
CONTENTS

<i>Foreword</i>	xvii
<i>Preface</i>	xix
<i>Acknowledgments</i>	xxix
<i>Part I</i>	
Putting the Domain Model to Work	1
Chapter 1: <i>Crunching Knowledge</i>	7
Ingredients of Effective Modeling	12
Knowledge Crunching	13
Continuous Learning	15
Knowledge-Rich Design	17
Deep Models	20
Chapter 2: <i>Communication and the Use of Language</i>	23
UBIQUITOUS LANGUAGE	24
Modeling Out Loud	30
One Team, One Language	32
Documents and Diagrams	35
<i>Written Design Documents</i>	37
<i>Executable Bedrock</i>	40
Explanatory Models	41
Chapter 3: <i>Binding Model and Implementation</i>	45
MODEL-DRIVEN DESIGN	47
Modeling Paradigms and Tool Support	50
Letting the Bones Show: Why Models Matter to Users	57
HANDS-ON MODELERS	60

<i>Part II</i>	
The Building Blocks of a Model-Driven Design	63
Chapter 4: <i>Isolating the Domain</i>	67
LAYERED ARCHITECTURE	68
<i>Relating the Layers</i>	72
<i>Architectural Frameworks</i>	74
The Domain Layer Is Where the Model Lives	75
THE SMART UI “ANTI-PATTERN”	76
Other Kinds of Isolation	79
Chapter 5: <i>A Model Expressed in Software</i>	81
Associations	82
ENTITIES (A.K.A. REFERENCE OBJECTS)	89
<i>Modeling ENTITIES</i>	93
<i>Designing the Identity Operation</i>	94
VALUE OBJECTS	97
<i>Designing VALUE OBJECTS</i>	99
<i>Designing Associations That Involve VALUE OBJECTS</i>	102
SERVICES	104
<i>SERVICES and the Isolated Domain Layer</i>	106
<i>Granularity</i>	108
<i>Access to SERVICES</i>	108
MODULES (A.K.A. PACKAGES)	109
<i>Agile MODULES</i>	111
<i>The Pitfalls of Infrastructure-Driven Packaging</i>	112
Modeling Paradigms	116
<i>Why the Object Paradigm Predominates</i>	116
<i>Nonobjects in an Object World</i>	119
<i>Sticking with MODEL-DRIVEN DESIGN When</i>	
<i>Mixing Paradigms</i>	120
Chapter 6: <i>The Life Cycle of a Domain Object</i>	123
AGGREGATES	125
FACTORIES	136
<i>Choosing FACTORIES and Their Sites</i>	139
<i>When a Constructor Is All You Need</i>	141
<i>Designing the Interface</i>	143

<i>Where Does Invariant Logic Go?</i>	144
<i>ENTITY FACTORIES Versus VALUE OBJECT FACTORIES</i>	144
<i>Reconstituting Stored Objects</i>	145
REPOSITORIES	147
<i>Querying a REPOSITORY</i>	152
<i>Client Code Ignores REPOSITORY Implementation;</i> <i>Developers Do Not</i>	154
<i>Implementing a REPOSITORY</i>	155
<i>Working Within Your Frameworks</i>	156
<i>The Relationship with FACTORIES</i>	157
Designing Objects for Relational Databases	159
Chapter 7: <i>Using the Language: An Extended Example</i>	163
Introducing the Cargo Shipping System	163
Isolating the Domain: Introducing the Applications	166
Distinguishing ENTITIES and VALUE OBJECTS	167
<i>Role and Other Attributes</i>	168
Designing Associations in the Shipping Domain	169
AGGREGATE Boundaries	170
Selecting REPOSITORIES	172
Walking Through Scenarios	173
<i>Sample Application Feature: Changing the Destination</i> <i>of a Cargo</i>	173
<i>Sample Application Feature: Repeat Business</i>	173
Object Creation	174
<i>FACTORIES and Constructors for Cargo</i>	174
<i>Adding a Handling Event</i>	175
Pause for Refactoring: An Alternative Design of the Cargo AGGREGATE	177
MODULES in the Shipping Model	179
Introducing a New Feature: Allocation Checking	181
<i>Connecting the Two Systems</i>	182
<i>Enhancing the Model: Segmenting the Business</i>	183
<i>Performance Tuning</i>	185
A Final Look	186

<i>Part III</i>	
Refactoring Toward Deeper Insight	187
Chapter 8: <i>Breakthrough</i>	193
Story of a Breakthrough	194
<i>A Decent Model, and Yet . . .</i>	194
<i>The Breakthrough</i>	196
<i>A Deeper Model</i>	198
<i>A Sobering Decision</i>	199
<i>The Payoff</i>	200
Opportunities	201
Focus on Basics	201
Epilogue: A Cascade of New Insights	202
Chapter 9: <i>Making Implicit Concepts Explicit</i>	205
Digging Out Concepts	206
<i>Listen to Language</i>	206
<i>Scrutinize Awkwardness</i>	210
<i>Contemplate Contradictions</i>	216
<i>Read the Book</i>	217
<i>Try, Try Again</i>	219
How to Model Less Obvious Kinds of Concepts	219
<i>Explicit Constraints</i>	220
<i>Processes as Domain Objects</i>	222
SPECIFICATION	224
<i>Applying and Implementing SPECIFICATION</i>	227
Chapter 10: <i>Supple Design</i>	243
INTENTION-REVEALING INTERFACES	246
SIDE-EFFECT-FREE FUNCTIONS	250
ASSERTIONS	255
CONCEPTUAL CONTOURS	260
STANDALONE CLASSES	265
CLOSURE OF OPERATIONS	268
Declarative Design	270
<i>Domain-Specific Languages</i>	272
A Declarative Style of Design	273
<i>Extending SPECIFICATIONS in a Declarative Style</i>	273
Angles of Attack	282

<i>Carve Off Subdomains</i>	283
<i>Draw on Established Formalisms, When You Can</i>	283
Chapter 11: <i>Applying Analysis Patterns</i>	293
Chapter 12: <i>Relating Design Patterns to the Model</i>	309
STRATEGY (A.K.A. POLICY)	311
COMPOSITE	315
Why Not FLYWEIGHT?	320
Chapter 13: <i>Refactoring Toward Deeper Insight</i>	321
Initiation	321
Exploration Teams	322
Prior Art	323
A Design for Developers	324
Timing	324
Crisis as Opportunity	325
<i>Part IV</i>	
Strategic Design	327
Chapter 14: <i>Maintaining Model Integrity</i>	331
BOUNDED CONTEXT	335
<i>Recognizing Splinters Within a BOUNDED CONTEXT</i>	339
CONTINUOUS INTEGRATION	341
CONTEXT MAP	344
<i>Testing at the CONTEXT Boundaries</i>	351
<i>Organizing and Documenting CONTEXT MAPS</i>	351
Relationships Between BOUNDED CONTEXTS	352
SHARED KERNEL	354
CUSTOMER/SUPPLIER DEVELOPMENT TEAMS	356
CONFORMIST	361
ANTICORRUPTION LAYER	364
<i>Designing the Interface of the ANTICORRUPTION LAYER</i>	366
<i>Implementing the ANTICORRUPTION LAYER</i>	366
<i>A Cautionary Tale</i>	370
SEPARATE WAYS	371
OPEN HOST SERVICE	374
PUBLISHED LANGUAGE	375

Unifying an Elephant	378
Choosing Your Model Context Strategy	381
<i>Team Decision or Higher</i>	382
<i>Putting Ourselves in Context</i>	382
<i>Transforming Boundaries</i>	382
<i>Accepting That Which We Cannot Change: Delineating the External Systems</i>	383
<i>Relationships with the External Systems</i>	384
<i>The System Under Design</i>	385
<i>Catering to Special Needs with Distinct Models</i>	386
<i>Deployment</i>	387
<i>The Trade-off</i>	388
<i>When Your Project Is Already Under Way</i>	388
Transformations	389
<i>Merging CONTEXTS: SEPARATE WAYS → SHARED KERNEL</i>	389
<i>Merging CONTEXTS: SHARED KERNEL → CONTINUOUS INTEGRATION</i>	391
<i>Phasing Out a Legacy System</i>	393
<i>OPEN HOST SERVICE → PUBLISHED LANGUAGE</i>	394
Chapter 15: Distillation	397
CORE DOMAIN	400
<i>Choosing the CORE</i>	402
<i>Who Does the Work?</i>	403
An Escalation of Distillations	404
GENERIC SUBDOMAINS	406
<i>Generic Doesn't Mean Reusable</i>	412
<i>Project Risk Management</i>	413
DOMAIN VISION STATEMENT	415
HIGHLIGHTED CORE	417
<i>The Distillation Document</i>	418
<i>The Flagged CORE</i>	419
<i>The Distillation Document as Process Tool</i>	420
COHESIVE MECHANISMS	422
<i>GENERIC SUBDOMAIN Versus COHESIVE MECHANISM</i>	424
<i>When a MECHANISM Is Part of the CORE DOMAIN</i>	425
Distilling to a Declarative Style	426
SEGREGATED CORE	428

<i>The Costs of Creating a SEGREGATED CORE</i>	429
<i>Evolving Team Decision</i>	430
ABSTRACT CORE	435
Deep Models Distill	436
Choosing Refactoring Targets	437
Chapter 16: <i>Large-Scale Structure</i>	439
EVOLVING ORDER	444
SYSTEM METAPHOR	447
<i>The “Naive Metaphor” and Why We Don’t Need It</i>	448
RESPONSIBILITY LAYERS	450
<i>Choosing Appropriate Layers</i>	460
KNOWLEDGE LEVEL	465
PLUGGABLE COMPONENT FRAMEWORK	475
How Restrictive Should a Structure Be?	480
Refactoring Toward a Fitting Structure	481
<i>Minimalism</i>	481
<i>Communication and Self-Discipline</i>	482
<i>Restructuring Yields Supple Design</i>	482
<i>Distillation Lightens the Load</i>	483
Chapter 17: <i>Bringing the Strategy Together</i>	485
Combining Large-Scale Structures and BOUNDED CONTEXTS	485
Combining Large-Scale Structures and Distillation	488
Assessment First	490
Who Sets the Strategy?	490
<i>Emergent Structure from Application Development</i>	491
<i>A Customer-Focused Architecture Team</i>	492
Six Essentials for Strategic Design Decision Making	492
<i>The Same Goes for the Technical Frameworks</i>	495
<i>Beware the Master Plan</i>	496
Conclusion	499
<i>Appendix: The Use of Patterns in This Book</i>	507
<i>Glossary</i>	511
<i>References</i>	515
<i>Photo Credits</i>	517
<i>Index</i>	519