

# Contents

## Part I Introduction

<b>1</b>	<b>Enterprise Modelling</b>	<b>3</b>
1.1	Enterprise Engineering	3
1.2	Enterprise Architecture	5
1.3	How to Develop Enterprise Models	12
1.3.1	What Exactly Is Modelling?	12
1.3.2	Which Aspects to Include into Models?	15
1.3.3	From Which Perspective Should Models Be Made?	17
1.3.4	Where to Start, How to Go On and When to Stop?	20
1.4	Modelling and Then What?	20
1.4.1	Using Models for Software Development	21
1.4.2	Following a Model-Driven Engineering Approach	23
1.5	Summary	29
<b>2</b>	<b>From Demand to Supply: Layers and Model Quality</b>	<b>31</b>
2.1	A Layered System Architecture Both for Requirements and System Architecture	32
2.1.1	Enterprise, Information Services and Business Process Layer	32
2.1.2	Layers Versus Requirements Gathering and Engineering	35
2.2	Formal Verification of Models	40
2.2.1	Model Quality	40
2.2.2	Modelling Language	44
2.2.3	Tool Support for Quality Control	46
2.2.4	Quality Checking in MERODE and JMERMAID	48
2.3	Summary	49
<b>3</b>	<b>Overview of MERODE</b>	<b>51</b>
3.1	The Modelling Phases	51
3.2	Domain Modelling	52

3.2.1	Business Object Types . . . . .	53
3.2.2	Business Event Types . . . . .	54
3.2.3	A Small Library Example . . . . .	59
3.3	Information System Service Modelling . . . . .	70
3.4	Business Process Modelling . . . . .	72
3.5	Implementation . . . . .	74
3.6	A Note to the Reader . . . . .	75

## Part II Domain Modelling Techniques

<b>4</b>	<b>The Existence-Dependency Graph . . . . .</b>	<b>79</b>
4.1	Why Existence Dependency? . . . . .	79
4.1.1	Frozen Versus Modifiable Association Ends in UML . . . . .	80
4.1.2	Association, AssociationClass or Class? . . . . .	81
4.1.3	Improved Consistency Checking . . . . .	82
4.1.4	Improved Possibilities for Transformation to Code . . . . .	83
4.2	The Existence-Dependency Graph . . . . .	83
4.2.1	Defining Existence Dependency . . . . .	83
4.2.2	The Existence-Dependency Graph . . . . .	84
4.2.3	Multiplicity of Existence Dependency . . . . .	85
4.2.4	Graphical Representation . . . . .	86
4.2.5	Life Cycle Implications of Existence Dependency . . . . .	88
4.3	How to Create an Existence-Dependency Graph . . . . .	94
4.3.1	UML Class . . . . .	95
4.3.2	Binary Association . . . . .	95
4.3.3	Unary Association . . . . .	97
4.3.4	N-ary Associations . . . . .	98
4.3.5	Aggregation . . . . .	98
4.3.6	Describing Versus Shaping the Real World . . . . .	101
4.4	Some Final Considerations . . . . .	103
4.5	Formal Definition of the Existence-Dependency Graph . . . . .	103
4.6	Meta-Model . . . . .	104
<b>5</b>	<b>Object Interaction . . . . .</b>	<b>107</b>
5.1	What Is a Business Event? . . . . .	107
5.2	The Object-Event Table . . . . .	112
5.2.1	Definition . . . . .	112
5.2.2	Graphical Representation . . . . .	113
5.3	Existence-Dependency Graph Versus Object-Event Table . . . . .	114
5.3.1	Propagation Rule . . . . .	114
5.3.2	Type of Involvement Rule . . . . .	117
5.3.3	Contract Rule . . . . .	120
5.4	Multiple Propagation . . . . .	122

5.5	Formal Definition of the Object-Event Table . . . . .	123
5.6	Meta-Model . . . . .	125
<b>6</b>	<b>Object and System Behaviour . . . . .</b>	<b>127</b>
6.1	Object Behaviour Modelling . . . . .	127
6.1.1	Finite-State Machines . . . . .	127
6.1.2	Basic Correctness Checks . . . . .	130
6.1.3	Stratification of Finite-State Machines . . . . .	131
6.1.4	Exploiting Parallelism . . . . .	132
6.2	Global System Behaviour . . . . .	134
6.3	Consistency Checking . . . . .	136
6.3.1	Consistency Checking with the OET . . . . .	137
6.3.2	Consistency Checking with the EDG . . . . .	138
6.3.3	Checking Global Behaviour . . . . .	141
6.4	Formal Definition of Interaction by Joint Participation to Business Events . . . . .	144
6.5	Meta-model . . . . .	147
<b>7</b>	<b>Attributes and Constraints . . . . .</b>	<b>149</b>
7.1	Defining Attributes . . . . .	149
7.1.1	What Are Attributes? . . . . .	149
7.1.2	The Difference Between Object Types and Attributes . . . . .	150
7.1.3	Data Types . . . . .	153
7.2	Defining Constraints . . . . .	154
7.2.1	Uniqueness Constraints . . . . .	155
7.2.2	Attribute Constraints . . . . .	156
7.2.3	Method Constraints . . . . .	156
7.2.4	Referential Integrity and Sequence Constraints . . . . .	157
7.3	Defining Methods . . . . .	161
7.4	Aliases . . . . .	161
7.5	Meta-model . . . . .	168
<b>8</b>	<b>Inheritance . . . . .</b>	<b>171</b>
8.1	Inheritance in the Class Diagram . . . . .	171
8.1.1	Definition of Generalisation/Specialisation . . . . .	171
8.1.2	Inheritance . . . . .	176
8.2	Behavioural Aspects of Inheritance . . . . .	177
8.2.1	Specialisation of Event Types . . . . .	177
8.2.2	Propagating Along Inherited Dependencies . . . . .	181
8.2.3	More Examples . . . . .	185
8.3	On the Use of Generalisation/Specialisation Hierarchies and Roles . . . . .	191
8.3.1	Attribute-Defined Subclass . . . . .	192
8.3.2	Existence-Defined Subclass . . . . .	193

8.3.3	State-Defined Subclass . . . . .	196
8.3.4	Guidelines for Using Roles and Generalisation/ Specialisation . . . . .	197
8.4	Formal Definition of Inheritance . . . . .	200
8.5	Meta-model . . . . .	200

### **Part III The Information System Layer and the Business Process Layer**

<b>9</b>	<b>The Information System Service Layer . . . . .</b>	<b>205</b>
9.1	Information Objects . . . . .	206
9.2	Information System Services . . . . .	207
9.2.1	What Is an Information System Service? . . . . .	207
9.2.2	Output Services . . . . .	208
9.2.3	Input Services . . . . .	209
9.2.4	Complex Services . . . . .	211
9.3	Refining the Architecture of the Information System Layer . . . . .	215
9.3.1	The Event Handling Layer . . . . .	215
9.3.2	Consistent Event Types and Transactions . . . . .	215
9.3.3	Cross-Cutting Concerns . . . . .	218
9.4	Case Study: The JMermaid Architecture . . . . .	219
9.4.1	Information Object Types . . . . .	219
9.4.2	Event Handling . . . . .	220
9.5	Meta-model . . . . .	222
<b>10</b>	<b>Bridging Business Process Modelling and Domain Modelling . . . . .</b>	<b>223</b>
10.1	Business Process Modelling . . . . .	223
10.2	Linking Business Processes to the Enterprise Layer and the IS Service Layer . . . . .	226
10.3	Distributing Constraints Across Layers . . . . .	230
10.3.1	General Principles . . . . .	230
10.3.2	An Order Handling Example . . . . .	233
10.3.3	Cross-Layer Consistency . . . . .	238
10.4	Meta-model . . . . .	240

### **Part IV Model Transformation**

<b>11</b>	<b>Model Transformation . . . . .</b>	<b>245</b>
11.1	Model-Driven Development . . . . .	245
11.2	Transformation Rule Examples . . . . .	247
11.2.1	General Implementation Architecture . . . . .	247
11.2.2	The Enterprise Layer . . . . .	248
11.2.3	The Event Handling Layer . . . . .	255
11.3	Transformation Technology Example . . . . .	256
11.3.1	The mxp File . . . . .	256
11.3.2	Template-Based Approach to Transformation . . . . .	258

<b>12 Application and Component Integration . . . . .</b>	<b>261</b>
12.1 Business Event-Based Coordination . . . . .	261
12.2 The Component-Event Table . . . . .	264
12.3 Architectures for Event Handlers . . . . .	265
12.4 Contract Management Capabilities . . . . .	267
12.5 Benefits of Event-Based Coordination . . . . .	270
<b>References . . . . .</b>	<b>271</b>
<b>Index . . . . .</b>	<b>277</b>

<http://www.springer.com/978-3-319-10144-6>

Enterprise Information Systems Engineering

The MERODE Approach

Snoeck, M.

2014, XX, 280 p. 178 illus., 27 illus. in color., Hardcover

ISBN: 978-3-319-10144-6