2nd Edition

Principles of Object-Oriented Software Development

Anton Eliëns

Suranaree University of Technology
31051000646980

Addison-Wesley

Contents

Foreword v

Foreword to the first edition vi

Preface ix

I Designing Object-Oriented Systems

1 Introduction 3
1.1 Themes and variations 4
1.1.1 Object terminology 5
1.1.2 Object computation 8
1.1.3 Design by Contract 13
1.2 Paradigms of programming 17
1.2.1 Procedural programming 18
1.2.2 Data abstraction 19
1.2.3 Object-oriented programming 20
1.3 The object-oriented software life-cycle 21
1.3.1 Analysis 24
1.3.2 Design 26
1.3.3 Implementation 29
1.4 Beyond object-orientation? 31

2 Idioms and patterns 37
2.1 Polymorphism 38
2.1.1 Inheritance and delegation in Java 38
2.1.2 Polymorphism in C++ 41
2.1.3 Assertions in C++ 42
2.1.4 Canonical class idioms 45
2.2 Idioms in *hush* 46

2.2.1 The handle/body idiom 50
2.2.2 Virtual self-reference 53
2.2.3 Dynamic role-switching 54
2.2.4 The art of *hush* programming 57
2.3 A catalogue of design patterns 60
2.3.1 Creational patterns 62
2.3.2 Structural patterns 63
2.3.3 Behavioral patterns 64
2.4 Event-driven computation 67
2.4.1 The Reactor pattern 67
2.4.2 Abstract event systems 70

3 Software engineering perspectives 77
3.1 Development methods 78
3.1.1 Perspectives of modeling 80
3.1.2 Requirements engineering – *Fusion* 82
3.1.3 Methods for analysis and design – a comparative study . . 84
3.2 Identifying objects 86
3.2.1 Modeling heuristics 88
3.2.2 Assigning responsibilities 91
3.2.3 Object roles and interaction 94
3.3 Contracts 95
3.3.1 Specifying contractual obligations 96
3.3.2 Refining contracts 99
3.3.3 Runtime consistency checking 102
3.4 Towards a formal approach 103

4 Application development 107
4.1 The *drawtool* application 108
4.2 Guidelines for design 120
4.2.1 Individual class design 121
4.2.2 Inheritance and invariance 122
4.2.3 An objective sense of style 123
4.3 From specification to implementation 126
4.3.1 Structural versus behavioral encapsulation 126
4.3.2 Model-based specification 127
4.3.3 Abstract systems and events 132

II Object-Oriented Languages and Systems 139

5 Object-oriented programming languages 141
5.1 The object paradigm 142
5.1.1 A classification of object-oriented languages 144
5.1.2 Alternative object models 146
5.1.3 Object extensions of Lisp, C and Prolog 147

5.1.4 Script languages – integration with Java 149
5.2 Comparing Smalltalk, Eiffel, C++ and Java 151
5.2.1 Criteria for comparison 151
5.2.2 Language characteristics 152
5.3 Design dimensions of object-oriented languages 156
5.3.1 Object-based versus object-oriented 157
5.3.2 Towards an orthogonal approach – type extensions 158
5.3.3 Multi-paradigm languages – logic 160
5.3.4 Active objects – synchronous Java/C++ 163
5.4 Prototypes – delegation versus inheritance 166
5.4.1 Alternative forms of sharing 166
5.4.2 Implementation techniques – Self 168
5.5 Meta-level architectures 170

6 Component technology 177
6.1 Objects versus components 178
6.1.1 Definitions 178
6.1.2 The technology matrix 179
6.1.3 Component myths 181
6.2 Standards for interoperability 182
6.2.1 Object linking and embedding – COM 182
6.2.2 Object Request Brokers – CORBA 183
6.2.3 Persistent objects – ODMG 185
6.3 The Java platform – technology innovation 187
6.4 An Internet-based workgroup application 190
6.5 Crush – extending *hush* with CORBA 192

7 Software architecture 205
7.1 Elements of architecture 206
7.2 Case study – multimedia feature detection 209
7.3 Crossing boundaries 216
7.3.1 Embedded logic – crossing the paradigm boundary 216
7.3.2 Native objects – crossing the language boundary 218
7.3.3 Combining Java and C++ 223
7.4 Architectural patterns and styles 227
7.4.1 From technology to style 228
7.4.2 Case study – perspectives in visualization 229
7.5 Cross-platform development 231

III Foundations of Object-Oriented Modeling 237

8 Abstract data types 239
8.1 Abstraction and types 240
8.1.1 Abstraction in programming languages 240
8.1.2 A foundational perspective – types as constraints 241

8.1.3 Objectives of typed OOP 243
8.2 Algebraic specification 245
8.2.1 Signatures – generators and observers 246
8.2.2 Equations – specifying constraints 249
8.2.3 Initial algebra semantics 252
8.2.4 Objects as algebras 255
8.3 Decomposition – modules versus objects 260
8.3.1 Abstract interfaces 262
8.3.2 Representation and implementation 263
8.3.3 Adding new generators 266
8.3.4 Adding new observers 267
8.4 Types versus classes 270

9 Polymorphism **277**
9.1 Abstract inheritance 278
9.2 The subtype relation 280
9.2.1 Types as sets 280
9.2.2 The subtype refinement relation 281
9.2.3 Objects as records 284
9.3 Flavors of polymorphism 287
9.4 Type abstraction 290
9.4.1 A simple type calculus 292
9.4.2 Intersection types 296
9.4.3 Bounded polymorphism 300
9.5 Existential types – hiding 304
9.6 Self-reference 307

10 Behavioral refinement **319**
10.1 Types as behavior 320
10.2 Verifying behavioral properties 322
10.2.1 State transformers 322
10.2.2 Assertion logic 324
10.3 On the notion of behavior 326
10.4 Objects as behavioral types 330
10.4.1 Abstraction and representation 330
10.4.2 The correspondence relation 334
10.5 Specifying behavioral compositions 338

IV Object-Oriented Application Frameworks **347**

11 Business process redesign **349**
11.1 Business objects – SanFrancisco framework 350
11.2 Business process modeling 353
11.2.1 Logistics-based modeling 353
11.2.2 Business process simulation 355

11.2.3 Example – requests for loans 357
11.3 Object-oriented simulation 359
11.4 Visualization support 367
11.4.1 So many users, so many perspectives 368
11.4.2 DIVA – distributed visualization architecture 369
11.5 Migrating from legacy applications 372

12 Web applications 377
12.1 Objects and the Web 378
12.1.1 Trends and technologies 379
12.1.2 The Object Web – CORBA/Java versus Microsoft 381
12.2 Programming the Web – a search for APIs 384
12.2.1 Models of computation 384
12.2.2 Intelligent agents 386
12.3 The DejaVU experience – jamming (on) the Web 388
12.4 Software architectures revisited 395

Appendices 403

A The language Smalltalk 403
B The language Eiffel 411
C The language C++ 417
D The language Java 425
E The language DLP 433
F Unified Modeling Language 441
G Interface Definition Language – IDL 447
H Hello (CORBA) Universe 451
I Software development projects 463
J Answers to questions 467

References 479

Index 493