Requirements Engineering

From System Goals to UML Models to Software Specifications

Axel van Lamsweerde

WILEY
A John Wiley and Sons, Ltd., Publication

Contents

Foreword **xvii**

Preface **xxi**

Part I Fundamentals of Requirements Engineering **1**

1 Setting the Scene **3**

1.1 What is requirements engineering? 3

1.1.1 The problem world and the machine solution 4

1.1.2 Introducing our running case studies 6

1.1.3 The WHY, WHAT and WHO dimensions of requirements engineering 12

1.1.4 Types of statements involved in requirements engineering 17

1.1.5 Categories of requirements 23

1.1.6 The requirements lifecycle: Processes, actors and products 30

1.1.7 Target qualities and defects to avoid 35

1.1.8 Types of software projects 40

1.1.9 Requirements in the software lifecycle 42

1.1.10 The relationship of requirements engineering to other disciplines 45

1.2 Why engineer requirements? 47

1.2.1 Facts, data and citations about the requirements problem 47

1.2.2 The role and stakes of requirements engineering 51

1.3 Obstacles to good requirements engineering practice 52

1.4 Agile development processes and requirements engineering 53

Summary 55

Notes and Further Reading 56

Exercises 58

2 Domain Understanding and Requirements Elicitation 61
2.1 Identifying stakeholders and interacting with them 62
2.2 Artefact-driven elicitation techniques 64
2.2.1 Background study 64
2.2.2 Data collection 65
2.2.3 Questionnaires 65
2.2.4 Repertory grids and card sorts for concept-driven acquisition 66
2.2.5 Storyboards and scenarios for problem world exploration 67
2.2.6 Mock-ups and prototypes for early feedback 70
2.2.7 Knowledge reuse 72
2.3 Stakeholder-driven elicitation techniques 76
2.3.1 Interviews 77
2.3.2 Observation and ethnographic studies 79
2.3.3 Group sessions 80
2.4 Conclusion 81
Summary 82
Notes and Further Reading 84
Exercises 85

3 Requirements Evaluation 87
3.1 Inconsistency management 88
3.1.1 Types of inconsistency 88
3.1.2 Handling inconsistencies 89
3.1.3 Managing conflicts: A systematic process 90
3.2 Risk analysis 93
3.2.1 Types of risk 94
3.2.2 Risk management 95
3.2.3 Risk documentation 101
3.2.4 Integrating risk management in the requirements lifecycle 102
3.3 Evaluating alternative options for decision making 105
3.4 Requirements prioritization 108
3.5 Conclusion 112
Summary 113
Notes and Further Reading 114
Exercises 116

4 Requirements Specification and Documentation 119
4.1 Free documentation in unrestricted natural language 120
4.2 Disciplined documentation in structured natural language 121
4.2.1 Local rules on writing statements 121
4.2.2 Global rules on organizing the requirements document 124

4.3 Use of diagrammatic notations 127
4.3.1 System scope: context, problem and frame diagrams 127
4.3.2 Conceptual structures: entity–relationship diagrams 130
4.3.3 Activities and data: SADT diagrams 133
4.3.4 Information flows: dataflow diagrams 134
4.3.5 System operations: use case diagrams 136
4.3.6 Interaction scenarios: event trace diagrams 136
4.3.7 System behaviours: state machine diagrams 138
4.3.8 Stimuli and responses: R-net diagrams 142
4.3.9 Integrating multiple system views and multiview specification in UML 142
4.3.10 Diagrammatic notations: Strengths and limitations 144
4.4 Formal specification 145
4.4.1 Logic as a basis for formalizing statements 146
4.4.2 History-based specification 151
4.4.3 State-based specification 155
4.4.4 Event-based specification 163
4.4.5 Algebraic specification 167
4.4.6 Other specification paradigms 172
4.4.7 Formal specification: strengths and limitations 173
4.5 Conclusion 174
Summary 176
Notes and Further Reading 179
Exercises 183

5 Requirements Quality Assurance 187
5.1 Requirements inspections and reviews 188
5.1.1 The requirements inspection process 188
5.1.2 Inspection guidelines 190
5.1.3 Requirements inspection checklists 191
5.1.4 Conclusion 195
5.2 Queries on a requirements database 196
5.3 Requirements validation by specification animation 198
5.3.1 Extracting an executable model from the specification 199
5.3.2 Simulating the model 199
5.3.3 Visualizing the simulation 200
5.3.4 Conclusion 200
5.4 Requirements verification through formal checks 202
5.4.1 Language checks 202
5.4.2 Dedicated consistency and completeness checks 203
5.4.3 Model checking 205
5.4.4 Theorem proving 208

5.5 Conclusion 211
Summary 213
Notes and Further Reading 214
Exercises 217

6 Requirements Evolution 219
6.1 The time–space dimensions of evolution: Revisions and variants 220
6.2 Change anticipation 223
6.3 Traceability management for evolution support 225
6.3.1 Traceability links 226
6.3.2 The traceability management process, its benefits and cost 233
6.3.3 Traceability management techniques 237
6.3.4 Determining an adequate cost–benefit trade-off for traceability management 244
6.4 Change control 246
6.4.1 Change initiation 247
6.4.2 Change evaluation and prioritization 248
6.4.3 Change consolidation 249
6.5 Runtime monitoring of requirements and assumptions for dynamic change 249
6.6 Conclusion 251
Summary 252
Notes and Further Reading 254
Exercises 256

7 Goal Orientation in Requirements Engineering 259
7.1 What are goals? 260
7.2 The granularity of goals and their relationship to requirements and assumptions 261
7.3 Goal types and categories 265
7.3.1 Types of goal: behavioural goals vs soft goals 265
7.3.2 Goal categories: Functional vs non-functional goals 269
7.4 The central role of goals in the requirements engineering process 272
7.5 Where are goals coming from? 275
7.6 The relationship of goals to other requirements-related products and processes 276
7.6.1 Goals and scenarios 276
7.6.2 Intentional and operational specifications 277
7.6.3 Goals and use cases 277
7.6.4 Goals and model-checked properties 277
7.6.5 Goal orientation and agent orientation 278
7.6.6 Goal orientation and object orientation 278
7.6.7 Goal orientation and top-down analysis 279

Summary 279

Notes and Further Reading 280

Exercises 283

Part II Building System Models for Requirements Engineering 287

8 Modelling System Objectives with Goal Diagrams 293

8.1 Goal features as model annotations 294

8.2 Goal refinement 297

8.3 Representing conflicts among goals 301

8.4 Connecting the goal model with other system views 302

8.5 Modelling alternative options 303

8.5.1 Alternative goal refinements 304

8.5.2 Alternative responsibility assignments 305

8.6 Goal diagrams as AND/OR graphs 307

8.7 Documenting goal refinements and assignments with annotations 308

8.8 Building goal models: Heuristic rules and reusable patterns 309

8.8.1 Eliciting preliminary goals 309

8.8.2 Identifying goals along refinement branches 311

8.8.3 Delimiting the scope of the goal model 316

8.8.4 Avoiding common pitfalls 317

8.8.5 Reusing refinement patterns 319

8.8.6 Reusing refinement trees associated with goal categories 326

Summary 328

Notes and Further Reading 329

Exercises 331

9 Anticipating What Could Go Wrong: Risk Analysis on Goal Models 335

9.1 Goal obstruction by obstacles 336

9.1.1 What are obstacles? 336

9.1.2 Completeness of a set of obstacles 337

9.1.3 Obstacle categories 338

9.2 Modelling obstacles 339

9.2.1 Obstacle diagrams 339

9.2.2 Conditions on obstacle refinement 341

9.2.3 Bottom-up propagation of obstructions in goal AND-refinements 342

9.2.4 Annotating obstacle diagrams 343

9.3 Obstacle analysis for a more robust goal model 344

9.3.1 Identifying obstacles 344

9.3.2 Evaluating obstacles 349

9.3.3 Resolving obstacles in a modified goal model 349

Summary 353

Notes and Further Reading 355
Exercises 356

10 Modelling Conceptual Objects with Class Diagrams 359
10.1 Representing domain concepts by conceptual objects 360
10.1.1 What are conceptual objects? 360
10.1.2 Object instantiation: classes and current instances 361
10.1.3 Types of conceptual object 362
10.1.4 Object models as UML class diagrams 363
10.1.5 Object features as model annotations 364
10.2 Entities 366
10.3 Associations 366
10.4 Attributes 371
10.5 Built-in associations for structuring object models 373
10.5.1 Object specialization 373
10.5.2 Object aggregation 376
10.6 More on class diagrams 377
10.6.1 Derived attributes and associations 377
10.6.2 OR-associations 378
10.6.3 Ordered associations 379
10.6.4 Associations of associations 379
10.7 Heuristic rules for building object models 380
10.7.1 Deriving pertinent and complete class diagrams from goal diagrams 380
10.7.2 Object or attribute? 384
10.7.3 Entity, association, agent or event? 384
10.7.4 Attribute of a linked object or of the linking association? 385
10.7.5 Aggregation or association? 386
10.7.6 Specializing and generalizing concepts 386
10.7.7 Avoiding common pitfalls 387
Summary 389
Notes and Further Reading 391
Exercises 392

11 Modelling System Agents and Responsibilities 395
11.1 What are agents? 396
11.2 Characterizing system agents 397
11.2.1 Basic features 397
11.2.2 Agent capabilities 397
11.2.3 Agent responsibilities and goal realizability 399
11.2.4 Agents as operation performers 401
11.2.5 Agent wishes and beliefs 402
11.2.6 Agent dependencies 403

11.3 Representing agent models 405
11.3.1 Agent diagrams and instance declarations 405
11.3.2 Context diagrams 406
11.3.3 Dependency diagrams 407
11.4 Refinement of abstract agents 408
11.5 Building agent models 411
11.5.1 Heuristics for building agent diagrams from goal models 411
11.5.2 Generating context diagrams from goal models 413
Summary 415
Notes and Further Reading 417
Exercises 418

12 Modelling System Operations 421
12.1 What are operations? 422
12.2 Characterizing system operations 425
12.2.1 Basic features 425
12.2.2 Operation signature 425
12.2.3 Domain pre- and post-conditions 426
12.2.4 Operation performer 427
12.3 Goal operationalization 427
12.3.1 Required pre-, post- and trigger conditions for goal satisfaction 427
12.3.2 Agent commitments 430
12.3.3 Goal operationalization and satisfaction arguments 432
12.4 Goals, agents, objects and operations: The semantic picture 434
12.5 Representing operation models 435
12.5.1 Operationalization diagrams 435
12.5.2 UML use case diagrams 435
12.6 Building operation models 437
12.6.1 Heuristics for building operationalization diagrams 437
12.6.2 Generating use case diagrams from operationalization diagrams 442
Summary 442
Notes and Further Reading 444
Exercises 445

13 Modelling System Behaviours 449
13.1 Modelling instance behaviours 450
13.1.1 Scenarios as UML sequence diagrams 450
13.1.2 Scenario refinement: Episodes and agent decomposition 452
13.2 Modelling class behaviours 454
13.2.1 State machines as UML state diagrams 455
13.2.2 State machine refinement: Sequential and concurrent sub-states 459
13.3 Building behaviour models 463
13.3.1 Elaborating relevant scenarios for good coverage 465

13.3.2 Decorating scenarios with state conditions 467
13.3.3 From scenarios to state machines 469
13.3.4 From scenarios to goals 473
13.3.5 From operationalized goals to state machines 475
Summary 477
Notes and Further Reading 480
Exercises 481

14 Integrating Multiple System Views 485
14.1 A meta-model for view integration 485
14.1.1 Overall structure of the meta-model 487
14.1.2 The goal meta-model 488
14.1.3 The object meta-model 489
14.1.4 The agent meta-model 490
14.1.5 The operation meta-model 491
14.1.6 The behaviour meta-model 492
14.2 Inter-view consistency rules 493
14.3 Grouping related view fragments into packages 496
Summary 498
Notes and Further Reading 498
Exercises 499

15 A Goal-Oriented Model-Building Method in Action 501
15.1 Modelling the system-as-is 503
15.1.1 Step 1: Build a preliminary goal model illustrated by scenarios 503
15.1.2 Step 2: Derive a preliminary object model 506
15.2 Modelling the system-to-be 507
15.2.1 Step 3: Update the goal model with new goals illustrated by scenarios 507
15.2.2 Step 4: Derive the updated object model 510
15.2.3 Step 5: Analyse obstacles, threats and conflicts 512
15.2.4 Step 6: Analyse responsibilities and build the agent model 515
15.2.5 Step 7: Make choices among alternative options 517
15.2.6 Step 8: Operationalize goals in the operation model 518
15.2.7 Step 9: Build and analyse the behaviour model 521
15.3 Handling model variants for product lines 524
Summary 528
Notes and Further Reading 529
Exercises 529

Part III Reasoning About System Models 535

16 Semi-Formal Reasoning for Model Analysis and Exploitation 537
16.1 Query-based analysis of the model database 538

16.1.1 Checking the structural consistency and completeness of the model 538
16.1.2 Generation of other views for dedicated analyses 540
16.1.3 Traceability management 540
16.1.4 Analogical model reuse 541
16.2 Semi-formal analysis of goal-oriented models 544
16.2.1 Conflict analysis 544
16.2.2 Heuristic identification of obstacles 549
16.2.3 Threat analysis: From goal models to anti-goal models 551
16.3 Reasoning about alternative options 557
16.3.1 Qualitative reasoning about alternatives 557
16.3.2 Quantitative reasoning about alternatives 560
16.4 Model-driven generation of the requirements document 562
16.5 Beyond RE: From goal-oriented requirements to software architecture 566
16.5.1 Deriving a software data architecture from the object model 567
16.5.2 Deriving an abstract dataflow architecture from the agent and operation models 568
16.5.3 Selecting an architectural style from architectural requirements 570
16.5.4 Architectural refinement from quality requirements 571
Summary 574
Notes and Further Reading 576
Exercises 578

17 Formal Specification of System Models 583
17.1 A real-time temporal logic for specifying model annotations 584
17.1.1 State assertions 584
17.1.2 Temporal assertions 585
17.1.3 Real-time temporal constructs 586
17.2 Specifying goals in the goal model 588
17.3 Specifying descriptive properties in the object model 592
17.4 Specifying operationalizations in the operation model 594
17.5 Back to the system's semantic picture 596
Summary 598
Notes and Further Reading 599
Exercises 599

18 Formal Reasoning for Specification Construction and Analysis 603
18.1 Checking goal refinements 604
18.1.1 Using a theorem prover 604
18.1.2 Formal refinement patterns 604
18.1.3 Using bounded SAT solvers 608
18.2 Deriving goal operationalizations 609
18.2.1 Using bounded SAT solvers 610

18.2.2 Formal operationalization patterns 610
18.3 Generating obstacles for risk analysis 613
18.3.1 Regressing obstructions through domain properties 614
18.3.2 Using formal obstruction patterns 617
18.4 Generating anti-goals for security analysis 618
18.4.1 Specifying security goals 618
18.4.2 Identifying security goals and initial anti-goals 620
18.4.3 Refining anti-goals 621
18.5 Formal conflict analysis 622
18.5.1 Deriving boundary conditions for conflict 623
18.5.2 Formal resolution of divergences 625
18.6 Synthesizing behaviour models for animation and model checking 627
18.6.1 Goal-driven model synthesis 628
18.6.2 Scenario-driven model synthesis 628
Summary 635
Notes and Further Reading 636
Exercises 637

Bibliography **641**

Index **669**