Denis Caromel · Ludovic Henrio

A Theory of Distributed Objects

Asynchrony – Mobility – Groups – Components

Preface by Luca Cardelli

With 114 Figures and 48 Tables

Springer

Denis Caromel

University of Nice Sophia Antipolis
I3S CNRS – INRIA
Institut universitaire de France
2004 Rt. des Lucioles, BP 93
06902 Sophia Antipolis Cedex, France
e-mail: Denis.Caromel@inria.fr

Ludovic Henrio

University of Westminster
Harrow School of Computer Science
Watford Rd, Northwick Park
Harrow HA1 3TP, UK
e-mail: Ludovic.Henrio@m4x.org

Library of Congress Control Number: 2005923024

ISBN-10 3-540-20866-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-20866-2 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springeronline.com

Printed in Germany

Typeset by the authors using a Springer TEX macro package
Production: LE-TEX Jelonek, Schmidt & Vöckler GbR, Leipzig
Cover design: KünkelLopka, Heidelberg

Printed on acid-free paper 45/3142/YL - 5 4 3 2 1 0

Contents

Preface by Luca Cardelli . . . . . **VII**

Table of Contents . . . . . **IX**

Lists of Figures, Tables, Definitions and Properties . . . . **XV**

Prologue . . . . . **XXV**

Reading Path and Teaching . . . . . **XXIX**

Part I Review

1 Analysis **3**
1.1 A Few Definitions 3
1.2 Distribution, Parallelism, Concurrency 5
1.2.1 Parallel Activities 5
1.2.2 Sharing 6
1.2.3 Communication 6
1.2.4 Synchronization 10
1.2.5 Reactive vs. Proactive vs. Synchronous 11
1.3 Objects 14
1.3.1 Object vs. Remote Reference and Communication . . 14
1.3.2 Object vs. Parallel Activity 14
1.3.3 Object vs. Synchronization 15
1.4 Summary and Orientation 17

2 Formalisms and Distributed Calculi **21**
2.1 Basic Formalisms 21
2.1.1 Functional Programming and Parallel Evaluation . . . 21
2.1.2 Actors 23

2.1.3 π-calculus 26
2.1.4 Process Networks 30
2.1.5 ς-calculus 31
2.2 Concurrent Calculi and Languages 35
2.2.1 MultiLisp 35
2.2.2 PICT 37
2.2.3 Ambient Calculus 40
2.2.4 Join-calculus 42
2.2.5 Other Expressions of Concurrency 43
2.3 Concurrent Object Calculi and Languages 45
2.3.1 ABCL 45
2.3.2 Obliq and Øjeblik 49
2.3.3 The $\pi o \beta \lambda$ Language 51
2.3.4 Gordon and Hankin Concurrent Calculus: **conc**ς-calculus 54
2.4 Synthesis and Classification 56

Part II ASP Calculus

3 An Imperative Sequential Calculus 63
3.1 Syntax 63
3.2 Semantic Structures 65
3.2.1 Substitution 65
3.2.2 Store 66
3.2.3 Configuration 66
3.3 Reduction 66
3.4 Properties 68

4 Asynchronous Sequential Processes 69
4.1 Principles 69
4.2 New Syntax 71
4.3 Informal Semantics 71
4.3.1 Activities 72
4.3.2 Requests 73
4.3.3 Futures 73
4.3.4 Serving Requests 73

5 A Few Examples 75
5.1 Binary Tree 76
5.2 Distributed Sieve of Eratosthenes 77
5.3 From Process Networks to ASP 79
5.4 Example: Fibonacci Numbers 80
5.5 A Bank Account Server 81

Part III Semantics and Properties

6 Parallel Semantics . . . 87
6.1 Structure of Parallel Activities . . . 87
6.2 Parallel Reduction . . . 89
6.2.1 More Operations on Store . . . 89
6.2.2 Reduction Rules . . . 91
6.3 Well-formedness . . . 98

7 Basic ASP Properties . . . 101
7.1 Notation and Hypothesis . . . 101
7.2 Object Sharing . . . 104
7.3 Isolation of Futures and Parameters . . . 105

8 Confluence Property . . . 107
8.1 Configuration Compatibility . . . 107
8.2 Equivalence Modulo Future Updates . . . 111
8.2.1 Principles . . . 113
8.2.2 Alias Condition . . . 114
8.2.3 Sufficient Conditions . . . 115
8.3 Properties of Equivalence Modulo Future Updates . . . 117
8.4 Confluence . . . 118

9 Determinacy . . . 121
9.1 Deterministic Object Networks . . . 121
9.2 Toward a Static Approximation of DON Terms . . . 124
9.3 Tree Topology Determinism . . . 126
9.4 Deterministic Examples . . . 126
9.4.1 The Binary Tree . . . 126
9.4.2 The Fibonacci Number Example . . . 127
9.5 Discussion: Comparing Request Service Strategies . . . 130

Part IV A Few More Features

10 More Confluent Features . . . 137
10.1 Delegation . . . 137
10.2 Explicit Wait . . . 141
10.3 Method Update . . . 141

11 Non-Confluent Features . . . 143
11.1 Testing Future Reception . . . 143
11.2 Non-blocking Services . . . 144
11.3 Testing Request Reception . . . 145
11.4 Join Patterns . . . 146
11.4.1 Translating Join Calculus Programs . . . 146

11.4.2 Extended Join Services in ASP 147

12 Migration 151
12.1 Migrating Active Objects 151
12.2 Optimizing Future Updates 153
12.3 Migration and Confluence 154

13 Groups 157
13.1 Groups in an Object Calculus 157
13.2 Groups of Active Objects 160
13.3 Groups, Determinism, and Atomicity 162

14 Components 169
14.1 From Objects to Components 169
14.2 Hierarchical Components 170
14.3 Semantics 172
14.4 Deterministic Components 175
14.5 Components and Groups: Parallel Components 176
14.6 Components and Futures 178

15 Channels and Reconfigurations 181
15.1 Genuine ASP Channels 181
15.2 Process Network Channels in ASP 183
15.3 Internal Reconfiguration 184
15.4 Event-Based Reconfiguration 186

Part V Implementation Strategies

16 A Java API for ASP: ProActive 189
16.1 Design and API 189
16.1.1 Basic API and ASP Equivalence 190
16.1.2 Mapping Active Objects to JVMs: Nodes 191
16.1.3 Basic Patterns for Using Active Objects 192
16.1.4 Migration 192
16.1.5 Group Communications 195
16.2 Examples 198
16.2.1 Parallel Binary Tree 198
16.2.2 Eratosthenes 201
16.2.3 Fibonacci 206

17 Future Update 213
17.1 Future Forwarding 213
17.2 Update Strategies 215
17.2.1 ASP and Generalization: Encompassing All Strategies 215
17.2.2 No Partial Replies and Requests 217

17.2.3 Forward-Based 219
17.2.4 Message-Based 220
17.2.5 Lazy Future Update 222
17.3 Synthesis and Comparison of the Strategies 223

18 Loosing Rendezvous 225
18.1 Objectives and Principles 225
18.2 Asynchronous Without Guarantee 227
18.3 Asynchronous Point-to-Point FIFO Ordering 229
18.4 Asynchronous One-to-All FIFO Ordering 232
18.5 Conclusion 235

19 Controlling Pipelining 237
19.1 Unrestricted Parallelism 238
19.2 Pure Demand Driven 238
19.3 Controlled Pipelining 239

20 Garbage Collection 241
20.1 Local Garbage Collection 241
20.2 Futures 242
20.3 Active Objects 242

Part VI Final Words

21 ASP Versus Other Concurrent Calculi 245
21.1 Basic Formalisms 245
21.1.1 Actors 245
21.1.2 π-calculus and Related Calculi 246
21.1.3 Process Networks 248
21.1.4 ς-calculus 249
21.2 Concurrent Calculi and Languages 249
21.2.1 MultiLisp 249
21.2.2 Ambient Calculus 250
21.2.3 join-calculus 250
21.3 Concurrent Object Calculi and Languages 250
21.3.1 Obliq and Øjeblik 250
21.3.2 The $\pi o \beta \lambda$ Language 251

22 Conclusion 253
22.1 Summary 253
22.2 A Dynamic Property for Determinism 254
22.3 ASP in Practice 255
22.4 Stateful Active Objects vs. Immutable Futures 256
22.5 Perspectives 257

23 Epilogue 261

Appendices

A Equivalence Modulo Future Updates **269**
A.1 Renaming 269
A.2 Reordering Requests ($R_1 \equiv_R R_2$) 269
A.3 Future Updates 270
A.3.1 Following References and Sub-terms 270
A.3.2 Equivalence Definition 273
A.4 Properties of $\equiv_F$ 276
A.5 Sufficient Conditions for Equivalence 281
A.6 Equivalence Modulo Future Updates and Reduction 283
A.7 Another Formulation 288
A.8 Decidability of $\equiv_F$ 290
A.9 Examples 291

B Confluence Proofs **295**
B.1 Context 295
B.2 Lemmas 296
B.3 Local Confluence 298
B.3.1 Local vs. Parallel Reduction 299
B.3.2 Creating an Activity 300
B.3.3 Localized Operations (SERVE, ENDSERVICE) 301
B.3.4 Concurrent Request Sending: REQUEST/REQUEST . 304
B.4 Calculus with service based on activity name: $Serve(\alpha)$ . . . 305
B.5 Extension 306

References **309**

Notation **321**

Syntax of ASP Calculus **327**

Operational Semantics **329**

Overview of Properties **331**

Overview of ASP Extensions **333**

Index **343**