TABLE OF CONTENTS

PART I: INTRODUCTION 21

CHAPTER 1. INTRODUCTION 23
1.1 Motivation 24
1.1.1 Self-Management 24
1.1.2 Runtime Maintenance 25
1.2 Overall Aim and Objectives 26
1.3 Research Methodology 28
1.4 Research Hypothesis 30
1.5 Thesis Overview 31

CHAPTER 2. CONTEXT 33
2.1 Introduction 33
2.2 Software Architectures 34
2.2.1 Definition 35
2.2.2 Basic Concepts 37
2.2.2.1 Component 37
2.2.2.2 Connector 38
2.2.2.3 Configuration 39
2.2.2.4 System 40
2.2.2.5 Port 40
2.2.2.6 Connection 40
2.2.2.7 Compositional Relationship 41
2.2.2.8 Other concepts 41
2.3 Aspect-Oriented Software Development 41
2.3.1 Aspect-Oriented Programming 42
2.3.2 Basic Concepts 44
2.3.3 Aspects in the Software Life Cycle 45
2.4 PRISMA 45
2.4.1 Model and ADL selection 46
2.4.2 Aspects as First-Class Citizens 47
2.4.3 PRISMA Architectural Elements 49

2.4.4 Levels of abstraction 51
2.4.5 Model-Driven Development Support 53
2.4.5.1 PRISMA in MOF 54
2.4.5.2 The PRISMA MDD Process 55
2.5 Conclusions 57

PART II: STATE OF THE ART 59

CHAPTER 3. DYNAMIC SOFTWARE EVOLUTION 61
3.1 Introduction 61
3.2 Software Evolution 61
3.2.1 Software Maintenance vs Software Evolution 62
3.2.2 Evolution as part of the Development Process 63
3.3 Dynamic Software Evolution 64
3.3.1 Changing software artefacts 64
3.3.2 Introducing changes at runtime 65
3.3.3 Dynamic Evolution: Definitions 66
3.3.4 Kinds of Dynamic Evolution 68
3.3.4.1 Granularity of changes 69
3.3.4.2 Activeness of change 70
3.4 Main Issues of Dynamic Evolution 71
3.4.1 Safe Stopping of Running Systems 72
3.4.1.1 Quiescence 72
3.4.1.2 Tranquillity 74
3.4.1.3 Other approaches for Safe Stopping 76
3.4.2 Updating Stateful Artefacts 78
3.4.2.1 No State Transfer 79
3.4.2.2 Delegated State Transfer 79
3.4.2.3 Automated State Transfer 80
3.5 Other approaches for dynamic change management 82
3.5.1 Control Systems 82
3.5.2 Autonomic Computing 83
3.5.3 Computational Reflection 87
3.6 Dynamic Evolution in Software Architectures 90
3.6.1 Dynamic Reconfiguration 90
3.6.2 Dynamic Evolution of Architectural Types 92
3.6.3 Combining both kinds of dynamism 93
3.7 Conclusions 93

CHAPTER 4. RELATED WORKS 95
4.1 Introduction 95
4.2 Dynamic Evolution Approaches 96
4.2.1 Formal, Dynamic ADLs for Reconfiguration 96
4.2.1.1 Process Algebra Formalisms 96
4.2.1.2 Graph-Based Formalisms 97
4.2.1.3 Reflection-Based Formalisms 99
4.2.2 Systems for Dynamic Change Support 100
4.2.2.1 Procedural and Object-Based Techniques 100
4.2.2.2 Dynamic Weaving Techniques in AOP 102
4.2.2.3 Component-Based Dynamic Frameworks 104
4.2.3 Self-Managed Software Architectures 108
4.2.3.1 Top-down approaches: Self-Adaptive Systems 108
4.2.3.2 Bottom-up approaches: Decentralized Architecture-Based Systems 114
4.2.4 AOSD & Evolution Concerns 117
4.3 Comparison of the different approaches 121
4.3.1 Description of the attributes selected 121
4.3.2 Comparison tables 124
4.4 Conclusions 128

PART III: DYNAMIC PRISMA 131

CHAPTER 5. CASE STUDY: AGROBOT 133
5.1 Introduction: Agricultural Robotics 133
5.2 Robotic Software Architectures 135
5.3 Dynamic Evolution in Robotic Software Architectures 136
5.4 Case Study: Agrobot, An Autonomous Robot for Plague control 137
5.4.1 Main Architecture 138
5.4.2 Composite Components: VisionSystem 140
5.4.3 Simple Components: ImageProcCard 143
5.4.4 Aspects: ImageProcSwController 145
5.5 Dynamic Evolution Requirements of the Agrobot 147
5.5.1 Dynamic reconfiguration scenario: VisionSystem fault-tolerance support 148
5.5.2 Dynamic type evolution scenario: Changing the ImageProcSoftware component 152
5.6 Conclusions 155

CHAPTER 6. AUTONOMIC RECONFIGURATION 157
6.1 Introduction 157
6.2 Characteristics of the approach 158
6.3 Reconfiguration management model 161
6.3.1 A control loop for self-reconfiguration 162
6.3.2 Aspects versus modules 164
6.4 Description of the autonomic reconfiguration aspects 165
6.4.1 The Monitoring Aspect 166
6.4.1.1 Introspection Services 168
6.4.1.2 Runtime Status Information 172
6.4.1.3 Event Interception Services 173
6.4.2 The Reconfiguration Analysis Aspect 177
6.4.2.1 Structure of the Reconfiguration Analysis Aspect 177
6.4.2.2 Reconfiguration Triggers 181
6.4.2.3 Configuration Transactions 183
6.4.2.4 Adding Inference Mechanisms 188
6.4.3 The Reconfiguration Coordination Aspect 188
6.4.3.1 Domain-Specific Reconfiguration Services 189
6.4.3.2 Transactional Management of Reconfiguration Plans 192
6.4.3.3 Generic Reconfiguration Services 200
6.4.4 The Reconfiguration Effector Aspect 212
6.4.4.1 Services for Safe Stopping 213
6.4.4.2 Services for Reconfiguration 214
6.4.4.3 Services for Updating, Recovery and Mobility 215
6.5 The Evolver Component 220
6.5.1 Structure of the Evolver Component 220
6.5.2 Support for Reactive Reconfigurations 223
6.5.3 Weaving the Reconfiguration Aspects Together 227
6.5.4 Evolver Specification 231
6.5.4.1 Evolver Template: The User-Defined Part 231
6.5.4.2 Evolver Mechanisms: The Generated Functionality 234
6.5.4.3 Consistence of Generated Code 236
6.6 Example: autonomic reconfiguration in the VisionSystem architecture 238
6.7 Conclusions & further works 242
6.7.1 Conclusions 242
6.7.2 Further works 244
6.7.3 Results 245

CHAPTER 7. DYNAMIC EVOLUTION OF ARCHITECTURAL TYPES.....247
7.1 Introduction.....247
7.2 Basis of the dynamic evolution of architectural types.....248
7.2.1 Definitions of Type, Instance and Architectural Type..... 248
7.2.2 Definition of the Dynamic Evolution of Architectural Types 250
7.3 Reflective Asynchronous Evolution of Architectural Types.....251
7.3.1 System-level evolutions vs type-level evolutions..... 252
7.3.2 A reflective model for evolvable types 254
7.3.2.1 Reflection: The Abstract Model..... 254
7.3.2.2 Type Meta-Instances: The Concrete Model..... 255
7.3.2.3 Reification of Types..... 257
7.3.2.4 Evolution Process Overview..... 259
7.3.3 Evolving instances through transformations..... 261
7.3.4 An asynchronous model for types evolution..... 264
7.3.4.1 Modelling evolutions over time 264
7.3.4.2 Additional characteristics..... 269
7.3.5 Description of the evolution infrastructure 271
7.3.5.1 Type-level Evolution 272
7.3.5.2 Instance-level Evolution..... 297
7.3.6 Summary of the evolution process 311
7.4 Conclusions & further works.....314
7.4.1 Conclusions..... 314
7.4.2 Further works 315
7.4.3 Results 316

CHAPTER 8. DESCRIPTION OF THE EVOLUTION SEMANTICS 319
8.1 Introduction.....319
8.2 Challenges of asynchronous evolution.....319
8.3 Evolution semantics 320
8.3.1 Specification of evolution processes..... 321
8.3.2 Version management: Evolution Tags 322
8.3.3 Formalisation of the evolution operations..... 325
8.3.3.1 Architecture-based Concrete Syntax 326
8.3.3.2 Graph-based Abstract Syntax..... 340
8.3.4 Discussion 347
8.4 Conclusions & further works.....348
8.4.1 Conclusions..... 349
8.4.2 Further works 349
8.4.3 Results 350

PART IV: CONCLUSIONS & FURTHER WORK 351

CHAPTER 9. CONCLUSIONS 353

9.1 Conclusions 353

9.1.1 Contributions 354
9.1.2 Evaluation of the approach 357
9.1.3 Evaluation of the research 363

9.2 Results of the PhD 365

9.3 Further research 371

9.3.1 Model-Driven Development support for evolvable systems 371
9.3.2 Proactive non-programmed evolutions 372
9.3.3 Definition of evolution constraints 372
9.3.4 Coordination of decentralized Evolvers 373
9.3.5 Formal analysis 373
9.3.6 Tool support for advanced graph transformations 375

PART V: APPENDIXES 377

APPENDIX A. PRISMA SPECIFICATIONS OF THE VISIONSYSTEM.....379

A.1 Specification of the VisionSystem type 379

A.1.1 Interfaces 379
A.1.2 Data Domains 380
A.1.3 External Functions 380
A.1.4 Architecture and configurations 381
A.1.5 Components 385
A.1.6 Connectors 387
A.1.7 Aspects 388

A.2 Reconfiguration Elements 393

A.2.1 Interfaces 393
A.2.2 Evolver Component 395
A.2.3 Reconfiguration Analysis Aspect 400
A.2.4 Reconfiguration Coordination Aspect 404
A.2.5 Architecture Monitoring Aspect 414
A.2.6 Architecture Effector Aspect 415

A.3 Type Evolution Elements 416

A.3.1 Data structures 416
A.3.2 Interfaces 420
A.3.3 Type Description Aspect 422
A.3.4 Type Evolution Aspect 426
A.3.5 Evolution Monitoring Aspect 427
A.3.6 Builder Aspect 428
A.3.7 Instance Evolution Planning aspect 430

A.3.8 Instance Monitoring aspect 432
A.3.9 Instance Effector aspect 432

APPENDIX B. EXTENSIONS OF THE PRISMA AOADL 435

B.1 Lists 435
B.2 Iterations and Loops 436
B.3 Partial Definitions of Software Artefacts 437

LIST OF FIGURES 441

REFERENCES 445