

APPLYING UML AND PATTERNS

AN INTRODUCTION TO
OBJECT-ORIENTED ANALYSIS AND DESIGN
AND ITERATIVE DEVELOPMENT

THIRD EDITION

CRAIG LARMAN



Prentice Hall PTR
Upper Saddle River, NJ 07458
www.phptr.com

TABLE OF CONTENTS

Foreword xix

Preface xxi

PART I INTRODUCTION

- 1 Object-Oriented Analysis and Design 3
 - What Will You Learn? Is it Useful? 3
 - The Most Important Learning Goal? 6
 - What is Analysis and Design? 6
 - What is Object-Oriented Analysis and Design? 7
 - A Short Example 8
 - What is the UML? 11
 - Visual Modeling is a Good Thing 14
 - History 15
 - Recommended Resources 16
- 2 Iterative, Evolutionary, and Agile 17
 - What is the UP? Are Other Methods Complementary? 18
 - What is Iterative and Evolutionary Development? 19
 - What About the Waterfall Lifecycle? 23
 - How to do Iterative and Evolutionary Analysis and Design? 25
 - What is Risk-Driven and Client-Driven Iterative Planning? 27
 - What are Agile Methods and Attitudes? 27
 - What is Agile Modeling? 30
 - What is an Agile UP? 31
 - Are There Other Critical UP Practices? 33
 - What are the UP Phases? 33
 - What are the UP Disciplines? 34
 - How to Customize the Process? The UP Development Case 37
 - You Know You Didn't Understand Iterative Development or the UP When... 38
 - History 39
 - Recommended Resources 40
- 3 Case Studies 41
 - What is and isn't Covered in the Case Studies? 41
 - Case Study Strategy: Iterative Development + Iterative Learning 43
 - Case One: The NextGen POS System 43
 - Case Two: The Monopoly Game System 44

PART II INCEPTION

- 4 Inception is Not the Requirements Phase 47
 - What is Inception? 48
 - How Long is Inception? 49
 - What Artifacts May Start in Inception? 49
 - You Know You Didn't Understand Inception When... 51
 - How Much UML During Inception? 51
- 5 Evolutionary Requirements 53
 - Definition: Requirements 54
 - Evolutionary vs. Waterfall Requirements 54
 - What are Skillful Means to Find Requirements? 56
 - What are the Types and Categories of Requirements? 56
 - How are Requirements Organized in UP Artifacts? 58
 - Does the Book Contain Examples of These Artifacts? 58
 - Recommended Resources 59
- 6 Use Cases 61

TABLE OF CONTENTS

	Example	63
	Definition: What are Actors, Scenarios, and Use Cases?	63
	Use Cases and the Use-Case Model	64
	Motivation: Why Use Cases?	64
	Definition: Are Use Cases Functional Requirements?	65
	Definition: What are Three Kinds of Actors?	66
	Notation: What are Three Common Use Case Formats?	66
	Example: Process Sale, Fully Dressed Style	67
	What do the Sections Mean?	72
	Notation: Are There Other Formats? A Two-Column Variation	78
	Guideline: Write in an Essential UI-Free Style	80
	Guideline: Write Terse Use Cases	81
	Guideline: Write Black-Box Use Cases	81
	Guideline: Take an Actor and Actor-Goal Perspective	82
	Guideline: How to Find Use Cases	82
	Guideline: What Tests Can Help Find Useful Use Cases?	87
	Applying UML: Use Case Diagrams	89
	Applying UML: Activity Diagrams	92
	Motivation: Other Benefits of Use Cases? Requirements in Context	92
	Example: Monopoly Game	93
	Process: How to Work With Use Cases in Iterative Methods?	95
	History	99
	Recommended Resources	99
7	Other Requirements	101
	How Complete are these Examples?	102
	Guideline: Should We Analyze These Thoroughly During Inception?	102
	Guideline: Should These Artifacts be at the Project Website?	103
	NextGen Example: (Partial) Supplementary Specification	104
	Commentary: Supplementary Specification	107
	NextGen Example: (Partial) Vision	109
	Commentary: Vision	111
	NextGen Example: A (Partial) Glossary	115
	Commentary: Glossary (Data Dictionary)	115
	NextGen Example: Business Rules (Domain Rules)	116
	Commentary: Domain Rules	117
	Process: Evolutionary Requirements in Iterative Methods	118
	Recommended Resources	119
	PART III ELABORATION ITERATION 1 — BASICS	
8	Iteration 1—Basics	123
	Iteration 1 Requirements and Emphasis: Core OOA/D Skills	124
	Process: Inception and Elaboration	126
	Process: Planning the Next Iteration	130
9	Domain Models	131
	Example	133
	What is a Domain Model?	134
	Motivation: Why Create a Domain Model?	137
	Guideline: How to Create a Domain Model?	139
	Guideline: How to Find Conceptual Classes?	139
	Example: Find and Draw Conceptual Classes	143
	Guideline: Agile Modeling—Sketching a Class Diagram	144
	Guideline: Agile Modeling—Maintain the Model in a Tool?	144
	Guideline: Report Objects—Include 'Receipt' in the Model?	145

TABLE OF CONTENTS

	Guideline: Think Like a Mapmaker; Use Domain Terms	145
	Guideline: How to Model the <i>Unreal</i> World?	146
	Guideline: A Common Mistake with Attributes vs. Classes	146
	Guideline: When to Model with 'Description' Classes?	147
	Associations	149
	Example: Associations in the Domain Models	156
	Attributes	158
	Example: Attributes in the Domain Models	166
	Conclusion: Is the Domain Model Correct?	168
	Process: Iterative and Evolutionary Domain Modeling	169
	Recommended Resources	170
10	System Sequence Diagrams	173
	Example: NextGen SSD	175
	What are System Sequence Diagrams?	176
	Motivation: Why Draw an SSD?	176
	Applying UML: Sequence Diagrams	177
	What is the Relationship Between SSDs and Use Cases?	177
	How to Name System Events and Operations?	178
	How to Model SSDs Involving Other External Systems?	178
	What SSD Information to Place in the Glossary?	179
	Example: Monopoly SSD	179
	Process: Iterative and Evolutionary SSDs	180
	History and Recommended Resources	180
11	Operation Contracts	181
	Example	183
	Definition: What are the Sections of a Contract?	183
	Definition: What is a System Operation?	183
	Definition: Postconditions	184
	Example: <i>enterItem</i> Postconditions	187
	Guideline: Should We Update the Domain Model?	188
	Guideline: When Are Contracts Useful?	188
	Guideline: How to Create and Write Contracts	189
	Example: NextGen POS Contracts	190
	Example: Monopoly Contracts	191
	Applying UML: Operations, Contracts, and the OCL	191
	Process: Operation Contracts Within the UP	193
	History	193
	Recommended Resources	194
12	Requirements to Design—Iteratively	195
	Iteratively Do the Right Thing, Do the Thing Right	196
	Provoking Early Change	196
	Didn't All That Analysis and Modeling Take Weeks To Do?	196
13	Logical Architecture and UML Package Diagrams	197
	Example	199
	What is the Logical Architecture? And Layers?	199
	What Layers are the Focus in the Case Studies?	200
	What is Software Architecture?	200
	Applying UML: Package Diagrams	201
	Guideline: Design with Layers	202
	Guideline: The Model-View Separation Principle	209
	What's the Connection Between SSDs, System Operations, and Layers?	210
	Example: NextGen Logical Architecture and Package Diagram	211
	Example: Monopoly Logical Architecture?	212

TABLE OF CONTENTS

	Recommended Resources	212
14	On to Object Design	213
	Agile Modeling and Lightweight UML Drawing	214
	UML CASE Tools	215
	How Much Time Spent Drawing UML Before Coding?	215
	Designing Objects: What are Static and Dynamic Modeling?	216
	The Importance of Object Design Skill over UML Notation Skill	217
	Other Object Design Techniques: CRC Cards	218
15	UML Interaction Diagrams	221
	Sequence and Communication Diagrams	222
	Novice UML Modelers Don't Pay Enough Attention to Interaction Diagrams!	225
	Common UML Interaction Diagram Notation	226
	Basic Sequence Diagram Notation	227
	Basic Communication Diagram Notation	240
16	UML Class Diagrams	249
	Applying UML: Common Class Diagram Notation	250
	Definition: Design Class Diagram	251
	Definition: Classifier	251
	Ways to Show UML Attributes: Attribute Text and Association Lines	252
	Note Symbols: Notes, Comments, Constraints, and Method Bodies	256
	Operations and Methods	256
	Keywords	258
	Stereotypes, Profiles, and Tags	259
	UML Properties and Property Strings	260
	Generalization, Abstract Classes, Abstract Operations	260
	Dependency	260
	Interfaces	263
	Composition Over Aggregation	264
	Constraints	265
	Qualified Association	265
	Association Class	266
	Singleton Classes	266
	Template Classes and Interfaces	267
	User-Defined Compartments	268
	Active Class	269
	What's the Relationship Between Interaction and Class Diagrams?	269
17	GRASP: Designing Objects with Responsibilities	271
	UML versus Design Principles	272
	Object Design: Example Inputs, Activities, and Outputs	272
	Responsibilities and Responsibility-Driven Design	276
	GRASP: A Methodical Approach to Basic OO Design	277
	What's the Connection Between Responsibilities, GRASP, and UML Diagrams?	277
	What are Patterns?	278
	Where are We Now?	281
	A Short Example of Object Design with GRASP	281
	Applying GRASP to Object Design	291
	Creator	291
	Information Expert (or Expert)	294
	Low Coupling	299
	Controller	302
	High Cohesion	314
	Recommended Resources	319
18	Object Design Examples with GRASP	321

TABLE OF CONTENTS

	What is a Use Case Realization?	322
	Artifact Comments	324
	What's Next?	327
	Use Case Realizations for the NextGen Iteration	327
	Use Case Realizations for the Monopoly Iteration	349
	Process: Iterative and Evolutionary Object Design	360
	Summary	362
19	Designing for Visibility	363
	Visibility Between Objects	363
	What is Visibility?	364
20	Mapping Designs to Code	369
	Programming and Iterative, Evolutionary Development	370
	Mapping Designs to Code	371
	Creating Class Definitions from DCDs	371
	Creating Methods from Interaction Diagrams	372
	Collection Classes in Code	374
	Exceptions and Error Handling	374
	Defining the Sale.makeLineItem Method	375
	Order of Implementation	375
	Test-Driven or Test-First Development	376
	Summary of Mapping Designs to Code	376
	Introduction to the NextGen POS Program Solution	377
	Introduction to the Monopoly Program Solution	380
21	Test-Driven Development and Refactoring	385
	Test-Driven Development	386
	Refactoring	389
	Recommended Resources	393
PART IV ELABORATION ITERATION 2 — MORE PATTERNS		
22	UML Tools and UML as Blueprint	397
	Forward, Reverse, and Round-Trip Engineering	398
	What is a Common Report of Valuable Features?	398
	What to Look For in a Tool?	399
	If Sketching UML, How to Update the Diagrams After Coding?	399
	Recommended Resources	400
23	Quick Analysis Update	401
	Case Study: NextGen POS	401
	Case Study: Monopoly	403
24	Iteration 2—More Patterns	407
	From Iteration 1 to 2	408
	Iteration-2 Requirements and Emphasis: Object Design and Patterns	409
25	GRASP: More Objects with Responsibilities	413
	Polymorphism	414
	Pure Fabrication	421
	Indirection	426
	Protected Variations	427
26	Applying GoF Design Patterns	435
	Adapter (GoF)	436
	Some GRASP Principles as a Generalization of Other Patterns	438
	"Analysis" Discoveries During Design: Domain Model	440
	Factory	440
	Singleton (GoF)	442
	Conclusion of the External Services with Varying Interfaces Problem	446
	Strategy (GoF)	447

TABLE OF CONTENTS

	Composite (GoF) and Other Design Principles	452
	Facade (GoF)	461
	Observer/Publish-Subscribe/Delegation Event Model (GoF)	463
	Conclusion	471
	Recommended Resources	471
PART V ELABORATION ITERATION 3 — INTERMEDIATE TOPICS		
27	Iteration 3—Intermediate Topics	475
	NextGen POS	476
	Monopoly	476
28	UML Activity Diagrams and Modeling	477
	Example	477
	How to Apply Activity Diagrams?	478
	More UML Activity Diagram Notation	481
	Guidelines	482
	Example: NextGen Activity Diagram	483
	Process: Activity Diagrams in the UP	483
	Background	484
29	UML State Machine Diagrams and Modeling	485
	Example	486
	Definitions: Events, States, and Transitions	486
	How to Apply State Machine Diagrams?	487
	More UML State Machine Diagram Notation	489
	Example: UI Navigation Modeling with State Machines	490
	Example: NextGen Use Case State Machine Diagram	491
	Process: State Machine Diagrams in the UP	492
	Recommended Resources	492
30	Relating Use Cases	493
	The include Relationship	494
	Terminology: Concrete, Abstract, Base, and Addition Use Cases	497
	The extend Relationship	497
	The generalize Relationship	499
	Use Case Diagrams	499
31	More SSDs and Contracts	501
	NextGen POS	501
32	Domain Model Refinement	507
	New Concepts for the NextGen Domain Model	508
	Generalization	509
	Defining Conceptual Superclasses and Subclasses	511
	When to Define a Conceptual Subclass?	514
	When to Define a Conceptual Superclass?	516
	NextGen POS Conceptual Class Hierarchies	516
	Abstract Conceptual Classes	519
	Modeling Changing States	521
	Class Hierarchies and Inheritance in Software	522
	Association Classes	522
	Aggregation and Composition	525
	Time Intervals and Product Prices—Fixing an Iteration 1 "Error"	527
	Association Role Names	528
	Roles as Concepts versus Roles in Associations	529
	Derived Elements	530
	Qualified Associations	531
	Reflexive Associations	532
	Using Packages to Organize the Domain Model	532

TABLE OF CONTENTS

	Example: Monopoly Domain Model Refinements	538
33	Architectural Analysis	541
	Process: When Do We Start Architectural Analysis?	542
	Definition: Variation and Evolution Points	542
	Architectural Analysis	543
	Common Steps in Architectural Analysis	544
	The Science: Identification and Analysis of Architectural Factors	545
	Example: Partial NextGen POS Architectural Factor Table	548
	The Art: Resolution of Architectural Factors	549
	Summary of Themes in Architectural Analysis	556
	Process: Iterative Architecture in the UP	556
	Recommended Resources	558
34	Logical Architecture Refinement	559
	Example: NextGen Logical Architecture	560
	Collaborations with the Layers Pattern	565
	Other Layer Pattern Issues	571
	Model-View Separation and "Upward" Communication	576
	Recommended Resources	577
35	More Object Design with GoF Patterns	579
	Example: NextGen POS	580
	Failover to Local Services; Performance with Local Caching	580
	Handling Failure	585
	Failover to Local Services with a Proxy (GoF)	591
	Designing for Non-Functional or Quality Requirements	595
	Accessing External Physical Devices with Adapters	595
	Abstract Factory (GoF) for Families of Related Objects	597
	Handling Payments with Polymorphism and Do It Myself	600
	Example: Monopoly	607
	Conclusion	610
36	Package Design	613
	Package Organization Guidelines	614
	Recommended Resources	620
37	UML Deployment and Component Diagrams	621
	Deployment Diagrams	621
	Component Diagrams	623
38	Designing a Persistence Framework with Patterns	625
	The Problem: Persistent Objects	626
	The Solution: A Persistence Service from a Persistence Framework	627
	Frameworks	627
	Requirements for the Persistence Service and Framework	628
	Key Ideas	628
	Pattern: Representing Objects as Tables	629
	UML Data Modeling Profile	629
	Pattern: Object Identifier	630
	Accessing a Persistence Service with a Facade	631
	Mapping Objects: Database Mapper or Database Broker Pattern	632
	Framework Design with the Template Method Pattern	634
	Materialization with the Template Method Pattern	634
	Configuring Mappers with a MapperFactory	640
	Pattern: Cache Management	641
	Consolidating and Hiding SQL Statements in One Class	641
	Transactional States and the State Pattern	642
	Designing a Transaction with the Command Pattern	645

	Lazy Materialization with a Virtual Proxy	647
	How to Represent Relationships in Tables	651
	PersistentObject Superclass and Separation of Concerns	652
	Unresolved Issues	652
39	Documenting Architecture: UML & the N+1 View Model	655
	The SAD and Its Architectural Views	656
	Notation: The Structure of a SAD	659
	Example: A NextGen POS SAD	660
	Example: A Jakarta Struts SAD	665
	Process: Iterative Architectural Documentation	669
	Recommended Resources	669
PART VI SPECIAL TOPICS		
40	More on Iterative Development and Agile Project Management	673
	How to Plan an Iteration?	674
	Adaptive versus Predictive Planning	674
	Phase and Iteration Plans	676
	How to Plan Iterations with Use Cases and Scenarios?	676
	The (Invalidity of Early Estimates	678
	Organizing Project Artifacts	680
	You Know You Didn't Understand Iterative Planning When...	681
	Recommended Resources	681
	Bibliography	683
	Glossary	689
	Index	695