

Contents

Part I Evolving Software Artefacts

1	An Overview of Requirements Evolution	3
	Neil Ernst, Alexander Borgida, Ivan J. Jureta and John Mylopoulos	
1.1	Introduction	4
1.2	Historical Overview of Requirements Evolution	6
1.2.1	From Software Evolution to Requirements Evolution	8
1.2.2	Empirical Studies of Requirements Evolution	10
1.3	A Survey of Industry Approaches	13
1.3.1	Standards and Industry	13
1.3.2	Requirements Management Tools	13
1.3.3	Task Managers	14
1.3.4	Summary	15
1.4	Recent Research	15
1.4.1	Problem Frames Approach	16
1.4.2	Extensions of the NFR Framework	16
1.4.3	Run-time Adaptive Requirements	17
1.4.4	KAOS-based Approaches	17
1.4.5	Paraconsistent and Default Logics	18
1.4.6	Traceability Approaches	20
1.4.7	Feature Models	21
1.4.8	Summary	21
1.5	A Framework for Requirements Evolution	24
1.5.1	The Payment Card Industry Example	25
1.5.2	Methodological Guidance for Solving Unanticipated Changes	26
1.5.3	Revising Requirements	28
1.5.4	Selecting Non-Dominated Solutions	30
1.5.5	Summary	31
1.6	Conclusions	32

2	Coupled Evolution of Software Metamodels and Models	33
	Markus Herrmannsdörfer and Guido Wachsmuth	
2.1	Introduction	34
2.1.1	Metamodels and Models	34
2.1.2	Metamodel Evolution	36
2.1.3	Model Migration	36
2.2	Analysis: Classification of Coupled Evolution	38
2.2.1	Metamodel Aspect	39
2.2.2	Granularity	39
2.2.3	Language Preservation	39
2.2.4	Model Preservation	40
2.2.5	Reversibility	41
2.2.6	Automatability	41
2.3	Empirical Results: Metamodel Evolution in Practice	42
2.3.1	Evolution of the Unified Modeling Language	43
2.3.2	Evolution of Automotive Metamodels	44
2.3.3	Evolution of the Graphical Modeling Framework	45
2.3.4	Discussion of the Empirical Results	46
2.4	State-of-the-Art: Approaches and their Classification	48
2.4.1	Classification Scheme	49
2.4.2	Manual Specification Approaches	51
2.4.3	Metamodel Matching Approaches	53
2.4.4	Operator-based Approaches	54
2.4.5	Discussion of State-of-the-Art	55
2.5	Tool support: Available Tools and their Comparison	56
2.5.1	COPE / Edapt	56
2.5.2	Epsilon Flock	57
2.5.3	Comparison of Migration and Transformation Tools	58
2.5.4	Comparison of Model Migration Tools	60
2.5.5	Discussion of Tool Support	62
2.6	Conclusions	62
3	Software Product Quality Models	65
	Rudolf Ferenc, Péter Hegedűs and Tibor Gyimóthy	
3.1	Introduction	66
3.2	Evolution of Software Product Quality Models	68
3.2.1	Software Metrics	69
3.2.2	Early Theoretical Quality Models	70
3.2.3	Metrics-based Empirical Prediction Models	74
3.2.4	State-of-the-art Practical Quality Models	77
3.3	Application of Practical Quality Models in Software Evolution	87
3.3.1	A Cost Model Based on Software Maintainability	87
3.4	Tools Supporting Software Quality Estimation	91
3.4.1	Software QUALity Enhancement project (SQUALE)	91

3.4.2	Software Quality Assessment based on Lifecycle Expectations (SQALE)	92
3.4.3	QUAMOCO Quality Model	92
3.4.4	SIG Maintainability Model	94
3.4.5	Columbus Quality Model	94
3.5	Comparing the Features of the Quality Models and Tools	95
3.5.1	Comparing the Properties of Different Practical Models	96
3.5.2	Evaluating the Properties of the Different Tools	98
3.6	Conclusions	100

Part II Techniques

4	Search Based Software Maintenance: Methods and Tools	103
	Gabriele Bavota, Massimiliano Di Penta and Rocco Oliveto	
4.1	Introduction	104
4.2	An Overview of Search-Based Optimization Techniques	105
4.2.1	Hill Climbing	106
4.2.2	Simulated Annealing	107
4.2.3	Particle Swarm Optimization	108
4.2.4	Genetic Algorithms	109
4.2.5	Multi-Objective Optimization	110
4.3	Search-based Software Modularization	112
4.3.1	The Bunch approach for software modularization	113
4.3.2	Multi-Objective Modularization	115
4.3.3	Achieving different software modularization goals	116
4.3.4	Putting the developer in the loop: interactive software modularization	118
4.4	Software Analysis and Transformation Approaches	121
4.4.1	Program transformation	122
4.4.2	Automatic Software Repair	124
4.4.3	Model transformation	125
4.5	Search-based Software Refactoring	128
4.5.1	The CODE-Imp tool	129
4.5.2	Other search-based refactoring approaches	133
4.6	Conclusions	135
5	Mining Unstructured Software Repositories	139
	Stephen W. Thomas, Ahmed E. Hassan and Dorothea Blostein	
5.1	Introduction	140
5.2	Unstructured Software Repositories	141
5.2.1	Source Code	141
5.2.2	Bug Databases	142
5.2.3	Mailing Lists and Chat Logs	142
5.2.4	Revision Control System	143
5.2.5	Requirements and Design Documents	143
5.2.6	Software System Repositories	143

5.3	Tools and Techniques for Mining Unstructured Data	144
5.3.1	NLP Techniques for Data Preprocessing	144
5.3.2	Information Retrieval	146
5.4	The State of The Art	150
5.4.1	Concept/Feature Location and AOP	150
5.4.2	Traceability Recovery and Bug Localization	150
5.4.3	Source Code Metrics	151
5.4.4	Software Evolution and Trend Analysis	152
5.4.5	Bug Database Management	153
5.4.6	Organizing and Searching Software Repositories	153
5.4.7	Other Software Engineering Tasks	154
5.5	A Practical Guide: IR-based Bug Localization	155
5.5.1	Collect data	156
5.5.2	Preprocess the source code	156
5.5.3	Preprocess the bug reports	157
5.5.4	Build the IR model on the source code	158
5.5.5	Query the LDA model with a bug report	158
5.6	Conclusions	160
6	Leveraging Web 2.0 for software evolution	163
	Yuan Tian and David Lo	
6.1	Introduction	164
6.2	Web 2.0 Resources	166
6.2.1	Software Forums, Mailing Lists and Q&A Sites	166
6.2.2	Software Blogs & Microblogs	168
6.2.3	Software Forges	169
6.2.4	Other Resources	170
6.3	Empirical Studies	171
6.3.1	Software Forums, Mailing Lists and Q&A Sites	171
6.3.2	Software Blogs & Microblogs	172
6.3.3	Software Forges	173
6.4	Supporting Information Search	174
6.4.1	Searching for Answers in Software Forums	174
6.4.2	Searching for Similar Applications in Software Forges ..	177
6.4.3	Other studies	180
6.5	Supporting Information Discovery	181
6.5.1	Visual Analytics Tool for Software Microblogs	182
6.5.2	Categorizing Software Microblogs	184
6.5.3	Other studies	188
6.6	Supporting Project Management	189
6.6.1	Recommendation of Developers	189
6.6.2	Prediction of Project Success	192
6.6.3	Other studies	195
6.7	Open Problems and Future Work	196
6.8	Conclusions	197

Part III Evolution of specific types of software systems

7	Evolution of Web Systems	201
	Holger M. Kienle and Damiano Distanto	
7.1	Introduction	202
7.1.1	Reengineering	202
7.1.2	Evolution Challenges and Drivers	203
7.1.3	Chapter's Organization	205
7.2	Kinds of Web Systems and their Evolution	206
7.2.1	Static Web Sites	207
7.2.2	Web Applications	208
7.2.3	Web Services	212
7.2.4	Ajax-based Web Systems	214
7.2.5	Web Systems Leveraging Cloud Computing	215
7.2.6	HTML5-based Web Systems	216
7.3	Dimensions of Evolution	216
7.3.1	Architecture Evolution	217
7.3.2	Design Evolution	219
7.3.3	Technology Evolution	221
7.4	Research Topics	224
7.5	Sources for Further Reading	227
7.6	Conclusions	228
8	Runtime Evolution of Highly Dynamic Software	229
	Hausi Müller and Norha Villegas	
8.1	Introduction	230
8.2	A Case Study: Dynamic Context Monitoring	231
8.3	Assessing the Need for Runtime Evolution	232
8.4	Dimensions of Runtime Software Evolution	235
8.5	Control in Runtime Software Evolution	238
8.5.1	Feedback Control	238
8.5.2	Feedforward Control	238
8.5.3	Adaptive Control	241
8.6	Self-Adaptive Software Systems	243
8.6.1	Self-Managing Systems	244
8.6.2	The Autonomic Manager	244
8.6.3	The Autonomic Computing Reference Architecture	247
8.6.4	Self-Management Properties	249
8.7	Self-Adaptation Enablers for Runtime Evolution	253
8.7.1	Requirements at Runtime	254
8.7.2	Models at Runtime	254
8.7.3	Runtime Monitoring	255
8.7.4	Runtime Validation and Verification	256
8.8	Realizing Runtime Evolution in SMARTERCONTEXT	257
8.8.1	Applying the MAPE-K Loop Reference Model	258

8.8.2	Applying Requirements and Models at Runtime	260
8.9	Open Challenges	262
8.10	Conclusions	264
9	Evolution of Software Product Lines	265
	Goetz Botterweck and Andreas Pleuss	
9.1	Introduction	266
9.2	Software Product Lines	266
9.3	Characteristics of SPL Evolution	270
9.4	Approaches to SPL Evolution	273
9.4.1	Process Models for SPL Evolution	274
9.4.2	Modeling Evolution and Change	278
9.4.3	Migration to SPLE	283
9.4.4	Analyzing Evolution	286
9.4.5	Planning Evolution	288
9.4.6	Implementing Evolution	291
9.5	Conclusions	294
10	Studying Evolving Software Ecosystems based on Ecological Models	297
	Tom Mens, Maëlick Claes, Philippe Grosjean and Alexander Serebrenik	
10.1	Introduction	298
10.2	Ecosystem terminology	299
10.2.1	Natural ecosystems and ecology	299
10.2.2	Software ecosystems	301
10.2.3	Comparing natural and software ecosystems	308
10.3	Evolution	310
10.3.1	Biological evolution	310
10.3.2	Comparing biological evolution with software evolution	312
10.3.3	Transposing biological models to the software realm	313
10.4	Exploratory case study	315
10.4.1	The GNOME OSS ecosystem	315
10.4.2	Comparing GNOME with a natural ecosystem	319
10.4.3	Migration of GNOME developers	323
10.5	Conclusions	325
Appendices		
A	Emerging trends in software evolution	329
	Alexander Serebrenik, Tom Mens	
B	List of acronyms	333
C	Glossary of Terms	337

Contents	xxiii
D Resources	343
E Datasets	349
References	351
Index	399



<http://www.springer.com/978-3-642-45397-7>

Evolving Software Systems

Mens, T.; Serebrenik, A.; Cleve, A. (Eds.)

2014, XXIII, 404 p., Hardcover

ISBN: 978-3-642-45397-7