

Contents

1	Introduction	1
1.1	Problem Statement	1
1.2	Research Questions & Contribution	5
1.2.1	Research Questions	5
1.2.2	Contribution	6
1.3	Outline	14
2	Background	19
2.1	Requirements Engineering	19
2.1.1	Quality Requirements	20
2.1.2	Problem Frames	22
2.2	Software Architecture Concepts	24
2.2.1	Definition of Software Architecture	24
2.2.2	Difference between Architecture and Design	25
2.2.3	Architectural Patterns	27
2.2.4	Quality-specific Mechanisms and Tactics	28
2.2.5	Viewpoint Models	29
2.2.6	Architecture Description Languages vs UML	30
2.2.7	Architecture Evaluation	31
2.3	UML Profiles	32
2.3.1	UML profile for Problem Frames	33
2.3.2	Architecture Profile	37
2.3.3	Dependability Profile	39
2.3.4	MARTE Profile	40
2.4	Life-Cycle Expressions	42
2.5	Variability Modeling	42

2.6	Case Study Smart Grid	43
2.6.1	Description of Smart Grids	45
2.6.2	Functional Requirements	46
2.6.3	Security Requirements	48
2.6.4	Performance Requirements	49
3	Framework for Identifying Meta-Requirements	51
3.1	Introduction	51
3.2	Meta-Requirement Derivation	54
3.2.1	Essential Meta-Requirements	57
3.2.2	Recommended Meta-Requirements	61
3.2.3	Optional Meta-Requirements	67
3.2.4	Method Characteristics	69
3.3	The Evaluation Framework NIMSAD	70
3.3.1	Methodology Context	71
3.3.2	Methodology User	72
3.3.3	Methodology Contents	72
3.3.4	Evaluation	73
3.4	Our Proposed Evaluation Framework	73
3.5	Related Review	75
3.6	Research Method	75
3.6.1	Planning Phase	75
3.6.2	Conducting Phase	79
3.7	Results and Discussion	85
3.7.1	Description of Selected Methods	86
3.7.2	Results of the SLR	91
3.8	Comparative Evaluation	97
3.8.1	Value Assignment Schema	97
3.8.2	Framework Application	104
3.9	Threats to Validity	107
3.10	Contributions	108
4	Phase 1: Context Elicitation & Problem Analysis	111
4.1	Introduction	111
4.2	UML4PF Extension for Quality Requirements	112
4.3	Method for Problem-oriented Requirement Analysis	113
4.4	Related Work	126
4.5	Contributions	127

5	Phase 2: Architectural Pattern Selection & Application	129
5.1	Introduction	129
5.2	Artifacts and their Relations	132
5.3	External Input for the Process	134
5.3.1	Question Catalog (Questions)	134
5.3.2	Question Catalog (Indicator Questions)	136
5.3.3	Relations between Problem Frames and Questions	137
5.3.4	Benefits and Liabilities of Architectural Patterns	138
5.3.5	Architectural Pattern Catalog	140
5.4	The Pattern Selection Process	141
5.5	Application to the Case Study Smart Grid	145
5.6	Derivation of Initial Architecture	161
5.6.1	Design Decision regarding Architectural Pattern Selection	162
5.6.2	Design Decision regarding Gateway Physical Boundary	162
5.6.3	Further Iterations - Problem Diagram Splitting	163
5.6.4	Method for Deriving Initial Architecture	165
5.7	Related Work	173
5.8	Contributions	174
6	Phase 3: Domain Knowledge Analysis	175
6.1	Introduction	175
6.2	Structured Meta-Process	177
6.3	Structured Object-Process	186
6.4	Related Work	192
6.5	Contributions	194
7	Phase 4: Requirements Interaction Analysis	195
7.1	Introduction	195
7.2	Functional Requirements Interaction Detection	198
7.2.1	Sunblind Example	198
7.2.2	Method for Functional Requirements Interaction Detection	200
7.2.3	Application to the Case Study Smart Grid	209
7.3	Method for Quality Requirements Interaction Detection	210
7.4	Method for Performance Requirements Analysis	220
7.5	Method for Generating Requirement Alternatives	231
7.6	Related Work	242
7.6.1	Related work with respect to Requirements Interaction	242
7.6.2	Related work with respect to Performance Analysis	244
7.7	Contributions	245

8	Phase 5: Quality-specific Pattern Analysis	247
8.1	Introduction	247
8.2	Problem-oriented Security Patterns	249
8.2.1	UML4PF Extension for Problem-oriented Security Patterns	250
8.2.2	Structure of the Problem-oriented Security Patterns	251
8.2.3	Problem-oriented Symmetric Encryption Pattern	253
8.2.4	Problem-oriented MAC Pattern	256
8.2.5	Problem-oriented RBAC Pattern	257
8.2.6	Problem-oriented Digital Signature Pattern	260
8.2.7	Problem-oriented Asymmetric Encryption Pattern	262
8.3	Problem-oriented Performance Patterns	264
8.3.1	UML4PF Extension for Problem-oriented Performance Patterns	264
8.3.2	Structure of the Problem-oriented Performance Patterns	265
8.3.3	Problem-oriented First Things First (FTF) Pattern	269
8.3.4	Problem-oriented Flex Time (FT) Pattern	270
8.3.5	Problem-oriented Master-Worker (MW) Pattern	271
8.3.6	Problem-oriented Load Balancer (LB) Pattern	273
8.4	Discussion	274
8.5	Mapping Requirements to Quality Solutions	278
8.5.1	UML4PF Extension for Mapping Requirements to their Solution Alternatives	278
8.5.2	Problem-Solution Diagram	282
8.6	Related Work	285
8.6.1	Related work with respect to Security and Performance	285
8.6.2	Related work with respect to Variability	286
8.7	Contributions	288
9	Phase 6: Quality-specific Pattern Selection & Application	289
9.1	Introduction	289
9.2	Method for Selecting & Applying Quality-specific Patterns	290
9.3	Contributions	327
10	Phase 7: Software Architecture Alternatives Derivation	329
10.1	Introduction	329
10.2	Method for Deriving Implementable Architecture Alternatives	330
10.3	Related Work	348
10.4	Contributions	351

11	Phase 8: Software Architecture Alternatives Evaluation	353
11.1	Introduction	353
11.2	Identification of Software Architecture Evaluation Methods	354
11.2.1	Research Method	355
11.2.2	Results	356
11.3	Comparative Framework for Software Architecture Evaluation Methods	357
11.4	Selection of Software Architecture Evaluation Methods	361
11.4.1	Requirements on the Evaluation Method	361
11.4.2	Application of the Comparative Framework	362
11.5	Evaluation of Architecture Alternatives using ATAM	367
11.5.1	Application of ATAM to Smart Grid's Architecture Alternatives	367
11.5.2	Discussion of the results	383
11.6	Related Work	386
11.7	Contributions	387
12	Validation of the QuaDRA Framework	389
12.1	Introduction	389
12.2	Evaluation Framework	390
12.3	Value Assignment Schema	392
12.4	Comparative Evaluation of the QuaDRA Framework	392
12.4.1	Value Assignment	393
12.4.2	Comparison of QuaDRA with the State-of-the-Art Methods	400
12.5	Contributions	402
13	Extending Problem-Oriented Requirements Engineering for SPL	403
13.1	Introduction	403
13.2	Alarm System Example	405
13.3	UML4PF Extension for Modeling Variability	405
13.4	PREVISE Method and its Application	408
13.4.1	Product Line Requirement Model Creation	411
13.4.2	Deriving a Concrete Product Requirement Model	423
13.5	Related Work	427
13.6	Contributions	429
14	Conclusions	431
14.1	Summary	431
14.2	Answer to Research Questions	433
14.3	Future Research	437

14.3.1	Risk Analysis for Deriving Security Requirements	437
14.3.2	Integrating Tactics into the Process of Architectural Pattern Selection	438
14.3.3	Aspect-oriented Requirements Engineering with Problem Frames	438
14.3.4	Providing Support for SPL in the Architecture Phase	439
14.3.5	Architecture Views	440
14.3.6	Tool Support	440
A	OCL Expressions related to the UML profile Extension for Quality Requirements	443
B	Architectural Pattern Selection	445
B.1	Problem Frames Catalog	445
B.2	Question Catalog	447
B.3	Relations between Problem Frames and Questions	449
B.4	Benefits and Liabilities	454
B.5	Architectural Pattern Catalog	457
B.6	Initial Architecture - Port Types	458
C	Quality-specific Pattern Selection & Application	459
C.1	Problem-oriented Security Pattern Template for A1	459
C.2	Problem-oriented Security Pattern Template for A2	461
C.3	Problem-oriented Security Pattern Template for A3	464
D	Quality-based Architecture	469
E	Architecture Evaluation Methods	473
	References	479

Bridging the Gap between Requirements Engineering
and Software Architecture

A Problem-Oriented and Quality-Driven Method

Alebrahim, A.

2017, XXVI, 500 p. 141 illus., Softcover

ISBN: 978-3-658-17693-8