the analysis of a net . . . . .	62
3.4. Net reductions . . . . .	65
3.4.1. Pre-agglomeration of transitions . . . . .	66
3.4.2. Post-agglomeration of transitions . . . . .	68
3.4.3. Deletion of redundant places . . . . .	69
3.5. The graph of a Petri net . . . . .	70
3.5.1. General results . . . . .	70
3.5.2. State machines . . . . .	74
3.5.3. Event graph . . . . .	75
3.5.4. Free choice net . . . . .	77
3.6. Bibliography . . . . .	85
<b>Chapter 4. Decidability and Complexity of Petri Net Problems . . . . .</b>	<b>87</b>
Serge HADDAD	
4.1. Introduction . . . . .	87
4.2. Decidability and complexity notions . . . . .	89
4.3. Theoretical results about the reachability graph . . . . .	92
4.4. Analysis of unbounded Petri nets . . . . .	95



4.4.1. Construction of the covering graph . . . . .	95
4.4.2. Shortest sequences . . . . .	101
4.4.3. Backward analysis . . . . .	103
4.5. The reachability problem . . . . .	105
4.5.1. A necessary condition for reachability . . . . .	106
4.5.2. A sufficient condition for reachability . . . . .	106
4.6. Extensions of Petri nets . . . . .	109
4.6.1. Netswith inhibitor arcs . . . . .	109
4.6.2. Self-modifying nets . . . . .	111
4.6.3. Recursive nets . . . . .	113
4.7. Languages of Petri nets . . . . .	116
4.8. Bibliography . . . . .	120
<b>Chapter 5. Time Petri Nets . . . . .</b>	<b>123</b>
Bernard BERTHOMIEU, Marc BOYER and Michel DIAZ	
5.1. Introduction. . . . .	123
5.2. Time Petri nets . . . . .	126
5.2.1. Time nets. . . . .	126
5.2.2. States and firing rule . . . . .	127
5.2.3. Set of states, schedules. . . . .	128
5.2.4. Firing domains . . . . .	129
5.3. Behavior characterization – state classes’ method . . . . .	130
5.3.1. State classes . . . . .	130
5.3.2. Transitions between state classes. . . . .	131
5.3.3. State class equality . . . . .	134
5.3.4. Class graph. . . . .	136
5.3.5. Marking graph and class graph . . . . .	137
5.4. Analysis – operating the state class graph . . . . .	138
5.4.1. Analyzing behavior of time-dependent systems . . . . .	138
5.4.2. Marking reachability . . . . .	139
5.4.3. Boundedness. . . . .	139
5.4.4. Specific properties for set of markings or firing sequences . . . . .	143
5.4.5. Time-dependent analyses, existence of schedules. . . . .	143
5.5. Application example . . . . .	144
5.6. Extensions and variations . . . . .	149
5.6.1. Interpreting multi-enabled transitions . . . . .	149
5.6.2. Other time extensions . . . . .	154
5.7. Implementation using the Tina tool. . . . .	156
5.7.1. Tina tool . . . . .	156
5.7.2. Application example . . . . .	156
5.8. Conclusion . . . . .	158
5.9. Bibliography . . . . .	159

<b>Chapter 6. Temporal Composition and Time Stream Petri Nets</b> . . . . .	163
Michel DIAZ and Patrick SÉNAC	
6.1. Time, synchronization and autonomous behaviors. . . . .	163
6.2. Limitation of time PNs . . . . .	164
6.3. Temporal composition . . . . .	164
6.4. Temporal composition and temporal synchronization. . . . .	165
6.4.1. The semantics of “waiting” . . . . .	165
6.4.2. Pragmatics and time assumptions . . . . .	167
6.5. Time stream PNs. . . . .	168
6.5.1. Definition of the model . . . . .	168
6.5.2. The different firing semantics . . . . .	169
6.5.3. Relating times behavior . . . . .	176
6.5.4. TSPN with structured streams . . . . .	177
6.6. Application to multimedia systems . . . . .	177
6.6.1. Jitter in streams . . . . .	177
6.6.2. Intra- and inter-stream drifts . . . . .	177
6.6.3. Modeling stream composition . . . . .	179
6.6.4. Principle of modeling multimedia systems . . . . .	180
6.6.5. Modeling multimedia scenarios . . . . .	180
6.6.6. TSPN for designing hypermedia architectures. . . . .	181
6.7. Conclusion . . . . .	182
6.8. Bibliography . . . . .	182
<b>Chapter 7. High Level Petri Nets</b> . . . . .	185
Claude GIRAULT and Jean-François PRADAT-PEYRE	
7.1. Introduction . . . . .	185
7.2. Informal introduction to high level nets . . . . .	186
7.2.1. A client-server model . . . . .	186
7.2.2. Client distinction . . . . .	188
7.2.3. Server distinction . . . . .	190
7.2.4. Equivalent unfolded net . . . . .	192
7.2.5. Colored model for the alternate bit protocol . . . . .	194
7.3. Colored net definition . . . . .	196
7.3.1. Notation . . . . .	196
7.3.2. The formalism of colored nets . . . . .	197
7.3.3. Unfolding of a colored net . . . . .	199
7.4. Well-formed net definition . . . . .	200
7.4.1. Color domains . . . . .	201
7.4.2. Color functions . . . . .	203
7.4.3. Guards . . . . .	205
7.4.4. The formalism of well-formed nets . . . . .	207
7.4.5. Regular nets and ordered nets . . . . .	210

7.5. Other high level formalisms . . . . .	212
7.5.1. Interpreted nets . . . . .	212
7.5.2. Algebraic nets . . . . .	214
7.6. Conclusion . . . . .	219
7.7. Bibliography . . . . .	219
<b>Chapter 8. Analysis of High Level Petri Nets . . . . .</b>	<b>221</b>
Claude GIRAULT and Jean-François PRADAT-PEYRE	
8.1. Introduction . . . . .	221
8.2. The symbolic reachability graph . . . . .	222
8.2.1. Symbolic markings . . . . .	223
8.2.2. Symbolic marking representation . . . . .	225
8.2.3. Symbolic firing rule . . . . .	230
8.2.4. Example of a symbolic reachability graph . . . . .	234
8.2.5. Properties of the SRG . . . . .	238
8.3. Colored invariants . . . . .	238
8.3.1. Definition of invariants of high level Petri nets . . . . .	239
8.3.2. Computing flows of a high level net: principles and difficulties . . . . .	242
8.3.3. Computing a non-parametrized generative flow family . . . . .	243
8.3.4. Parametrized generative family of flows for regular nets . . . . .	250
8.3.5. Computation of positive flows . . . . .	252
8.4. Structural reductions . . . . .	253
8.4.1. Principles of extension to high level nets . . . . .	254
8.4.2. Pre-agglomeration and post-agglomeration of transitions . . . . .	255
8.4.3. Deletion of an implicit place . . . . .	261
8.4.4. Application examples . . . . .	261
8.5. Conclusion . . . . .	265
8.6. Bibliography . . . . .	266
<b>Chapter 9. Stochastic Petri Nets . . . . .</b>	<b>269</b>
Serge HADDAD and Patrice MOREAUX	
9.1. Introduction . . . . .	269
9.2. A stochastic semantics for discrete event systems . . . . .	271
9.2.1. The stochastic model . . . . .	271
9.2.2. Analysis with renewing theory . . . . .	273
9.2.3. Discrete time Markov chains . . . . .	274
9.2.4. Continuous time Markov chains . . . . .	276
9.2.5. Semi-Markovian processes . . . . .	278
9.2.6. Regenerative Markovian processes . . . . .	279
9.3. Stochastic Petri nets . . . . .	280
9.3.1. Stochastic Petri nets with general distributions . . . . .	280
9.3.2. Stochastic Petri nets with exponential distributions . . . . .	282

9.3.3. Generalized stochastic Petri nets . . . . .	283
9.3.4. Deterministic stochastic Petri nets . . . . .	284
9.3.5. Phase-type stochastic Petri nets . . . . .	286
9.4. Some standard analysis methods . . . . .	287
9.4.1. Research of a product form . . . . .	287
9.4.2. Bound computations . . . . .	290
9.4.3. Approximation methods . . . . .	293
9.4.4. Unbounded Petri nets . . . . .	297
9.5. Conclusion . . . . .	298
9.6. Bibliography . . . . .	299
<b>Chapter 10. Stochastic Well-formed Petri Nets . . . . .</b>	<b>303</b>
Serge HADDAD and Patrice MOREAUX	
10.1. Introduction . . . . .	303
10.2. Markovian aggregation . . . . .	305
10.3. Presentation of stochastic well-formed Petri nets . . . . .	307
10.3.1. The stochastic process of a well-formed Petri net . . . . .	308
10.3.2. Definition of stochastic well-formed Petri nets. . . . .	309
10.3.3. Modeling a multiprocessor system . . . . .	310
10.4. From the symbolic graph to Markovian aggregation. . . . .	313
10.4.1. Verification of the aggregation condition . . . . .	314
10.4.2. Computation of the parameters of the aggregated chain . . . . .	316
10.4.3. Performance indices of the multiprocessor system . . . . .	317
10.5. Conclusion . . . . .	318
10.6. Bibliography . . . . .	319
<b>Chapter 11. Tensor Methods and Stochastic Petri Nets . . . . .</b>	<b>321</b>
Serge HADDAD and Patrice MOREAUX	
11.1. Introduction . . . . .	321
11.2. Synchronized Markov chains . . . . .	322
11.2.1. Tensor products . . . . .	324
11.2.2. Tensor sum and continuous time Markov chains . . . . .	326
11.3. Tensor algebra and SPN . . . . .	329
11.3.1. Synchronous decomposition of generalized stochastic Petri nets . . . . .	329
11.3.2. Asynchronous decomposition of generalized stochastic Petri nets . . . . .	332
11.3.3. Tensor analysis of phase-type Petri nets. . . . .	333
11.4. Tensor decomposition of stochastic well-formed Petri nets. . . . .	334
11.4.1. Problems . . . . .	335
11.4.2. The specification problem . . . . .	335
11.4.3. The resolution problem. . . . .	336

11.4.4. A tensor decomposition method for SWN . . . . .	338
11.4.5. Application in the asynchronous case . . . . .	340
11.5. Conclusion . . . . .	344
11.6. Bibliography . . . . .	344
<b>PART 2. VERIFICATION AND APPLICATION OF PETRI NETS . . . . .</b>	<b>347</b>
<b>Chapter 12. Verification of Specific Properties . . . . .</b>	<b>349</b>
Serge HADDAD and François VERNADAT	
12.1. Introduction . . . . .	349
12.2. Kripke structures and transitions systems . . . . .	352
12.3. Temporal logic . . . . .	354
12.3.1. Syntax and semantics . . . . .	354
12.3.2. Methods evaluation . . . . .	357
12.3.3. Temporal logic and Petri nets . . . . .	369
12.4. Behavioral approach . . . . .	371
12.4.1. Bisimulation relations . . . . .	375
12.4.2. Weak equivalences . . . . .	385
12.4.3. Modal characterizations of behavioral equivalences . . . . .	395
12.5. Decidability of bisimulation and of evaluation of formulas . . . . .	401
12.5.1. Undecidability results . . . . .	401
12.5.2. Decidability results . . . . .	407
12.6. Bibliography . . . . .	411
<b>Chapter 13. Petri Net Unfoldings – Properties . . . . .</b>	<b>415</b>
Jean-Michel COUVREUR and Denis POITRENAUD	
13.1. Introduction . . . . .	415
13.2. Elementary concepts . . . . .	416
13.2.1. Preliminary information . . . . .	416
13.2.2. Net homomorphisms . . . . .	417
13.2.3. Occurrence nets . . . . .	418
13.3. Branching processes and unfoldings . . . . .	421
13.3.1. Branching processes . . . . .	421
13.3.2. Unfoldings . . . . .	423
13.4. Finite prefixes . . . . .	424
13.4.1. Definition . . . . .	424
13.4.2. Adequate orders and complete finite prefixes . . . . .	425
13.4.3. Verification of safety properties . . . . .	427
13.4.4. Detection of infinite behaviors . . . . .	428
13.5. Conclusion . . . . .	432
13.6. Bibliography . . . . .	432

<b>Chapter 14. Symmetry and Temporal Logic</b> . . . . .	435
Serge HADDAD and Jean-Michel ILIÉ	
14.1. Introduction . . . . .	435
14.2. Principles of the dynamic symmetry method . . . . .	437
14.2.1. Verification of Kripke structures . . . . .	437
14.2.2. Symmetric Kripke structures . . . . .	439
14.2.3. Verification of symmetric Kripke structures . . . . .	441
14.3. Illustration of the dynamic symmetry method. . . . .	444
14.3.1. Presentation of the model . . . . .	444
14.3.2. Specification of the property to be verified . . . . .	452
14.3.3. Building of the symbolic synchronized product . . . . .	454
14.4. Efficient implementations and further work . . . . .	457
14.5. Conclusion . . . . .	458
14.6. Bibliography . . . . .	459
<b>Chapter 15. Hierarchical Time Stream Petri Nets</b> . . . . .	461
Patrick SÉNAC and Michel DIAZ	
15.1. Introduction . . . . .	461
15.2. Structured time stream Petri nets . . . . .	462
15.2.1. Motivations. . . . .	462
15.2.2. From composition to abstraction . . . . .	463
15.2.3. Verification of temporal coherence . . . . .	466
15.3. Combining abstraction and temporal composition . . . . .	468
15.3.1. Modularity and abstraction of temporal behaviors. . . . .	468
15.3.2. Hierarchical time stream Petri nets. . . . .	468
15.3.3. Interrupts as a combination of abstraction and temporal composition . . . . .	471
15.3.4. HTSPN state . . . . .	472
15.4. Examples . . . . .	474
15.4.1. Modeling hypermedia systems . . . . .	474
15.4.2. A solution to “lip-synchronization” using HTSPNs . . . . .	475
15.5. Conclusion . . . . .	477
15.6. Bibliography . . . . .	478
<b>Chapter 16. Petri Nets and Linear Logic</b> . . . . .	481
Brigitte PRADIN, Robert VALETTE and Nicolas RIVIÈRE	
16.1. Introduction . . . . .	481
16.2. Linear logic . . . . .	483
16.2.1. Bases: a logic which handles resources . . . . .	483
16.2.2. Connectors and their interpretation. . . . .	483
16.2.3. Sequent calculus. . . . .	484

16.3. Petri nets and linear logic . . . . .	485
16.3.1. Various approaches . . . . .	485
16.3.2. Approach with marking . . . . .	486
16.3.3. Equivalence between reachability in a Petri net and provability of one sequent in linear logic . . . . .	488
16.4. Sequent labeling and graph of precedence relations . . . . .	490
16.4.1. Labeling . . . . .	490
16.4.2. Graph of precedence relations . . . . .	491
16.4.3. Conflicts of transitions and tokens . . . . .	492
16.5. Temporal evaluation of scenarios . . . . .	493
16.5.1. Introduction . . . . .	493
16.5.2. Simple temporal networks . . . . .	494
16.5.3. Temporal labeling . . . . .	496
16.6. Conclusion . . . . .	499
16.7. Bibliography . . . . .	500
<b>Chapter 17. Modeling of Multimedia Architectures: the Case of Videoconferencing with Guaranteed Quality of Service . . . . .</b>	<b>501</b>
Philippe OWEZARSKI and Marc BOYER	
17.1. Introduction . . . . .	501
17.2. Problems of multimedia synchronization . . . . .	502
17.2.1. Multimedia information: characteristics and requirements . . . . .	502
17.2.2. Asynchronous distributed systems . . . . .	505
17.3. Modeling of multimedia synchronization constraints . . . . .	507
17.3.1. Modeling requirements . . . . .	507
17.3.2. Modeling example for a videoconference application . . . . .	510
17.4. Modeling of a synchronization architecture . . . . .	512
17.4.1. Introduction . . . . .	512
17.4.2. Modeling of a videoconference application . . . . .	512
17.4.3. Using a partial order transport . . . . .	516
17.5. Conclusion . . . . .	522
17.6. Bibliography . . . . .	523
<b>Chapter 18 Performance Evaluation in Manufacturing Systems. . . . .</b>	<b>527</b>
Isabel DEMONGODIN, Nathalie SAUER and Laurent TRUFFET	
18.1. Introduction . . . . .	527
18.2. Modeling of manufacturing systems . . . . .	528
18.2.1. Discrete event systems aspects . . . . .	529
18.2.2. Cyclic aspects . . . . .	533
18.2.3. High throughput aspects . . . . .	538
18.2.4. Weighted marked graphs . . . . .	543
18.3. Evaluation of manufacturing systems . . . . .	544

18.3.1. Performance evaluation methods . . . . .	544
18.3.2. Deterministic and stochastic discrete marked graphs . . . . .	549
18.3.3. Discrete weighted marked graphs . . . . .	551
18.3.4. Continuous weighted marked graphs . . . . .	552
18.4. Optimization of manufacturing systems . . . . .	559
18.4.1. Deterministic marked graphs . . . . .	560
18.4.2 Stochastic marked graphs. . . . .	561
18.4.3. Extension to deterministic weighted marked graphs . . . . .	563
18.4.4. Applications . . . . .	567
18.5. Conclusions . . . . .	571
18.6. Bibliography . . . . .	572
<b>Conclusion</b> . . . . .	<b>577</b>
<b>List of Authors</b> . . . . .	<b>579</b>
<b>Index</b> . . . . .	<b>581</b>



## Preface

Future advanced architectures, such as embedded systems, having a greater complexity and new quality requirements, will need a more precise specification and better control of their design process. In order to acquire the corresponding fundamental knowledge, it is essential to rely upon approaches based on the use of adequate system models. In particular, such approaches need to acquire a deep understanding of the system, including its local behaviors and its communications, based on a well-defined representation of the designed architecture. This representation should be used as early as possible to analyze and validate the design. The goal of this volume is to present a family of formal specification models, based on Petri nets and extensions of Petri nets, because they are defined by simple and clear semantics, allow easy modeling of system key mechanisms, and are supported by strong analysis methods and tools. Furthermore, this set of models can be used for all design aspects, i.e. to specify functional behaviors, and to include temporal or stochastic requirements.

The main results related to this approach are given in this volume, in two parts, one presenting the fundamental models, and the other being dedicated to verification and applications. We have tried to highlight the important characteristics and the main properties of these models, and to show how they lead to the emergence of a full design methodology, which is both complete, in terms of all possible functional and other analysis, and integrated, because the same basic semantics are used for the full design support. We think that this volume should greatly help any designer to build the new forthcoming generation of distributed systems.

Lastly, I would like to thank all the authors who contributed to this book, for their expertise, their seriousness, their technical inputs, and for the great job they have done.

*Michel DIAZ*

This page intentionally left blank

# Introduction

New technologies in processors and networks allow system designers to conceive and build advanced and sophisticated parallel and distributed architectures, which need to integrate non-functional real time and stochastic constraints with functional distributed processing and communication.

The global behavior of such systems depends first on the local activities and data, but also on the messages sent and received by the various interconnected sub-systems. As a matter of fact, understanding, expressing, specifying and validating such global behaviors proves to be a problem of very high complexity, leading to many design and implementation difficulties and bugs. For example, when considering  $n$  connected processors, they can run, at a given instant in time, using  $2 \times 2$ ,  $3 \times 3$  communications, etc., or a full communication, in which all  $n$  processors interact. The sum of the resulting combinations, of the order of  $2^n$ , shows the complexity of the resulting conceptual problems, and explains in particular the increasing difficulty obtained when passing from an interconnection of a few processors to an interconnection of a large number of processors: when the number of processors varies from 2 to 10, the difficulty coefficient goes from 4 to about 1,000.

It should then be clearly understood that designing such distributed architectures leads to a very complex conceptual task, which has to be based on a well-defined methodology to be able to manage all system requirements and behaviors.

## **Design and specification**

The design process starts by giving the different functions and agents which are required, and the way they are structured; second, the designers define the behaviors of the various processes and entities, and the way they communicate; then, if they want to analyze the correctness of the design as soon as possible, an adequate

approach is needed to represent, in an explicit way, the (full) system global behavior, in particular to be able to check potential unanticipated sub-behaviors.

To check the design correctness, it is essential to use a precise *model* of all critical mechanisms, functions, sub-systems, etc., and then, whenever possible, to use a *formal model*, to define a mathematical representation of the system. Checking the *correctness* validation of the design at this step is then conducted by checking the behavior of this system model.

Note that, after a given adequate sequence of more or less formal validation steps based on models, the system will be defined as ‘fully designed’ and will be implemented using adequate tools and languages.

*Formal approaches* have been used for many years for the verification of communication protocols. Two principal approaches have been used., i.e. basic formal models, such as automata, Petri nets, process algebras, etc., and formal description techniques for protocols, such as Estelle, LOTOS, SDL, etc.

This volume proposes and develops a design and validation methodology that relies on the use of a family of basic formal models that are rather easy to understand, and able to:

- describe the semantics of all basic building mechanisms;
- clearly specify the interconnection and communication semantics;
- unambiguously describe the resulting behaviors;
- validate the system during the first phases of its design by using support tools.

In general, basic non-language oriented graphical models, that do not include language-specific operators and statements, lead to the simplest solutions for representing basic mechanisms in a very abstract and integrated way.

For this reason too, this volume selected a basic, language-independent set of models to represent and manipulate the fundamental concepts of communicating architectures.

### **Selecting a *model***

Several models exist, and each model has particular characteristics, more or less relevant for a specific design. Consequently, the choice of the right model depends on the designed system and on the properties to be analyzed, as the model must be able to describe the design, and also to allow the designer to check the validity of the required properties.

*In general, the designer must have a good understanding of the fundamental semantics of the system, i.e. of its basic building mechanisms. Thus, for simple architectures, modeling will be able to represent in a faithful way all details of the system. However, for complex systems, it will generally be impossible, for economic reasons, to represent the details of all existing functions, and it will become necessary to select and validate certain building blocks, i.e those most likely to lead to erroneous behaviors.*

Of all existing models, *Petri nets* (PN) and their extensions are of undeniable fundamental interest, because they:

- provided the first modeling approaches for the semantics of concurrent systems, and were used to model the behaviors of the first parallel and distributed basic mechanisms;
- define easy graphic support for the representation and the understanding of these basic mechanisms and behaviors;
- prove to be, starting from state machines, an easy extension of previous approaches and handle, at the same time, the creation and the analysis of models;
- express very simply the main basic concepts in communication, including waiting and synchronization, and furthermore take into account their temporal and stochastic parameters;
- ensure, being unrelated to a particular implementation language, the independence of the specification with respect to its implementation.

Furthermore, many validation methods have been developed, using a great number of theoretical results and support tools, able to manipulate functional, temporal, and stochastic behaviors. Finally, models based on PNs will help us to understand, define and analyze the behavior of these systems, in the preliminary and first steps of their design.

For all these reasons, a set of Petri net models was selected in this book to represent and manipulate the fundamental concepts of communicating architectures.

## **Petri nets**

PNs were introduced by A.C. Petri in 1962 to synchronize communicating automata, and were then extended to define a large set of models, with increasing complexity and capabilities.

As will be seen, this family of PNs, starting from the simple traditional state machines, now allows system designers to handle in an integrated way the

functional (qualitative) and the non-functional (e.g. quantitative) temporal and stochastic capabilities of systems.

Extensions of PNs were proposed according to two important axes:

a) for *qualitative properties* and behaviors, to use simpler and more compact models, by high level PNs, for handling generic behaviors (e.g. individual) and data, predicates and functions;

b) complementing this first axis, for *quantitative properties* and behaviors, to extend the previous models by integrating quantitative constructs and parameters related to temporal and stochastic requirements.

It is significant to note that all first and conceptual studies in these quantitative fields were carried out using PN-based models.

### ***Functional qualitative properties***

The first PN model, called the Condition–Event PN, was based on the use of Boolean values: true or false. It was generalized by Place–Transition PNs, now simply called PNs, which can use integers. This volume will begin with their presentation and validation.

### ***Non-functional quantitative properties***

The fundamental contributions of the second axis considers:

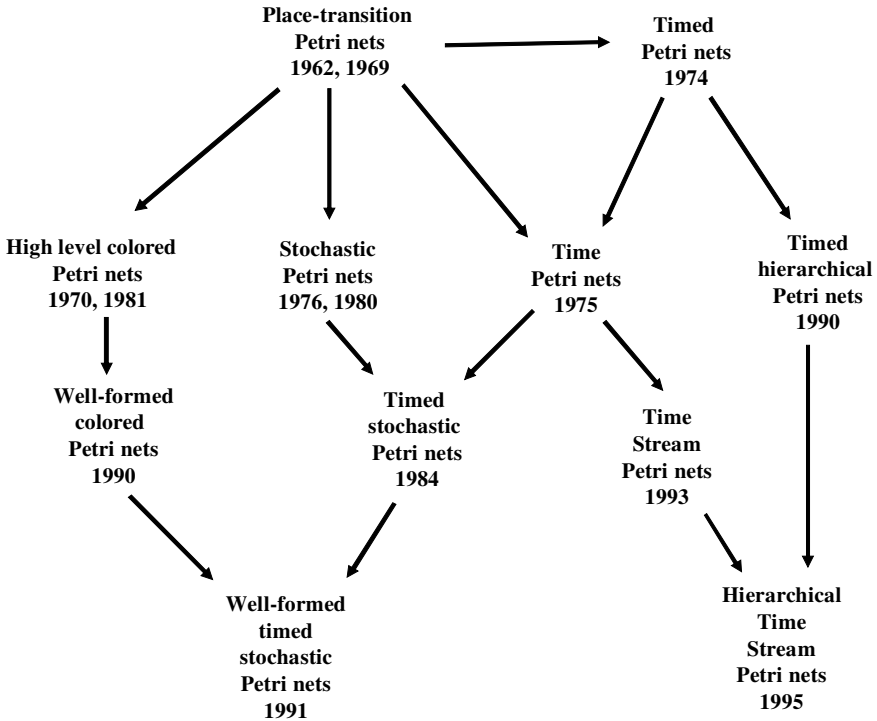
- *time PNs*, or TPNs, used for systems whose behaviors depend explicitly on temporal values;
- *stochastic PNs*, or SPNs, for which distributions are attached to the model, in particular for performance evaluation and reliability.

### **Families of PNs**

When applied to the modeling of systems, it rapidly becomes apparent that these models do not have the same application power, in terms of:

- definition and description of the concepts for parallelism, distribution, and synchronization;
- understanding and using the temporal and stochastic semantics;
- analyzing the possibly different mechanisms and behaviors, in very different contexts and applications.

Figure 1 represents some of the principal models of this family. In this figure, an arrow means that the model at the end of the arrow was proposed after the model at the beginning of the arrow, and so gives the steps followed by the research to propose and develop these principal PN-based models.



**Figure 1.** *The main Petri net models*

Figure 2 gives a more conceptual view of these models, by clarifying their syntactic and semantic relationships. In this figure, three fields are respectively defined by:

- a discrete state semantics, for non-temporal and non-stochastic nets, behaviors being represented by a finite graph of all model states;
- a semantics on continuous time, for extended behaviors based on dense time models;
- and stochastic semantics, for behaviors including distributions.

Let us emphasize that these models have three models of reference, respectively PN, TPN, and SPN. Moreover, each model is a pure extension of a previous one, as it can be simplified to become a basic PN model.

As seen in the figure, the models derived from the reference models:

- lead to more compact models, i.e. are **abbreviations**, that do not increase the expressiveness of the model, but simplify the model and the system specification;
- or are more powerful in terms of expressive power, i.e. are able to describe mechanisms which could not be described by the unextended models (e.g. introducing time parameters, stochastic distributions, etc., for real-time or dependable systems).

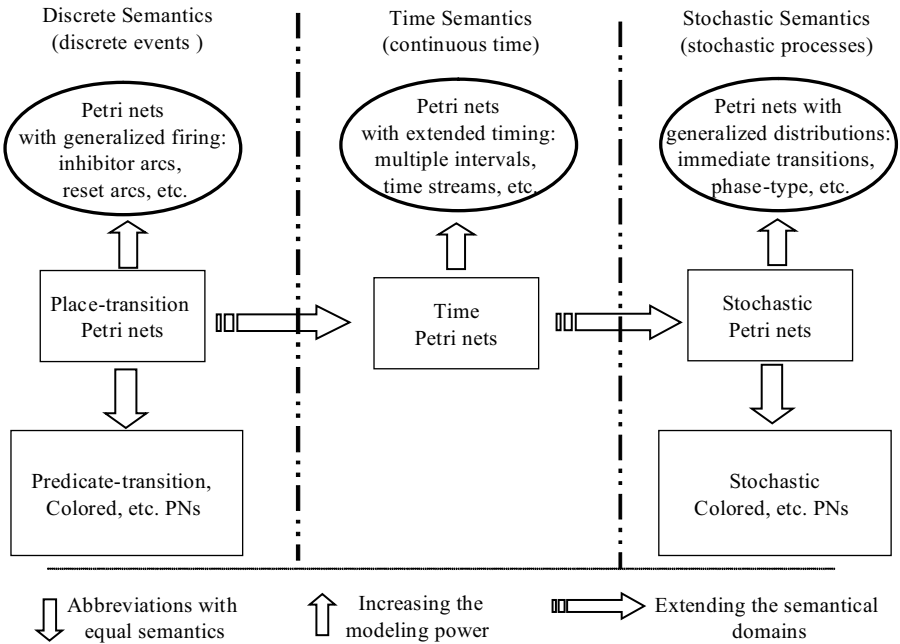


Figure 2. Semantics domains of the Petri net-based models

For example:

- PN led to PN with inhibitor arcs (to test the presence of zero token in one place), PN with reset arcs, etc.;
- TPN led to TPN with streams to compose and synchronize independent behaviors with independent temporal constraints, etc.;



– SPN led to SPN with immediate transitions in order to manage the case where transition cannot be delayed, etc.

Consequently, many different models exist, of different power and for different fields of application, but they follow the same semantics basis, and will allow the designers to carry out coherent complementary analyses to validate the correct operation of the modeled (and designed) systems.

The semantics of these models and their properties were used to select, define, and study the most important members of the PN family in the two parts of this volume.

## **Table of contents of the volume**

Part 1 is dedicated to fundamental models and contains 11 chapters. Part 2 addresses verification and applications, and contains the last 7 chapters.

### ***Part 1***

Chapter 1 introduces Place–Transition PNs, more simply called Petri Nets (PNs). It gives their fundamental definitions, presents some basic models and clarifies their interest.

Chapter 2 illustrates an application in a very important area: communication protocols; simple PN examples show at the same time the power of the model and the interest of the formal analysis.

Chapter 3 first introduces the general properties that can be checked using PNs (blocking, reachability or accessibility, etc.) and the verification approach that uses the graph of the reachable (or accessible) states. The set of reachable (or accessible) states is the set of states that are reachable or accessible from a given initial state. Two optimization methods of analysis are then presented, one based on linear algebra techniques, and the other one exploiting the topological structure of the PNs.

Chapter 4 deals with the decidability and complexity problems related to checking these general properties.

Chapters 5 and 6 consider models and behaviors based on explicit values of time, and show how to model temporal mechanisms.

Chapter 5 presents the general model, Time PN or TPN, which associates a given interval (minimum, maximum) to each transition; this gives the first semantics for handling time and verifying temporal behaviors.

Chapter 6 presents a general model for composing temporal behaviors and systems. It gives the semantics of temporal composition by a new model, Time Stream PN, for composing autonomous (temporal) flows. It emphasizes their interests and applications for systems having independent temporal constraints, which sometimes interact.

Chapters 7 and 8 again consider PNs, i.e. non-temporal PN models, but define an abbreviation of a PN by a general model, which becomes able to represent, in a very compact way, a given set of similar parallel behaviors. The problems associated with this abbreviation are, on the one hand, to define a compact formalism and, on the other hand, to propose new validation techniques to handle this model, i.e. to avoid the obvious solution that consists of unfolding it into a very large PN.

Chapter 7 presents the main PN abbreviations, while concentrating on Colored PNs, which is the most frequently used model.

Chapter 8 gives one well-defined version of this formalism, Well-formed Colored PNs, which allows the development of efficient analysis techniques.

Chapters 9, 10 and 11 introduce distributions, which take into account probabilistic properties of systems. They introduce stochastic PNs, or SPNs, and define their semantics in terms of stochastic processes, and, for some classes of models, their relationships with Markov chains. The principal methods of analyzing SPNs are then presented. Chapter 9 introduces stochastic PNs.

Chapter 10 introduces well-formed SPNs by combining the formalisms presented in Chapter 7 (Well-formed Colored PNs) and Chapter 9. Modeling a multiprocessor architecture illustrates the expressivity of this formalism and its interest for performance evaluation. Chapter 11 develops a tensorial composition of classical and well-formed SPNs, showing that such a compositional approach reduces the complexity of the corresponding validation.

## ***Part 2***

The second part of this volume presents important advanced analysis techniques and finally gives some significant and illustrative case studies.

Chapters 12 and 13 address checking and verifying non-temporal behaviors. They present the main approaches that are based on building and manipulating the

(system) accessibility or reachability graph, i.e. the graph representing all possible behaviors of the model. Checking these properties, by algorithms applied to the accessibility graph, suffers from the problems of combinatorial explosion. The general problems to be solved to control such an increase in the number of states, as well as general solutions, are then given. Three specific techniques, based respectively on the unfolding of the colored PNs, on symmetries, and on partial orders, are then presented.

Chapters 14 and 15 focus on the temporal validation of behaviors. Chapter 14 analyzes the relationships existing between symmetry and temporal logic for the verification of properties that depend on the specificities of the system. Chapter 15 introduces a parallel–serial hierarchy of temporal behaviors. This hierarchy simplifies the description of complex systems and is very well adapted for modeling complex multimedia and hypermedia objects, documents, and systems.

Chapter 16 presents how to use the main relationships that exist between linear logic and Petri nets for specification and validation. Logical reasoning is constructed based on the behavior of PNs that does not need to produce the reachability graph. The interest of linear logic is illustrated by showing in particular how to handle symbolic temporal intervals (minimum, maximum).

Chapters 17 and 18 present two important case studies that are illustrative while being manageable and easily understandable. Chapter 17 is devoted to the modeling and design of a multilayered, multimedia architecture that is able to guarantee temporal properties at the application level. Chapter 18 presents the application of PN-based models to performance evaluation in the field of computer-integrated manufacturing systems.

Finally, a conclusion summarizes the contents of this volume.

This page intentionally left blank

Part 1

Fundamental Models

This page intentionally left blank