

# Contents

---

Figures	xvii
Tables	xxv
Preface	xxvii
Part 1 Introduction	
1 Introduction	3
1.1 Why Analysis or Design?	3
1.1.1 Object and object-orientation defined	6
1.1.2 eXtreme Programming	7
1.2 Why Analysis <i>and</i> Design?	9
1.2.1 Terminology problems	9
1.2.2 History	10
1.2.3 The fundamental difference between analysis and design	12
1.2.4 Business systems analysis, systems analysis, requirements analysis and subject matter analysis	13
1.2.5 The boundary between analysis and design	16
1.3 Case Studies	18
1.3.1 POIROT – a crime system	18
1.3.2 ALICE – a restaurant system	19
1.3.3 STREPT – an academic system	19
1.3.4 POLLY – a hotel system	20
Exercises	20
2 Three Models	23
2.1 Models and Modeling	23
2.2 The Three Models	25
2.2.1 Subject matter model: first model	27
2.2.2 Object type model: second model	33
2.2.3 Technical model: third model	38
2.3 A Greater Continuity	42
2.4 Iterative Development	43
2.5 An Agile Method	45
2.6 Other Approaches	46
2.6.1 Functionality driven methods	46
Exercises	49

3	Model Presentation and Packaging	51
3.1	The UML	51
3.1.1	Introduction	51
3.1.2	Making sense of the UML	52
3.1.3	Would we always use the UML?	53
3.2	Diagrams	54
3.3	Packaging	54
3.4	Choosing Packages	55
3.4.1	The mission statement	56
3.5	Legacy Systems	57
3.6	Case Study	57
3.6.1	Packaging ALICE	57
	Exercises	60
Part 2 Analysis		
4	Analysis	63
4.1	“Real Programmers Don’t Analyze”	63
4.2	Bespoke Programs Make Analysis Inevitable	64
4.3	“But I Already Understand!”	64
4.4	When Don’t We Analyze?	65
	Exercises	66
5	Analysis Inputs	67
5.1	Requirements	67
5.1.1	Separating the requirements	69
5.1.2	The nature of requirements	70
5.1.3	Staged development	75
5.1.4	Change cases	76
5.1.5	Checking and re-checking the requirements	76
5.1.6	Checking the requirements coverage	77
5.1.7	Recognizing the limits of requirements	77
5.2	The Subject Matter	78
5.3	A Previous or Existing System	78
5.4	An Architectural Vision	78
5.5	Mock-ups	79
5.6	Analysis Patterns	80

5.7	Case Study	80
5.7.1	Requirements practice	80
	Exercises	85
6	The Subject Matter Model	89
6.1	What is Modeling?	90
6.1.1	Abstraction	90
6.1.2	Engineering models, sculptures, paintings and maps	91
6.1.3	Chunks, cohesion and coupling	92
6.2	The Primary Model Element: Entities	95
6.2.1	The candidates	95
6.2.2	What shall we call them?	98
6.2.3	Finding entities	99
6.2.4	Assessing entities	101
6.2.5	Depicting entities	106
6.2.6	Characterizing entities	106
6.3	Secondary Model Elements: Properties and Connections	107
6.3.1	Attributes	107
6.3.2	Entities are not classes; attributes are not variables	112
6.3.3	Associations	113
6.3.4	Aggregation	129
6.3.5	Conflict resolution	132
6.4	Case Study	133
6.4.1	Finding the initial, candidate entities	133
6.4.2	Finding the attributes	137
6.4.3	Relationships	148
	Exercises	163
7	Subject Matter Model: Further Aspects	167
7.1	Classification (Generalization)	167
7.1.1	Finding generalizations	173
7.1.2	Assessing generalizations	174
7.1.3	Depicting generalizations	174
7.1.4	Pitfalls of generalizations	177
7.2	Case Study	177
7.2.1	Varieties of staff and things to eat and drink	179
7.3	Entity Life Patterns (State Machines)	185
7.3.1	Things happen; things change state	185
7.3.2	Happenings were the traditional route	185

## Contents

7.3.3	We will marginalize actions and processing	186
7.3.4	We might model state though	186
7.3.5	Another way to approach state	187
7.3.6	State machines and entity life histories	189
7.3.7	State machines	190
7.3.8	State machine organization	197
7.3.9	Frills and furbelows	207
7.4	Case Study	209
7.4.1	The life of a table	212
7.5	Dealing with Algorithms	216
7.5.1	Analysis, design and processing	216
7.5.2	Activity diagrams	218
7.5.3	The game's off if you haven't got requirements	227
7.6	Frills and Furbelows	227
7.6.1	Constraints	227
7.6.2	Exotic UML provisions	228
7.6.3	Event list	230
7.7	First Model Summary	231
7.7.1	You can rely on state	231
7.7.2	State presents itself in many ways	232
7.7.3	Communication and actions lead us astray	233
7.7.4	Avoiding analysis paralysis	233
	Exercises	234
8	Systems Analysis	237
8.1	Computerization	237
8.2	Differentiating Systems Analysis	238
8.3	Re-computerization	239
8.4	Mixtures	239
8.5	Subject Matter Analysis Should Come First	240
8.6	Depicting Systems Analyses	241
	Exercises	241
Part 3 Design		
9	Design	247
9.1	Defining Design	248
9.2	Coping With Complexity	249

9.3	Making Best Use of the Technology	250
9.4	Malleability and the Impact of Change	250
9.5	Modularity	252
9.5.1	Cohesion	253
9.5.2	Structure	254
9.5.3	Coupling	255
9.5.4	Separate solvability	255
	Exercises	255
10	Design Inputs	257
10.1	Architecture Proposals	257
10.2	Technology Choices	258
10.3	Requirements During Design	258
10.4	Analysis	259
10.5	Design Patterns	259
10.6	Spiral Development	260
11	The Object Type Model	261
11.1	What Do We Want?	261
11.2	Objects, not Classes	262
11.3	Objects, not Entities	263
11.4	Why Now?	264
11.5	The Naïve Approach	264
11.6	Outside-In Design	265
11.7	Type Design	266
11.8	CRC Workshops	267
11.8.1	Or T/CRC workshops	267
11.8.2	Background	267
11.8.3	The inputs to CRC	268
11.8.4	The CRC session	268
11.8.5	A minor dilemma: relationships	270
11.8.6	CRC session details	271
11.8.7	CRC cautions	278
11.9	Outputs of the Object Type Model	279
11.10	Depicting the Object Type Model	279
11.10.1	Object types	279
11.10.2	Interaction diagrams (sequence diagrams)	280

11.10.3	Structure diagrams	288
11.10.4	State machines	291
11.11	Case Study	293
11.11.1	Structural (naïve) approach	294
11.11.2	Making a booking	297
11.11.3	Structure revisited	310
11.11.4	The interfaces	311
11.12	Second Model Summary	312
11.12.1	Know your object instances	312
11.12.2	Work from the outside inward	313
	Exercises	313
12	The Technical Model	315
12.1	The Three-Model Proposal Revisited	316
12.2	What Have We Got?	317
12.3	What Do We Want?	317
12.3.1	Primary deliverables	317
12.3.2	Secondary deliverables	321
12.3.3	Just a minute; what about inheritance?	326
12.4	The USPs of Object-Orientation Revisited	326
12.4.1	An organizing principle	327
12.4.2	Encapsulation and information hiding	327
12.4.3	Flexibility	327
12.5	The Type System	328
12.5.1	Keep it simple	328
12.5.2	Implementing abstract types	329
12.5.3	Good types, bad types	337
12.5.4	Depicting abstract types	339
12.5.5	Choosing abstract types	340
12.6	Concrete Classes	341
12.6.1	CRC still helps	344
12.6.2	Sequence diagrams still help	345
12.6.3	Depicting instances	345
12.6.4	Good classes, bad classes	345
12.6.5	Implementing classes	350
12.6.6	Class names	351
12.6.7	Class metrics	352
12.6.8	Entity classes versus attribute classes	354
12.6.9	Depicting concrete classes	357

<b>12.7 Relationships</b>	360
12.7.1 Interface relationships	360
12.7.2 Inter-class relationships	364
12.7.3 Inter-instance relationships	365
<b>12.8 Case Study</b>	378
12.8.1 Eating	378
12.8.2 Seating	380
<b>12.9 Implementation Inheritance and Abstract Classes</b>	383
12.9.1 The origins of inheritance	383
12.9.2 The nature of inheritance	384
12.9.3 What happens exactly?	385
12.9.4 Overriding and metrics	388
12.9.5 Minimizing the class coupling	389
12.9.6 Self, overriding and super	391
12.9.7 Abstract classes	393
12.9.8 Inheritance of implementation versus composition	400
12.9.9 Being sensible about multiple inheritance	403
12.9.10 Pitfalls of implementation inheritance	404
12.9.11 Depicting abstract classes and inheritance	405
<b>12.10 Case Study</b>	408
12.10.1 Eating	408
<b>12.11 Strategic Structures</b>	412
<b>12.12 Properties</b>	414
<b>12.13 Methods</b>	415
12.13.1 Method cohesion	415
12.13.2 Method coupling	415
12.13.3 Method visibility	417
12.13.4 Method names and signatures	418
12.13.5 Exceptions	420
12.13.6 Depicting methods	421
12.13.7 Method specification	425
12.13.8 Further analysis	428
<b>12.14 Instance Variables</b>	429
12.14.1 Always private?	429
12.14.2 The data structure and being honest	430
12.14.3 What are they really called in the UML?	431
<b>12.15 Construction and Constructors</b>	431
12.15.1 Constructor coupling	431
12.15.2 Factories and prototypes	432
<b>12.16 Class Variables and Class Methods</b>	433

## Contents

12.17 State Machines	433
12.17.1 Interface specifications	434
12.17.2 Implementation state machines	437
12.17.3 Other cautions	439
12.18 Generic (Template) Classes	440
12.19 Case Study	441
12.19.1 DiningTable	441
12.19.2 Dish and recipe	442
12.19.3 Drink and CellarDrink	442
Exercises	452

## Appendices

A References	455
B Thumbnail Sketch of Object Technology	459
C UML	467
C.1 Introduction	467
C.2 History	467
C.3 Using This Appendix	468
C.4 Infrastructure of the UML	469
Superstructure	469
Infrastructure	469
Object Constraint Language	471
Diagram Interchange	472
C.5 (Super)structure of the UML	472
C.6 Structural Diagrams	474
C.6.1 “Class” diagrams	474
Compartments	477
Interfaces	478
Features	480
Comments	484
Packages	485
Associations	486
Aggregation	491
Constraints	492
Generalization	495
Implementation	497

	Data types	498
	Dependency relationships	498
	Association classes	500
	Power types	501
C.6.2	Composite structure diagrams	502
	Structure and nesting	502
	Collaboration	503
	Connectors	504
	Ports	504
C.6.3	Component diagrams	504
C.6.4	Deployment diagrams	506
<b>C.7</b>	<b>Behavioral Diagrams</b>	507
C.7.1	Sequence diagrams	507
	The instance or lifeline	508
	Messages	509
	Type versus class	509
	Methods	510
	Returns and other message varieties	510
	Nested messages and methods	512
	Loops and branches	513
	Birth and death	515
C.7.2	Communication diagrams	515
C.7.3	Interaction overview diagrams	516
C.7.4	Timing diagrams	517
C.7.5	State machines	518
	Introduction	518
	States (simple states)	519
	Transitions and events	520
	Guards	522
	Composite states and hierarchical state machines	523
	History pseudo state	524
	Initial pseudo state	525
	Entry points and exit points	526
	Local and external transitions	527
	Concurrent regions	528
	Forked transitions	530
	Final pseudo states	530
	Submachine states	532
	Activities	532
	Completion transitions	535
	Postconditions	535
	State invariants	535

	Events revisited	536
	State machine specialization.	537
	Deferred events	538
C.7.6	Activity diagrams	538
	Introduction	539
	Action nodes	540
	Flows	542
	Pins	543
	Object (or data) nodes	543
	Data stores	544
	Soapbox interlude	544
	Decisions and merges	545
	Forks and joins	546
	Loops	547
	Expansion regions	547
	Initial nodes and final nodes	547
	Activities	548
	Streaming parameters	549
	Exception output parameters	550
	Local preconditions and postconditions	550
	Partitions	551
	More	551
C.7.7	Use cases	551
	System boundary	552
	Use cases	553
	Actors	553
	Generalization	554
	Extends	554
	Includes	555
<b>D</b>	<b>Glossary</b>	<b>557</b>
<b>E</b>	<b>Summary of Practices and Principles</b>	<b>565</b>
<b>F</b>	<b>The Forbidden Words</b>	<b>575</b>
<b>G</b>	<b>Entity Life Histories</b>	<b>577</b>
	Index	583