

Iain Craig

The Interpretation of Object-Oriented Programming Languages

Suranaree University of Technology



31051000623004



Springer

Contents

1. Introduction	1
1.1 Introduction	1
1.2 Essential Properties of Objects	2
1.3 Objects and Messages	5
1.4 Pure and Impure Languages	7
1.5 Mixed-Paradigm Languages	8
1.6 Organization of this Book	8
2. Class Fundamentals	13
2.1 Introduction	13
2.2 Classes	15
2.3 Instances	19
2.4 Slots and Methods	21
2.5 Slot Access	22
2.6 Visibility and Accessibility	24
2.7 Instance Creation	29
2.8 Inheritance	32
2.8.1 Introduction	32
2.8.2 Definition of Inheritance	33
2.9 Abstract Classes	38
2.10 Iterators	42
2.11 Part Objects	46
3. Prototype and Actor Languages	53
3.1 Introduction	53
3.2 Prototype Languages	53
3.2.1 The Concept of the Prototype	54
3.2.2 Slots and Methods	59
3.2.3 Message Passing	60
3.2.4 Creating New Objects	61
3.2.5 Delegation and Shared Structure	62
3.3 Methods in Prototype Languages	66
3.4 Actor Languages	67
3.4.1 Introduction	67

3.4.2	Actors	68
3.4.3	Extensions to the Actor Concept	72

4.	Inheritance and Delegation	77
4.1	Introduction	77
4.2	Interpretations of Inheritance	78
4.3	Inheritance as Subtyping	79
4.4	Inheritance as Code Sharing	80
4.5	Single Inheritance	83
4.6	Calling More Abstract Methods	85
4.7	Multiple Inheritance	91
4.8	Multiple Inheritance Graph Shape	93
4.9	Approaches to Multiple Inheritance	98
4.9.1	Tree Inheritance	99
4.9.2	Graph Inheritance	100
4.9.3	Linearized Inheritance	102
4.10	Implemented Multiple Inheritance Techniques	104
4.10.1	The CLOS Search Method	104
4.10.2	Multiple Inheritance in C++	105
4.10.3	Multiple Inheritance in Eiffel	106
4.11	Mixin Classes	109
4.12	Alternatives to Multiple Inheritance	111
4.12.1	Perspectives	111
4.12.2	Interfaces in Java	112
4.13	Delegation and Prototypes	113
4.14	Aggregation	115
5.	Methods	119
5.1	Introduction	119
5.2	Methods and Objects	121
5.3	Object Constructors and Methods	123
5.4	Environments and Closures	125
5.4.1	Introduction	125
5.4.2	Environments: A More Formal Definition	126
5.4.3	Blocks in Smalltalk and SELF	128
5.4.4	Block Structure in Beta	132
5.4.5	Higher-Order Methods	133
5.5	Methods and Inheritance	135
5.5.1	Method Hierarchies in Theta	135
5.5.2	Method Combination in CLOS	136
5.6	Static and Dynamic Binding	137

6. Types I: Types and Objects	143
6.1 Introduction	143
6.2 Inheritance and Types	145
6.2.1 Telling What the Type Is	147
6.3 Polymorphism	151
6.3.1 Signatures	151
6.4 Genericity	153
6.5 Overloading and Over-Riding	156
6.6 Languages with Root Classes	160
6.7 Polyadicity and Default Parameters	161
6.7.1 Variance	162
6.8 Downcasting and Subtypes	165
6.9 Review	167
7. Types II: Types and Objects—Alternatives	169
7.1 Introduction	169
7.2 Types and Implementations	169
7.3 Hiding Implementation Details	174
7.4 Classes and Type Operations	177
7.5 Containers and Objects	180
8. Reflection	183
8.1 Introduction	183
8.2 Class and Meta Class	185
8.3 Meta Class and Reflection	187
8.3.1 Introduction	187
8.3.2 Redefinition of Instance Creation Methods	188
8.3.3 Redefinition of Inheritance Methods	190
8.3.4 Redefinition of Method Call Protocols	190
8.3.5 Slots as Objects: An Example of Reification	191
8.3.6 Other Examples	192
8.4 Meta-Object Protocols	194
8.5 Self Representation, Abstract Syntax and Abstract Classes	195
8.6 Reflection in Java	197
8.7 Reflection in Prototype-based Languages	198
8.8 Prospects for the Future	201
9. Mixed-Paradigm Languages	203
9.1 Introduction	203
9.2 <i>Functional Programming: An Overview</i>	205
9.2.1 Control Structures and Semantics	209
9.2.2 Evaluation Strategies	210
9.2.3 Higher-Order Functions	214
9.2.4 Hindley-Milner Type Inference	217
9.2.5 Syntactic Sugar	218

XII Contents

9.3	An Impure Language	220
9.3.1	The Object-oriented Component	221
9.3.2	The Functional Component	231
9.4	Review	238

References	244
------------------	-----

Index	251
-------------	-----