

Contents

Part I Concepts and Notations

1	Programming Paradigms	3
1.1	History of Increasing the Level of Abstraction	3
1.2	Object-Oriented Versus Other Programming Paradigms.....	6
2	Basic Principles of the Object-Oriented Paradigm	9
2.1	Abstraction	9
2.2	Encapsulation	12
2.3	Modularity	12
3	Object-Oriented Concepts and Their UML Notation	15
3.1	Object.....	15
3.2	Classes.....	16
3.3	Attributes	17
3.4	Operations	18
3.5	Polymorphism.....	19
3.6	Interfaces	20
3.7	Components	24
3.8	Packages.....	25
3.9	Systems and Subsystems	26
3.10	Notes.....	28
3.11	Stereotypes	29
4	Relationships	31
4.1	Associations	31
4.2	Aggregation	34
4.3	Composition.....	35
4.4	Dependency	36
4.5	Generalization.....	37
4.6	Abstract Classes.....	41

4.7	Abstract Classes Versus Interfaces	44
4.8	Realization	45
5	Use Cases and Actors	47
5.1	Actors.....	47
5.2	Use Cases.....	48
5.3	Extend Relationship	50
5.4	Include Relationship	51
6	UML Diagrams	53
6.1	The Use Case Diagram	53
6.2	Use Cases Versus Functional Decomposition	55
6.3	Interaction Diagrams.....	56
6.3.1	Need for Interaction	56
6.3.2	Sequence Diagrams	58
6.3.3	Collaboration Diagrams	59
6.3.4	Sequence Versus Collaboration Diagrams	61
6.4	Activity Diagrams	61
6.5	Statechart Diagrams	63
7	Design Patterns	67
7.1	A Short History of Design Patterns	67
7.2	Fundamental Design Patterns	68
7.2.1	The Delegation Pattern	68
7.3	Creational Patterns	70
7.3.1	The Factory Method Pattern.....	70
7.3.2	The Abstract Factory Pattern	72
7.3.3	The Singleton Pattern	74
7.4	Structural Patterns	75
7.4.1	The Adaptor Pattern	75
7.4.2	The Proxy Pattern	78
7.4.3	The Iterator Pattern	79
7.4.4	The Faade Pattern.....	82
7.5	Behavioral Patterns	84
7.5.1	The State Pattern	84
7.5.2	The Strategy Pattern	86
7.5.3	The Observer Pattern	89
8	Exercises	93
8.1	Exercise 1	93
8.2	Questions	98
8.3	Exercise 2	99
8.4	Questions	103
8.5	Exercise 3	104
8.6	Questions	110
8.7	Exercise 4	112
8.8	Questions	117

8.9	Exercise 5	119
8.10	Questions	119
9	The Object Constraint Language (OCL)	121
9.1	Introduction	121
9.2	The Design by Contract Approach	122
9.2.1	Preconditions and Postconditions	123
9.2.2	Invariants	123
9.3	OCL Generalities	123
9.3.1	The Self Keyword	124
9.3.2	The Basic Types and Operators	124
9.3.3	The String Type	125
9.3.4	The Boolean Type	125
9.3.5	The Integer and Real Types	126
9.3.6	The Model Types	127
9.4	Using OCL Within UML Models	127
9.4.1	Using OCL to Navigate a UML Model	129
9.4.2	More on Collections	129
9.4.3	Examples of Invariants, Preconditions, and Postconditions	130
9.5	OCL and Java	133
10	The Model-Driven Architecture Approach	135
10.1	Model-Driven Development	135
10.2	MVC Pattern	136
10.3	MDA Approach	151
10.4	MDA and UML	154
10.5	MDA Transformations	155
10.5.1	PIM to Java Transformation	156
10.5.2	PIM to Relational Transformation	157
10.6	Modeling Behavior in MDA	158
10.7	MDA with Virtual Enterprise	162
10.7.1	Basic Concepts	162
10.7.2	The Persistence Layer	165
10.8	The Online Student System	166
10.8.1	The Use Case Model	166
10.8.2	The UML Model	167
10.8.3	Providing Objects with Behavior in Virtual Enterprise	171
10.8.4	Developing Portals in Virtual Enterprise	176

Part II Applications

11	The Kraalingen Approach to Crop Simulation	181
11.1	System Requirements	182
11.2	The Use Case Model	183
11.2.1	The Use Case Description	184

11.2.2	Basic Flow	184
11.2.3	Alternate Flow	185
11.2.4	Preconditions	185
11.2.5	Postconditions	185
11.3	The Use Case Realization	186
11.3.1	Sequence Diagram for the Use Case	186
11.3.2	Collaboration Diagram for the Use Case	188
11.4	Conceptual Models	190
11.4.1	Conceptual Model for the <i>Kraalingen</i> Approach	191
11.5	Discover Potential Classes	193
11.5.1	Boundary Classes	193
11.5.2	Control Classes	195
11.5.3	Entity Classes	196
11.6	Class Diagram for the <i>Kraalingen</i> Approach	197
11.7	Critique of the <i>Kraalingen</i> Class Diagram	202
11.7.1	Communication Boundary Control	202
11.7.2	Communication Control-Entity	204
11.7.3	Communication Entity-Entity	205
11.8	Final Class Diagram for the <i>Kraalingen</i> Approach	208
11.9	The Benefits of Using Interfaces	209
11.10	Implementation of the <i>Kraalingen</i> Model in Java	209
11.10.1	Interface IPlant	210
11.10.2	Interface ISoil	212
11.10.3	Interface IWeather	214
11.10.4	Class WeatherDataFromFile	215
11.10.5	Class WeatherDataFromStation	221
11.10.6	Interface ISimulator	224
11.10.7	Packaging the Application	227
12	The Plug and Play Architecture	229
12.1	Definition	229
12.2	Implementation	230
12.3	Reflection	231
12.4	The Plug and Play <i>SimulationController</i>	232
12.5	Testing Unit for a Class/Component	234
13	Soil Water-Balance and Irrigation-Scheduling Models: A	
	Case Study	239
13.1	Introduction	239
13.2	Conceptual Models: Examples	240
13.3	Template for Developing New Models	242
13.4	Analysis of a Water-Balance Model	244
13.5	Analysis of an Irrigation-Scheduling Model	246
13.6	The Benefits of a General Template	249

14 Distributed Models	251
14.1 Introduction	251
14.2 Common Object Request Broker Architecture	252
14.2.1 The Interface Definition Language	253
14.2.2 The Object Request Broker	254
14.2.3 Interface Adaptors	256
14.2.4 A CORBA Soil Server	256
14.2.5 A Simple CORBA Client	260
14.3 The Remote Method Invocation	261
14.3.1 An RMI Soil Server	264
14.3.2 A Simple RMI Client	266
14.4 Distributed Crop Simulation Model	267
15 MDA with Oliva Nova	273
15.1 The OO-Method	273
15.2 Conceptual Model	274
15.3 The MDA-Based Kraalingen Model	275
15.4 Providing Objects with Behavior	277
15.5 Debugging and Evaluating a Conceptual Diagram	281
15.6 Code Generation	283
16 Answers	287
16.1 Answers to Exercise 1	287
16.2 Answers to Exercise 2	288
16.3 Answers to Exercise 3	289
16.4 Answers to Exercise 4	290
16.5 Answers to Exercise 5	291
Glossary	293
References	297

Software Engineering Techniques Applied to
Agricultural Systems
An Object-Oriented and UML Approach
Papajorgji, P.J.; Pardalos, P.
2014, XVII, 301 p. 239 illus., 89 illus. in color.,
Hardcover
ISBN: 978-1-4899-7462-4