

Table of Contents

Preface

Acknowledgement

Summary Table of Contents

Table of Contents

Chapter 1

Introduction	1
1.1 Open Programming Languages.....	4
1.2 Reflective Systems.....	4
1.3 Object-Oriented Frameworks.....	4
1.4 A Framework for an Object-Based Programming Language.....	5
1.5 A Layer for Object-Oriented Programming.....	6
1.6 A Layer for Reflective Object-Oriented Programming.....	8
1.7 Related Work.....	8

Chapter 2

Computational Reflection and Open Systems	11
2.1 Introduction.....	11
2.2 Model of Computation	12
2.3 Absorption and Reification in Programming Languages.....	14
2.4 Open Implemented Computational Systems	17
2.5 Reflection: Accessing One's Own Meta-system.....	22
2.5.1 Reflective Architectures.....	23
2.5.2 Reflective Facilities.....	24
2.5.3 Meta-Programming.....	25
2.6 Managing Reflective Overlap and Tower Architectures.....	26
2.7 Computational Systems with an Open Design.....	27
2.8 Full Abstraction and Compositionality in Programming Languages.....	28
2.8.1 Full Abstraction and Compositionality in Semantics of Programming Languages.....	29
2.8.2 Full Abstraction and Compositionality in Implementation of Programming Languages.....	31
2.9 Conclusion.....	33

Chapter 3

A Framework for Object-Based Programming Languages.....	35
3.1 Introduction.....	35
3.2 Design Issues in Object-Oriented Programming Languages.....	36
3.2.1 Objects, Interfaces, Messages and Encapsulation.....	40
3.2.2 Alternative Object Models.....	44
3.2.3 Operations on Objects.....	48
3.2.4 Classes and Class-based Inheritance.....	49
3.2.5 Classless Delegation.....	55
3.2.6 Mixin-Method Based Inheritance.....	58
3.2.7 Encapsulation as an Explicit Operation on Objects and Generators.....	61
3.2.8 Objects with State, State Changes and Object Identity.....	63
3.3 Object-Oriented Frameworks.....	65
3.3.1 Reusability in Object-Oriented Programs.....	66
3.3.2 Reusability in Object-Oriented Frameworks.....	71
3.3.3 Abstract Classes.....	71
3.3.4 Operations on Abstract Classes.....	72
3.3.5 Role of Abstract Classes in Frameworks.....	75
3.3.6 Frameworks, Conclusion.....	76
3.4 A Simple Object-based Programming Language.....	77
3.4.1 A Calculus for Object-based Programming.....	77
3.5 Definition of the Framework.....	81
3.5.1 Representation of Programs and Compositionality.....	81
3.5.2 Representation of Objects and Full Abstraction.....	85
3.5.3 Message Passing.....	86
3.6 Concretisation to a Simple Object-based Language.....	87
3.6.1 Abstraction Expressions and Object Structures.....	87
3.6.2 Association Expressions and Slots.....	89
3.6.3 Message Passing.....	90
3.6.4 Implementation of Simple, Summary.....	91
3.7 Improving the Framework.....	92
3.7.1 Reifier Methods.....	92
3.7.2 Extra Indirection Needed in Context and Client Objects.....	94
3.7.3 Evaluation Categories and Category Patterns.....	96
3.7.4 Making the Layered Structure Explicit.....	100
3.8 Conclusion.....	100

Chapter 4

Specialising the Framework with Inheritance	103
4.1 Introduction.....	103
4.2 Inheritance, Design Issues.....	104
4.2.1 Inheritance and Encapsulation Problems.....	104
4.2.2 The Need for Flexible and Controllable Inheritance.....	105
4.2.3 Multiple Inheritance.....	106
4.2.4 Mixin-based inheritance.....	118
4.2.5 Mixin-Method Based Inheritance.....	119
4.2.6 Mixin-based inheritance, A Solution to Multiple Inheritance Problems?.....	123
4.3 Visibility and Nesting in Object-Oriented Languages.....	126
4.3.1 Is There a Need for Scope Rules for Encapsulated Attributes?.....	127
4.3.2 Nested Classes, Classes as Attributes.....	129
4.3.3 Nested Mixins.....	130
4.4 Design of Agora.....	132
4.4.1 Introduction.....	132
4.4.2 Agora Syntax.....	133
4.4.3 Standard Agora Reifiers.....	134

4.5 The Agora Framework.....	141
4.5.1 Abstract Grammar, Expression Objects and Reifier Methods.....	142
4.5.2 Message Passing.....	144
4.5.3 Mixin Application and Object Structures.....	146
4.5.4 Agora Internal Object Structure.....	147
4.5.5 External Object Structures and Wrapper Objects.....	151
4.5.6 Extending Objects, Execution of Mixin Methods.....	152
4.5.7 Object Cloning.....	154
4.5.8 Mixin, Method and Instance Variable Declaration Reifiers and Slots.....	155
4.5.9 Summary of the Application of the Framework to Agora.....	156
4.6 Extensions to Agora.....	157
4.6.1 Public Instance Variables and Private Methods.....	157
4.6.2 Cloning Methods.....	158
4.6.3 Stubs for Multiple Inheritance.....	159
4.6.4 Single Slot Nested Objects.....	160
4.6.5 Classes.....	161
4.6.6 Abstract Methods, and Abstract Class Attributes.....	162
4.6.7 A Simple Form of Monotonic Reclassification.....	162
4.6.8 Classifiers.....	163
4.7 Conclusion.....	164

Chapter 5

A Reflective Framework.....	165
5.1 Introduction.....	165
5.2 Object-based Reflection.....	166
5.2.1 Linguistic Symbiosis.....	166
5.2.2 Simple Meta-Programming Operators for Agora.....	174
5.2.3 Simple Reflection Operators.....	177
5.2.4 Nature of Meta-Programs and Reflective Overlap.....	179
5.2.5 Dynamic Reflection and Infinite Regress.....	180
5.2.6 Abstraction and Compositionality.....	182
5.3 Object-Oriented Reflection.....	183
5.3.1. The Evaluation and Declaration of Reifiers.....	183
5.3.1. Need for a More Fine-Grained Linguistic Symbiosis.....	187
5.4 Conclusion and Open Issues.....	189

Chapter 6

Related Work.....	191
6.1 Reflection and Open Systems.....	191
6.2 Object-Oriented Reflection.....	192
6.3 Object-Oriented Systems.....	193
6.3.1 Object-Oriented Frameworks.....	194
6.3.2 Mixin-Based Inheritance.....	194

Chapter 7

Conclusion.....	197
7.1 Contributions.....	198
7.2 Future work.....	199

Bibliography

Index

Appendix A

Appendix B

