

# Contents

<b>1. Introduction</b>	1
1.1 Motivation	2
1.2 Specification and Verification Technique	4
1.3 The Problem	5
1.3.1 Modular Correctness	7
1.3.2 The Frame Problem	8
1.3.3 Modular Verification of Type Invariants	10
1.3.4 The Extended State Problem	11
1.3.5 Alias Control	13
1.4 Modularity Aspects of Programs, Specifications, and Proofs	16
1.4.1 Modularity of Programs	17
1.4.2 Modularity of Universal Specifications	21
1.4.3 Modularity of Interface Specifications	22
1.4.4 Modularity of Correctness Proofs	26
1.5 Approach, Outline, and Contributions	28
1.5.1 Approach	28
1.5.2 Outline	31
1.5.3 Contributions	32
1.6 Related Work	33
1.6.1 Specification Techniques	33
1.6.2 Verification and Analysis Techniques	36
<b>2. Mojave and the Universe Type System</b>	39
2.1 Mojave: The Language	39
2.1.1 The Language Core	39
2.1.2 Modularity	45
2.2 Universes: A Type System for Flexible Alias Control	51
2.2.1 The Ownership Model	52
2.2.2 The Universe Programming Model	54
2.2.3 Programming with Universes	58
2.2.4 Examples	60
2.2.5 Formalization of the Universe Type System	66
2.2.6 Discussion	70
2.3 Related Work	74

<b>3. The Semantics of Mojave</b> .....	77
3.1 Programming Logic .....	77
3.1.1 Formal Data and State Model .....	77
3.1.2 Axiomatic Semantics .....	92
3.1.3 Programming Logic .....	97
3.2 Language Properties .....	97
3.2.1 Type Safety .....	99
3.2.2 Liveness Properties .....	108
3.2.3 Properties of Readonly Methods .....	109
3.3 Correctness .....	110
3.3.1 Correctness of Closed Programs .....	110
3.3.2 Correctness of Open Programs: Modular Correctness ..	110
3.3.3 Modular Soundness .....	112
3.3.4 Composition of Modular Correct Open Programs .....	112
3.4 Related Work .....	120
<b>4. Modular Specification and Verification of Functional Behavior</b> .....	123
4.1 Foundations of Interface Specifications .....	123
4.2 Specification of Functional Behavior .....	125
4.2.1 Abstract Fields .....	125
4.2.2 Pre-post-specifications .....	131
4.3 Verification of Functional Behavior .....	135
4.3.1 Verification of Method Bodies .....	135
4.3.2 Proofs for Virtual Methods .....	136
4.3.3 Example .....	137
4.4 Related Work .....	139
4.4.1 Specification of Functional Behavior .....	139
4.4.2 Verification of Functional Behavior .....	141
<b>5. Modular Specification and Verification of Frame Properties</b> .....	143
5.1 Approach .....	144
5.1.1 Meaning of Modifies-Clauses .....	145
5.1.2 Explicit Dependencies .....	147
5.1.3 Modularity Rules .....	148
5.2 Formalization of Explicit Dependencies .....	155
5.2.1 Declaration of Dependencies .....	156
5.2.2 Axiomatization of the Depends-Relation .....	156
5.2.3 Consistency with Representation .....	157
5.2.4 Formalization of the Modularity Rules .....	159
5.2.5 Axiomatization of the Notdepends-Relation .....	160
5.2.6 Example .....	167
5.2.7 Discussion .....	169
5.3 Formalization of Modifies-Clauses .....	176

5.4	Verification of Frame Properties . . . . .	178
5.4.1	Verification of Method Bodies . . . . .	178
5.4.2	Local Update Property . . . . .	180
5.4.3	Accessibility Properties . . . . .	181
5.4.4	Modularity Theorem for Frame Properties . . . . .	183
5.4.5	Example . . . . .	184
5.5	Related Work . . . . .	188
5.5.1	Leino's and Nelson's Work on Dependencies . . . . .	188
5.5.2	Other Work on the Frame Problem . . . . .	193
<b>6.</b>	<b>Modular Specification and Verification of Type Invariants</b>	<b>195</b>
6.1	Motivation and Approach . . . . .	195
6.1.1	Invariant Semantics for Nonmodular Programs . . . . .	195
6.1.2	Problems for Modular Verification of Invariants . . . . .	197
6.1.3	Approach . . . . .	198
6.2	Specification of Type Invariants . . . . .	201
6.2.1	Declaration of Type Invariants . . . . .	201
6.2.2	Example . . . . .	203
6.2.3	Formal Meaning of Invariants . . . . .	204
6.3	Verification of Type Invariants . . . . .	204
6.3.1	Verification Methodology . . . . .	205
6.3.2	Example . . . . .	206
6.4	Discussion . . . . .	207
6.4.1	Module Invariants . . . . .	207
6.4.2	History Constraints . . . . .	208
6.5	Related Work . . . . .	209
<b>7.</b>	<b>Conclusion</b> . . . . .	<b>213</b>
7.1	Summary and Contributions . . . . .	213
7.2	The Lopex Project . . . . .	217
7.3	Tool Support . . . . .	217
7.4	Directions for Future Work . . . . .	219
<b>A.</b>	<b>Formal Background and Notations</b> . . . . .	<b>223</b>
A.1	Formal Background . . . . .	223
A.2	Notations . . . . .	225
<b>B.</b>	<b>Predefined Type Declarations</b> . . . . .	<b>227</b>
<b>C.</b>	<b>Examples</b> . . . . .	<b>229</b>
C.1	Doubly Linked List . . . . .	229
C.2	Property Editor . . . . .	235

<b>D. Auxiliary Lemmas, Proofs, and Models</b> .....	237
D.1 Auxiliary Lemmas and Proofs from Chapter 3.....	237
D.2 Auxiliary Lemmas and Proofs from Chapter 5.....	243
D.3 Auxiliary Lemmas and Proofs from Chapter 6.....	261
D.4 A Model for the Axiomatization of the Depends-Relation ....	266
<b>Bibliography</b> .....	271
<b>List of Figures</b> .....	285
<b>Index</b> .....	287



<http://www.springer.com/978-3-540-43167-1>

Modular Specification and Verification of  
Object-Oriented Programs

Müller, P.

2002, XIV, 298 p., Softcover

ISBN: 978-3-540-43167-1