Contents

Preface **9**

1 Introduction **11**

1.1 Basic Terms . . . 14
1.2 Motivations and Contributions . . . 15
1.2.1 Atomic tasks and versioning . . . 15
Contribution . . . 18
1.2.2 Declarative synchronization . . . 20
Contribution . . . 20
1.2.3 Dynamic protocol update and rebinding . . . 22
Contribution . . . 23

2 Versioning for Atomicity **25**

2.1 Model . . . 27
2.1.1 Services and protocols . . . 27
Example protocol stack . . . 28
2.1.2 Tasks and atomic tasks . . . 29
2.2 Scheduling of Task Operations . . . 30
2.3 Versioning Algorithms for Atomic Tasks . . . 34
2.3.1 Basic versioning . . . 34
2.3.2 Supremum versioning . . . 36
2.3.3 Route versioning . . . 38
2.4 The SAMOA Protocol Framework . . . 41
2.5 Related Work . . . 42

3 Atomic Tasks **45**

3.1 Design Choices . . . 46
3.1.1 Fine-grain, rollback-free concurrency control . . . 46
3.1.2 Typing for safe rollback-free atomicity . . . 47

3.1.3 Isolation of operations with I/O effects 47
3.1.4 Task multithreading and nesting 48
3.2 Example Program 49
3.3 The Calculus of Atomic Tasks 51
3.3.1 Syntax 51
Types 51
Values and expressions 52
Threads and atomic tasks 52
Verlocks 52
3.3.2 Operational semantics 53
3.3.3 The Versioning Algorithm (VA) 56
Task creation and destruction 58
Serialized and isolated task evaluation 59
Verlock acquisition and release 60
Correctness assumptions 60
3.3.4 Typing 61
3.4 Type System 63
3.4.1 Absence of races 65
3.4.2 Absence of non-declared verlocks 66
3.4.3 Isolation preservation 69
Deadlocks 69
3.4.4 Proving type soundness 70
Type safety 70
Evaluation progress 71
3.5 Related Work 72
3.5.1 Atomic transactions 72
3.5.2 Atomic code blocks 73
3.5.3 Transaction models 74

4 Declarative Synchronization 77
4.1 Example Program 79
Synchronization policy 80
Combinators satisfiability 81
Typing for verification 82
4.2 The Calculus of Concurrency Combinators 82
4.2.1 Syntax 82
Services and composite services 82
Concurrency combinators 83
Combinator declarations 84
Types 85

Values 85
Expressions 85
Programs 85
4.2.2 Consistency of synchronization policy 86
4.2.3 Operational semantics 86
Basic evaluation rules 88
Folding and unfolding 90
Concurrency control 90
4.3 Example Reduction 91
4.4 Type System 93
4.4.1 Satisfiability of concurrency combinators 93
4.4.2 Typing for combinator satisfiability 94
4.4.3 Well-typed programs satisfy combinators 97
4.5 Role-based Synchronization (RBS) 98
4.5.1 Semantic roles 98
4.5.2 Synchronization constraints 99
Synchronization policy 99
Synchronization guards 100
4.5.3 The RBS constraint language 100
4.5.4 Example thread family 104
4.5.5 Implementation 106
4.6 Related Work 106
4.6.1 Separation of concurrency aspects 106
4.6.2 Aspect-oriented programming 107

5 Dynamic Protocol Update 109
5.1 Model 111
5.1.1 Basic terms 112
5.1.2 Distributed systems 113
5.1.3 Message passing 114
5.1.4 Protocol rounds 115
5.1.5 Dynamic bindings 116
5.2 Dynamic Protocol Update (DPU) 117
5.2.1 Replaceability 117
5.2.2 Global update 119
5.2.3 Local update 120
5.2.4 Correctness of dynamic protocol update 120
5.2.5 Properties of dynamic protocol update 122
5.3 Switching Algorithms 124
5.3.1 Synchronized protocol update 124

5.3.2 Lazy protocol update . . . 128
5.4 Example: Adaptive Group Communication . . . 130
5.5 Related Work . . . 131
5.5.1 Protocol frameworks for DPU . . . 132
5.5.2 Formalization of DPU . . . 133
5.5.3 Language support for DSU . . . 134

6 Dynamic Rebinding **135**
6.1 The Object Calculus of Dynamic Rebinding . . . 136
6.1.1 Syntax . . . 136
Types . . . 136
Signatures . . . 137
Methods . . . 137
Classes . . . 138
Values . . . 138
Basic expressions . . . 138
Dynamic binders and exceptions . . . 138
Threads . . . 139
Programs . . . 139
6.2 Properties of Dynamic Rebinding . . . 139
6.2.1 Language properties . . . 139
6.2.2 Semantic properties . . . 140
6.3 Example Program . . . 141
6.3.1 Solution using `atomic` . . . 144
6.3.2 Solution using concurrency combinators . . . 145
6.4 Operational Semantics . . . 146
6.4.1 Basic definitions . . . 146
6.4.2 Reduction rules . . . 149
6.5 Related Work . . . 150
6.5.1 Object calculi . . . 150
6.5.2 Dynamic rebinding . . . 151

7 Conclusions **153**
7.1 Atomic Tasks and Versioning . . . 153
7.2 Declarative Synchronization . . . 154
7.3 Dynamic Protocol Update and Rebinding . . . 154

A Correctness proofs **157**
A.1 Well-typed Programs Satisfy Isolation . . . 157
A.1.1 Absence of non-declared verlocks . . . 157

A.1.2 The main result of isolation preservation 160
A.1.3 Type soundness 161
Type safety 162
Evaluation progress 172
A.2 Dynamic Correctness of the VA Algorithm 175
A.2.1 Assumptions and definitions 175
A.2.2 Verlock access 176
A.2.3 Access ordering 178
A.2.4 Isolated task execution 180

Bibliography **181**

Index **193**

Streszczenie **203**