Mälardalen University Press Dissertations
No. 171

PRESERVATION OF EXTRA-FUNCTIONAL PROPERTIES IN EMBEDDED SYSTEMS DEVELOPMENT

Mehrdad Saadatmand

2015

School of Innovation, Design and Engineering

ISBN 978-91-7485-182-3
ISSN 1651-4238
Printed by Arkitektkopia, Västerås, Sweden

Mälardalen University Press Dissertations
No. 171

PRESERVATION OF EXTRA-FUNCTIONAL PROPERTIES IN EMBEDDED SYSTEMS DEVELOPMENT

Mehrdad Saadatmand

Akademisk avhandling

som för avläggande av teknologie doktorsexamen i datavetenskap vid
Akademin för innovation, design och teknik kommer att offentligen försvaras
tisdagen den 24 februari 2015, 13.15 i Gamma, Mälardalens högskola, Västerås.

Fakultetsopponent: Associate Professor Vittorio Cortellessa, University of L'Aquila, Italy

Akademin för innovation, design och teknik

Abstract

The interaction of embedded systems with their environments and their resource limitations make it important to take into account properties such as timing, security, and resource consumption in designing such systems. These so-called Extra-Functional Properties (EFPs) capture and describe the quality and characteristics of a system, and they need to be taken into account from early phases of development and throughout the system's lifecycle. An important challenge in this context is to ensure that the EFPs that are defined at early design phases are actually preserved throughout detailed design phases as well as during the execution of the system on its platform. In this thesis, we provide solutions to help with the preservation of EFPs; targeting both system design phases and system execution on the platform. Starting from requirements, which form the constraints of EFPs, we propose an approach for modeling Non-Functional Requirements (NFRs) and evaluating different design alternatives with respect to the satisfaction of the NFRs. Considering the relationship and trade-off among EFPs, an approach for balancing timing versus security properties is introduced. Our approach enables balancing in two ways: in a static way resulting in a fixed set of components in the design model that are analyzed and thus verified to be balanced with respect to the timing and security properties, and also in a dynamic way during the execution of the system through runtime adaptation. Considering the role of the platform in preservation of EFPs and mitigating possible violations of them, an approach is suggested to enrich the platform with necessary mechanisms to enable monitoring and enforcement of timing properties. In the thesis, we also identify and demonstrate the issues related to accuracy in monitoring EFPs, how accuracy can affect the decisions that are made based on the collected information, and propose a technique to tackle this problem. As another contribution, we also show how runtime monitoring information collected about EFPs can be used to fine-tune design models until a desired set of EFPs are achieved. We have also developed a testing framework which enables automatic generation of test cases in order verify the actual behavior of a system against its desired behavior. On a high level, the contributions of the thesis are thus twofold: proposing methods and techniques to 1) improve maintenance of EFPs within their correct range of values during system design, 2) identify and mitigate possible violations of EFPs at runtime.

ISBN 978-91-7485-182-3
ISSN 1651-4238

Abstract

The interaction of embedded systems with their environments and their resource limitations make it important to take into account properties such as timing, security, and resource consumption in designing such systems. These so-called Extra-Functional Properties (EFPs) capture and describe the quality and characteristics of a system, and they need to be taken into account from early phases of development and throughout the system's lifecycle. An important challenge in this context is to ensure that the EFPs that are defined at early design phases are actually preserved throughout detailed design phases as well as during the execution of the system on its platform.

In this thesis, we provide solutions to help with the preservation of EFPs, targeting both system design phases and system execution on the platform. Starting from requirements, which form the constraints of EFPs, we propose an approach for modeling Non-Functional Requirements (NFRs) and evaluating different design alternatives with respect to the satisfaction of the NFRs. Considering the relationship and trade-off among EFPs, an approach for balancing timing versus security properties is introduced. Our approach enables balancing in two ways: in a static way resulting in a fixed set of components in the design model that are analyzed and thus verified to be balanced with respect to the timing and security properties, and also in a dynamic way during the execution of the system through runtime adaptation. Considering the role of the platform in preservation of EFPs and mitigating possible violations of them, an approach is suggested to enrich the platform with necessary mechanisms to enable monitoring and enforcement of timing properties. In the thesis, we also identify and demonstrate the issues related to accuracy in monitoring EFPs, how accuracy can affect the decisions that are made based on the collected information, and propose a technique to tackle this problem. As another contribution, we also show how runtime

monitoring information collected about EFPs can be used to fine-tune design models until a desired set of EFPs are achieved. We have also developed a testing framework which enables automatic generation of test cases in order verify the actual behavior of a system against its desired behavior.

On a high level, the contributions of the thesis are thus twofold: proposing methods and techniques to 1) improve maintenance of EFPs within their correct range of values during system design, 2) identify and mitigate possible violations of EFPs at runtime.

Sammanfattning

Interaktionen mellan inbyggda system och dess miljöer gör det viktigt att ta hänsyn till egenskaper såsom timing, säkerhet, och tillförlitlighet vid utformning av sådana system. Dessa så kallade Extra-Funktionella Egenskaper (EFE:er) innefattar och beskriver kvalitet och egenskaper hos ett system, och måste beaktas tidigt i utvecklingsfasen samt under hela systemets livscykel. En central utmaning i detta sammanhang är att säkerställa att de EFE:er som är definierade i tidiga designfaser bevaras genom de detaljerade konstruktionsfaserna så väl som under exekveringen av systemet på sin plattform.

I denna avhandling tillhandahåller vi lösningar för att stödja bevarandet av EFE:er, med inriktning på både systemkonstruktionsfaserna och plattformsexekvering. Med utgångspunkt från kraven, som begränsar EFE:erna, föreslår vi en strategi för modellering av Icke-Funktionella Krav (IFK) och utvärdering av olika designalternativ med avseende på uppfyllandet av IFK. Med kopplingen och avvägningen emellan EFE:er i åtanke, introduceras ett tillvägagångssätt för att balansera tidsegenskaperna gentemot säkerhetsaspekterna. Vår metod gör det möjligt att balansera på två sätt: statiskt, som resulterar i en fast uppsättning komponenter i konstruktionsmodellen som analyseras och därigenom verifieras som balanserade med avseende på tids och säkerhetsegenskaper, samt dynamiskt genom anpassning under systemexekvering. Med hänsyn till plattformens roll i bevarandet av EFE:er samt mildrande av eventuella kränkningar, föreslås ett sätt för att berika en plattform med nödvändiga mekanismer för att möjliggöra övervakning samt säkerställande av tidsegenskaper. I avhandlingen både identifierar och demonstrerar vi problematiken med noggrannhet i övervakning av EFE:er, och hur noggrannheten kan påverka de beslut som fattas grundat på insamlad information, samt föreslår en teknik för att lösa detta problem.

Som ytterligare bidrag visar vi också på hur information om EFE:er som insamlats genom övervakning under drift kan användas för att finjustera designmodeller tills en önskad uppsättning EFE:er uppnås. Vi har också utvecklat ett ramverk för testning som möjliggör automatisk generering av testfall för kontrollera att faktiskt beteende hos ett system överensstämmer med önskat beteende.
Som helhet är således avhandlingens bidrag dubbelt: föreslagna metoder och tekniker till att 1) förbättra bevarandet av EFE:er inom tillåtet intervall under systemdesign, 2) identifiera och mildra eventuella överträdelser av EFE:er under körning. Ur detta perspektiv bidrar våra lösningar till att producera inbyggda system med bättre kvalitetssäkring.

Dissertation Opponent:

- Assoc. Prof. Vittorio Cortellessa - University of L'Aquila, Italy.

Grading Committee:

- Prof. Antonia Bertolino - National Research Council (CNR), Italy.
- Prof. Jan Bosch - Chalmers University of Technology, Sweden.
- Adj. Prof. Tiberiu Seceleanu - ABB Corporate Research, Sweden.

Grading Committee Reserve:

- Prof. Kristina Lundqvist - Mälardalen University, Sweden.

PhD Advisors:

- Prof. Mikael Sjödin - Mälardalen University, Sweden.
- Dr. Antonio Cicchetti - Mälardalen University, Sweden.

To my dear family,

Massoud, Forough, Mahnaz, Farshid & Farshad.

“A PhD is someone who knows everything about something & something about everything” - anonymous.

Acknowledgements

The journey towards a PhD degree is full of joy, ups and downs, excitements, and challenges. There have been many people who have been with me throughout this journey; people from which I have learned a lot, shared our joys and excitements together, those that together we worked countless hours to solve problems and tackle challenges, those who provided support and made the progress smoother and easier, and those people whose mere acquaintance and presence have been a great source of inspiration and motivation.

Hereby, I would like to thank my supervisors Mikael Sjödin and Antonio Cicchetti for their support, encouragement, and all the things that I learned from them helping me become a better researcher. Thanks to Radu Dobrin, Jan Carlson and Cristina Seceleanu for their invaluable comments and tips. I had the pleasure of sharing the office with Federico and Antonio with whom I have also great and unforgettable memories from all the travels that we did together. I would also like to thank my managers and colleagues at Alten and Enea, particularly Detlef Scholle.

The success of Mälardalen Real-Time Research Centre (MRTC) at IDT with its friendly, pleasant and enriching environment is due to the hard work and presence of many great people and researchers and I am glad that I have had the chance to be part of such an environment. My studies at MDH also gave me the opportunity to meet new friends and work with many wonderful people. I would like to thank them all here for the all the great moments. Thanks also to the ITS-EASY graduate school staff for their support, and the nice educational events they organized.

My deepest gratitude to my family who have always been there for me. Without them I could have never reached this far.

Mehrdad Saadatmand
Västerås, February 2015

List of Publications

Papers Included in the PhD Thesis[1]

Paper A *Model-Based Trade-off Analysis of Non-Functional Requirements: An Automated UML-Based Approach.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. Journal of Advanced Computer Science, Vol. 3, No. 11, November, 2013.

Paper B *Managing Timing Implications of Security Aspects in Model-Driven Development of Real-Time Embedded Systems.* Mehrdad Saadatmand, Thomas Leveque, Antonio Cicchetti, Mikael Sjödin. International Journal On Advances in Security, Vol. 5, No. 3&4, December, 2012.

Paper C *Monitoring Capabilities of Schedulers in Model-Driven Development of Real-Time Systems.* Mehrdad Saadatmand, Mikael Sjödin, Naveed Ul Mustafa. 17th IEEE International Conference on Emerging Technologies & Factory Automation (ETFA), Krakow, Poland, September, 2012.

Paper D *An Automated Round-trip Support Towards Deployment Assessment in Component-based Embedded Systems.* Federico Ciccozzi, Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 16th International Symposium on Component-Based Software Engineering (CBSE), Vancouver, Canada, June, 2013.

[1] The included articles have been reformatted to comply with the PhD thesis layout.

Paper E *Towards Accurate Monitoring of Extra-Functional Properties in Real-Time Embedded Systems.* Mehrdad Saadatmand, Mikael Sjödin. 19th Asia-Pacific Software Engineering Conference (APSEC), Hong Kong, December, 2012.

Paper F *A Model-Based Testing Framework for Automotive Embedded Systems.* Raluca Marinescu, Mehrdad Saadatmand, Alessio Bucaioni, Cristina Seceleanu, Paul Petterson. 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Verona, Italy, August, 2014.

Paper G *Testing of Timing Properties in Real-Time Systems: Verifying Clock Constraints.* Mehrdad Saadatmand, Mikael Sjödin. 20th Asia-Pacific Software Engineering Conference (APSEC), Bangkok, Thailand, December, 2013.

Related Publications not Included in the PhD Thesis

Licentiate Thesis[2]

1. *Satisfying Non-Functional Requirements in Model-Driven Development of Real-Time Embedded Systems.* Mehrdad Saadatmand, Licentiate Thesis, ISSN 1651-9256, ISBN 978-91-7485-066-6, May, 2012.

Conferences & Workshops

1. *A Fuzzy Decision Support Approach for Model-Based Tradeoff Analysis of Non-Functional Requirements.* Mehrdad Saadatmand, Sahar Tahvili. 12th International Conference on Information Technology : New Generations (ITNG), Las Vegas, USA, April, 2015.

2. *Mapping of State Machines to Code: Potentials and Challenges.* Mehrdad Saadatmand, Antonio Cicchetti. The Ninth International Conference on Software Engineering Advances (ICSEA), Nice, France, October, 2014.

3. *OSLC Tool Integration and Systems Engineering - The Relationship Between The Two Worlds.* Mehrdad Saadatmand, Alessio Bucaioni. 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Verona, Italy, August, 2014.

4. *Runtime Verification of State Machines and Defect Localization Applying Model-Based Testing.* Mehrdad Saadatmand, Detlef Scholle, Cheuk Wing Leung, Sebastian Ullström, Joanna Fredriksson Larsson. First Workshop on Software Architecture Erosion and Architectural Consistency (SAEroCon) (ACM) (Co-located with WICSA 2014), Sydney, Australia, April, 2014.

5. *Run-Time Monitoring of Timing Constraints: A Survey of Methods and Tools.* Nima Asadi, Mehrdad Saadatmand, Mikael Sjödin. The Eighth International Conference on Software Engineering Advances (ICSEA), Venice, Italy, October 27 - November 1, 2013.

[2] A licentiate degree is a Swedish graduate degree halfway between M.Sc. and Ph.D.

6. *On Combining Model-Based Analysis and Testing.* Mehrdad Saadatmand, Mikael Sjödin. 10th International Conference on Information Technology : New Generations (ITNG), Las Vegas, USA, April, 2013.

7. *Toward Model-Based Trade-off Analysis of Non-Functional Requirements.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 38th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Cesme-Izmir, Turkey, September, 2012.

8. *Modeling Security Aspects in Distributed Real-Time Component-Based Embedded Systems.* Mehrdad Saadatmand, Thomas Leveque. 9th International Conference on Information Technology : New Generations (ITNG), Las Vegas, USA, April, 2012.

9. *Design of Adaptive Security Mechanisms for Real-Time Embedded Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 4th International Symposium on Engineering Secure Software and Systems (ESSoS), Eindhoven, The Netherlands, February, 2012.

10. *UML-Based Modeling of Non-Functional Requirements in Telecommunication Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. The Sixth International Conference on Software Engineering Advances (ICSEA), Barcelona, Spain, October, 2011

11. *On Generating Security Implementations from Models of Embedded Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. The Sixth International Conference on Software Engineering Advances (ICSEA), Barcelona, Spain, October, 2011.

12. *Enabling Trade-off Analysis of NFRs on Models of Embedded Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 16th IEEE International Conference on Emerging Technologies & Factory Automation (ETFA), WiP session, Toulouse, France, September, 2011.

13. *A Methodology for Designing Energy-aware Secure Embedded Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 6th IEEE International Symposium on Industrial Embedded Systems (SIES), Västerås, Sweden, June, 2011.

14. *Toward a Tailored Modeling of Non-Functional Requirements for Telecommunication Systems.* Mehrdad Saadatmand, Antonio Cicchetti, Diarmuid Corcoran, Mikael Sjödin. 8th International Conference on Information Technology : New Generations (ITNG), Las Vegas, USA, April, 2011.

15. *On the Need for Extending MARTE with Security Concepts.* Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin. 2nd International Workshop on Model Based Engineering for Embedded Systems Design (M-BED), Grenoble, France, March, 2011.

Contents

I

Thesis

Chapter 1

Introduction

Once upon a time programmers used to get excited when they found out that the programs they had punched on a card just worked and could perform the expected calculations and functions. Although the computers back then also had their own limitations in terms of memory and processing capacity, getting the right functionality done was a big enough challenge in itself, which also meant not needing to redo programming on another punchcard. With respect to complexity, computer systems today are rapidly becoming more and more complex. There are several factors that contribute to this complexity. One factor is the growing demands and expectations on the services provided by these systems. For instance, more tasks are delegated to software and electric/electronic components in a car nowadays, which used to be performed solely by mechanical and hydraulic parts. On the other hand, the complexity of each service itself is increasing as well. Another aspect that contributes to the complexity issue is the complexity and variety of the infrastructure and platforms on which systems are deployed; i.e., different operating systems and frameworks (e.g., Android, iOS, .NET, JVM), different hardware (e.g., single core, multicore, 32/64-bit processors). In case of embedded systems, limitations and constraints on available resources also add another dimension to the complexity issue. This means that such systems should operate within certain constraints. In other words, other concerns than just logical correctness of operations can play an important role in determining the success and correctness of embedded systems. These limitations and constraints are captured in the form of

Non-Functional Requirements (NFRs) and Extra-Functional Properties (EFPs). However, EFPs like NFRs [1] cannot be considered individually as they have interdependencies and mutual impacts and tuning one EFP can affect another one.

The ultimate goal in the design of a software system is to deliver a product which satisfies all the requirements of different stakeholders. To assess the fulfillment of this goal, not only it is needed to be able to verify certain properties in the end product (e.g., timing properties), but also it is important from early design phases to consider a set of system components, among different alternatives, that have desired properties in line with and contributing to the overall satisfaction of system requirements and not violating them. One important challenge in this context is that as we build a system and go down the abstraction levels and finally get to its execution, the properties of interest which are related and relevant to the satisfaction of system requirements should be *preserved* and not deviate from their desired values. For example, if at runtime the execution time of a task exceeds a certain threshold and value, it can impact the satisfaction of a timing requirement on end-to-end response time in the system.

Considering the aforementioned points, two challenges with respect to EFPs in embedded systems can be identified: 1-coming up with a system design (e.g., models) which respects the desired set of properties (considering their interdependencies and trade-offs); and 2-making sure that those properties remain valid in the final product and during execution. The solutions that are provided in this thesis under the title of preservation of extra-functional properties mainly tackle these two issues. The importance of the contributions of this thesis lies in the fact that regardless of the type and amount of analyses done in building a system, if there are any deviations between the expected EFPs and the actual ones at runtime, the developed system might be then a failure.

1.1 Background and Motivation

The number of computer systems that we use in our daily life which are embedded as part of other devices and systems is rapidly growing. Examples of such systems are microwave ovens, automobiles, TV sets, digital cameras, and refrigerators. Embedded computer systems are systems that are designed basically to control and operate as part of other

devices. These systems are usually designed for specific and dedicated operations, and interact with their external environment through sensors and actuators [2, 3]. This interaction often brings along additional requirements such as real-time and safety. However, besides such requirements, resource constraints in these systems also introduce other requirements with respect to power consumption, processing capacity, memory usage, etc. Due to resource constraints that these systems have, their correctness depends not only on providing the right functionality but also respecting the constraints that they have. For this reason, it is of great importance to be able to evaluate EFPs and preserve them within their acceptable range of values mitigating possible violations of them at runtime.

In this context, requirements serve as a means for capturing and expressing the needs of different stakeholders of the system. Considering the variety of stakeholders, requirements can be originated from different sources; such as customers and end users, standards and regulations, developers and refinement of higher level requirements, development tools, operational context and environment of the system, and so on. As for Non-Functional Requirements (NFRs), they have specific challenges which can make their satisfaction in a system a complicated task [4, 5, 6]. For instance, NFRs cannot usually be considered in isolation as they are interconnected and have dependencies, and also can span different aspects and parts of a system. Therefore, in satisfying an NFR, its impacts on other NFRs should also be taken into account, and trade-off analysis among NFRs needs to be done [7, 8]. A similar procedure needs to be done for EFPs when adjusting and tuning different properties of the system.

There is a clear and important (but sometimes misunderstood) relationship between NFR and EFP. For example, 'response time of a component should not exceed 2ms' is a requirement, while 'response time of component A never exceeds 2ms' or 'response time of component B is equal to 1ms' are expressions of properties. It is only in relation to a requirement that it becomes possible to then talk about validity/invalidity or 'goodness/badness' of the value of a property. For instance, just knowing that a software component has the worst-case execution time of 200ms does not help in determining if it is suitable to be used in building a particular system or not. Only when the requirements are taken into account, this value gets meaning in the sense that if it is good for that system and context or not; in which case, another component

with a different worst-case execution time might be adopted. Similarly, when monitoring and evaluating EFPs, requirements are needed in order to determine if the value of an EFP is valid and acceptable or not. Balancing trade-offs among requirements can also incur adjusting system properties that are related to those requirements. Therefore, there is a relationship between an NFR and EFPs in a system and in order to satisfy an NFR, its related extra-functional properties should have valid values. For example, to satisfy performance requirements in a real-time system, execution and response time values of tasks (among others) should remain within a valid range.

To tackle the design complexity issue of embedded systems, Model-Driven Development (MDD) is a promising approach. It aids by raising abstraction level and reducing design complexity, which also enables analysis at earlier phases of development. This helps with the identification of problems before the implementation phase [9, 10], noting that the cost of problems when found at later phases, especially in the code and at runtime, grows exponentially [11, 12, 13]. The implementation of the system can also be generated from the design models through (a set of) model transformations. As more abstraction levels are introduced in designing a system, it becomes also important to verify consistency of design artifacts and their properties at each level. This means that for each transformation, input properties should be preserved in the output of the transformation. This also includes the system execution at runtime which is the result of executing the generated source code, implying that the execution platform should be able to actively monitor and preserve extra-functional properties at runtime. To this end, the execution platform also requires to be *semantically aware* of the specified properties and related events in order to monitor them and detect their deviations. In the context of MDD, preservation of EFPs also gains special importance and interest as it is ultimately the object code which is executing and controlling a system, not models per se. Moreover, in terms of timing properties, for instance, "models are only approximation of system implementations" and therefore, "inevitable time deviations of the implementation from the model appear" [14, 15].

Regardless of the applied development method and despite all types of analyses done at earlier phases, the ultimate goal is to have a system which behaves correctly and as intended at runtime and during its execution. Moreover, for performing static analysis different assumptions are taken into account. At runtime, situations may still occur that lead to

the violation of those assumptions which in turn can mean invalidation of analyses' results [16, 17]. This again emphasizes the need for preservation of EFPs which constitute such assumptions, such as worst-case execution times of tasks, etc. For instance, it is very common nowadays that modern CPUs have Dynamic Voltage and Frequency Scaling (DVFS) support for power management purposes. When DVFS is applied, the CPU can go down to a lower frequency which then affects the execution times of tasks. In such a scenario, the results of timing analysis done assuming certain execution times of tasks based on a different CPU frequency may not be valid anymore. Of course, in an ideal case, all such scenarios should be considered in the analysis. However, such extensive and exhaustive analyses may not always be economical or even possible to perform, particularly in complex systems.

1.2 Problems and Contributions Overview

To provide support for preservation of EFPs, we first start from NFRs, and considering the relationships between different NFRs we introduce a method to evaluate their interdependencies and mutual impacts in a generic manner. This step is necessary considering the relationship of NFRs and EFPs as discussed in the previous section. Besides, early analysis of NFRs is also important considering that different sets of NFRs on the same set of functional requirements can result in different design decisions and implementations. For instance, to sort some data under a specific timing constraint, only certain sorting algorithms may be suitable to use. However, if that timing constraint is relaxed but instead there is a constraint on maximum memory usage, a different set of sorting algorithms can then be considered as suitable. Moreover, due to resource limitations, the NFRs that are defined for an embedded system by different stakeholders need to be balanced and trade-off analysis among them should be performed. As the next step in building a system which respects its specified constraints, we introduce an approach for establishing balance among different EFPs in an embedded system. This approach is demonstrated by looking particularly into the relationship between security and timing.

Another challenge is collecting information about each EFP in the system. In other words, appropriate mechanisms for monitoring EFPs are needed in order to determine whether a violation with respect to the

constraints of an EFP has occurred or not. Such monitoring information can then be used for different purposes: for example, to perform runtime adaptation, enforcing the system constraints, testing EFPs, or simply providing feedback on EFPs values collected during the system execution. Along with this goal, we have developed a method for collecting necessary information about the timing behavior of real-time systems to identify violations such as deadline misses and execution time overruns of real-time tasks. In this thesis, we also discuss and demonstrate a method how such monitoring capabilities and the EFPs information collected using them can help with adjusting system models towards reaching deployment configurations which ultimately result in a desired set of EFPs' values at runtime.

Moreover, we demonstrate how monitored information can be used to perform runtime adaptation and also test a system with respect to its timing properties. For the former, a runtime adaptation mechanism has been developed to balance timing aspects of a system versus its security level by adopting different encryption algorithms with different timing properties. For the latter, we have developed a testing approach to verify whether the timing constraints (e.g., specified in the form of clock constraints) in a system get violated at runtime or not.

As can be understood so far, one fundamental feature needed in achieving preservation of EFPs and to mitigate their possible violations is the ability to monitor and collect information about EFPs. While such capability is assumed as provided in many previous works, it has its own unique challenges. For instance, EFPs are of different nature and kind, and therefore, the way that necessary information is collected for each one is a challenge in itself. Also accuracy of the collected information plays an important role when deciding if an EFP is within an acceptable range (as specified by an NFR). As another contribution of the thesis, we demonstrate the importance of this issue, how it can affect taking correct decisions about EFPs in a system, identify the factors that can contribute to the accuracy of the collected information (or lack thereof), and provide a solution for mitigating it.

In summary, the main goal of the thesis is to highlight the challenges with respect to the preservation of EFPs and provide techniques and methods applicable at different levels of abstraction to increase our confidence that at the final step, namely runtime and during execution, EFPs do not deviate from their expected values and the system behaves as intended. While the proposed solutions have the potential to be ap-

plied for various EFPs, our main focus in this work is particularly on timing properties.

1.3 Thesis Outline

The thesis consists of two parts: **Part I** includes five chapters. Chapter 1 provided an introduction to the thesis, described the background, and main problems that this thesis addresses as well as an overall description of the thesis contributions. Chapter 2 elaborates the research context, goals, and process. Contributions of the thesis are described in more detail in Chapter 3. Chapter 4 discusses the related work and in Chapter 5, the work is summarized and future directions are mentioned. **Part II** of the thesis contains the published peer-reviewed papers that constitute and present the technical contributions of the thesis in detail, which are organized in Chapters 6-12.

Chapter 2

Research Context

The main objective in this research work is to introduce methods for preservation of extra-functional properties in embedded systems development and to mitigate their possible violations at runtime. This goal is based on the following principles and assumptions:

- For each EFP to be preserved, a set of values containing acceptable and valid values for that EFP can be considered. Values that fall outside of this set are considered invalid, and thus violate the constraints of the EFP (considering a specific context).

- Analysis can be performed on system models in order to evaluate satisfaction feasibility of its non-functional requirements and to evaluate/calculate extra-functional properties of the system and its components, such as response time. Analyses are based on some assumptions and preconditions. Violation of these assumptions can lead to having invalid and erroneous analysis results.

- At runtime, several factors such as transient loads, difference between the ideal execution environment (taken into account for analysis) and the actual one, can lead to the violation of the assumptions that were used to perform analysis [16].

- It may not be practical and/or economical to perform analysis on all types of extra-functional properties. In such cases, runtime monitoring becomes even more important.

- As we go down the abstraction levels, an EFP may be refined and translated into one or more EFPs at the next levels. In that case, preservation of the former can incur preservation of the latter at a different abstraction level.

In this context, the term *preserve* that we use in our work implies that there is some knowledge about valid and invalid values (e.g., range of values) that an extra-functional property can have in a specific context and system; i.e., we actually preserve the *validity* of the value, rather than the actual value.

2.1 Research Goals

To achieve the objective of the work, the following main research goals have been defined:

- **G1: Define an approach for establishing balance among NFRs at the model level:** The relationship between NFRs and EFPs makes it important to start from NFRs and evaluate and balance them. For example, a requirement on end-to-end deadlines and response times in a system might require use of certain components with specific timing properties. If that requirement is changed, another set of components with different timing properties might be needed.

- **G2: A model-based EFP-aware method resulting in property preserving system designs:** Model-Based Development can enable identification of problems earlier in the development phase. By exploiting this feature, system models can be analyzed and adjusted to have extra-functional properties within their desired and acceptable range of values, and thus, support EFP preservation from earlier phases of development.

- **G3: A framework for monitoring and testing of extra-functional timing properties:** Without monitoring EFPs at runtime we cannot be sure whether they are preserved or not (somehow analogous to the Schrödinger's cat experiment [18]). A monitoring mechanism is also needed if a system requires to apply enforcement of EFPs (for instance, to preempt a task before it exceeds its worst-case execution time). Moreover, it should be

possible to test a system with respect to its EFPs, which requires to obtain and know the value of an EFP. In this research, we limit ourselves to timing properties while an industrial tool should support a wider range of EFPs.

2.2 Research Process

The main steps that have taken place in performing this research work are summarized and illustrated in Figure 2.1.

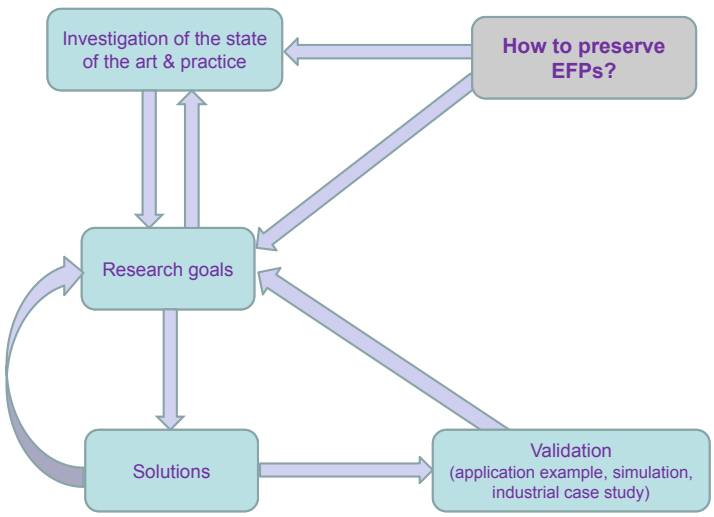

Figure 2.1: Research Steps

To achieve the overall objective of the thesis, namely preservation of EFPs, several research goals have been defined. Moreover, the state of the art and practice have been consulted, which resulted in additional research (sub-)goals or modification of already defined ones. To fulfill the research goals, solutions have been proposed. These solutions were

validated in different ways: by an application example to demonstrate how, for instance, a method is applied; by simulation; or by performing an industrial case study. The arrow in the figure from `Validation` to `Research Goals` indicates that in validating the solutions, new research goals or modification of existing ones have also formed and occurred.

Chapter 3

Contributions

In this section we provide the description of the main contributions of this work:

C1) Trade-off analysis and evaluation of conflicts among NFRs: To model interdependencies of NFRs and evaluate their impacts on each other, we have proposed a UML profile. The profile offers necessary concepts to generically model an NFR along with its refinements which can include one or several other NFRs as well as functional parts that contribute to its satisfaction. This way, it enables to create a hierarchy of NFRs, form child-parent relationship among them, and also establish relationships to the functional elements in the model that provide realization and implementations for NFRs. To enable trade-off analysis of NFRs in a quantitative manner, we introduce numerical properties as part of the defined stereotypes in the profile. This allows to calculate the satisfaction level of an NFR by taking into account both the contribution degree of each of its children NFRs and any impacts that other NFRs in the system may have on it. To automate and perform the analysis, an in-place model transformation is used to traverse model elements, perform necessary calculations based on the algorithms that are implemented as part of the transformations, and then update the respective properties of model elements with the calculated values. Based on the analysis results, it can be determined if the planned decision designs to fulfill NFRs are acceptable or not.

In the profile, we also introduce the concept of *deviation indicator*

(in Paper A in the thesis) for requirements, which helps to identify parts of a system which have greater degree of deviation with respect to satisfaction of an NFR. This provides for several interesting features, such as to prioritize test cases and focus testing activities on parts of the system with higher deviation indicator values (i.e., probably having more 'severe' problems) as demonstrated in a separate work of ours in [19]. This is particularly interesting and beneficial considering that there is only limited amount of resources (e.g., time or money) available for performing testing activities. Such prioritization based on the deviation indicator values allows for more efficient use of available resources for testing.

Another variation of our approach based on the NFR profile is presented in [20]; in which we have used fuzzy logic and decision support systems to identify best design alternatives (e.g., from a set of available components with different properties) in order to construct a system design which is optimized with respect to the overall satisfaction level of its NFRs. Application of fuzzy logic helps to relax the need for providing accurate quantified values for relationships among NFRs, as is expected in the original approach.

C2) Establishing balance between security and timing properties: The choice of security mechanisms in real-time embedded systems where timing requirements are critical may not be trivial. For example, performing encryption using a strong encryption algorithm may take longer time than using a weaker (but faster) encryption algorithm and this may lead to the violation of timing requirements. Therefore, in implementing security requirements, timing implications of the chosen security mechanisms should also be considered.

In this thesis, we provide two approaches to tackle this challenge. In the first one, we address it at the model level by identifying parts of the system (i.e., sensitive data) that need to be protected. For such parts, appropriate security features (in here, encryption and decryption components) to protect them are added to the original component model of the system. Then timing analysis is performed on the derived component model to ensure that the security features which are added do not violate the timing requirements. In this approach, the original component model of the system is used as input for a transformation that considers the sensitive data flows in the input model and adds appropriate security components considering their timing properties. This approach leads to

a static design in the sense that a fixed and particular security mechanism which is analyzed and thus, known to respect its allowed time budget is always used in each execution. Our approach also provides a form of separation of concerns in the sense that instead of defining security on the architectural (i.e., component) model, security engineers (or system designers) can now focus and annotate the data model, based on which the component model is then updated (through model transformation) to include appropriate security mechanisms. Moreover, in this way we also enable to bring security concerns into earlier phases of development. Considering security at earlier phases of development is a need which is getting more and more attention in computer systems today, particularly in the embedded domain where security has its own unique challenges (such as timing implications as we already discussed) [21].

However, the aforementioned solution may not be practical for systems with high complexity which are hardly analyzable or systems with unknown timing properties of their components. For such systems, an adaptive approach to select appropriate security mechanisms based on the state of the system can be used to adapt its behavior at runtime and stay within the timing constraints. To this end, as a second and complementary way to establish balance between security and timing, we have suggested an approach for selecting appropriate encryption algorithms (in terms of their timing behaviors) at runtime in an adaptive fashion. The approach works by keeping a log for each execution of the encryption process. Based on the logged information, if a timing violation is observed, a more suitable encryption algorithm (in terms of timing properties) is adopted in the next execution. In other words, the system adapts itself at runtime to keep the timing constraints and reduce the number of timing violations. The idea behind this adaptation mechanism is that when it is detected that an executing encryption algorithm is exceeding its allowed time budget, it can be more costly to terminate it in the middle of the encryption process and restart encrypting the data with a less time-consuming encryption algorithm. Instead we let it finish its job and then use another encryption algorithm with a lower execution time in the next invocation of the encryption process.

In summary, the former approach aims to help with establishing balance between security and timing properties at the design phase when system models are being created. In the second approach, this balance is established in an adaptive way at a later phase, namely during system

execution and at runtime. These two approaches can of course be used together in building a system in order to provide a higher degree of assurance to keep timing and security properties balanced and at acceptable levels.

C3) Runtime monitoring and enforcement of EFPs: The capabilities of platforms play an important role in preservation of extra-functional properties and mitigation of possible violations. To be able to actively monitor and enforce EFPs, the platform needs to be semantically aware of them. With respect to timing properties, this awareness can include properties of a real-time task such as activation pattern (i.e., periodic, sporadic, aperiodic), deadline, execution time, and so on. The platform needs to be aware of such properties so that, for example, it can detect deadline misses or execution time overruns. This also implies that the platform needs to have respective monitoring mechanisms for different timing properties. We have constructed a framework that brings such awareness about several timing properties to the platform and provides mechanisms for runtime monitoring and enforcement of such properties.

Moreover, in the context of a model-based development method, it is important to verify that the actual values of EFPs at runtime are in compliance with those expected and defined at the model level. The importance of such capability becomes more clear remembering that the end goal of all different development methods and techniques is to have a system which executes and behaves as expected at runtime with respect to both functional and extra-functional aspects. For this purpose, we show how having a monitoring framework enables to also build and provide a round-trip support in the development chain, constituting from design models down to code and its execution on the platform and back again to the models. In such a round-trip chain, monitoring information is propagated back to the design models where the expected values are compared with the actual values. If necessary, refinements and modifications are done to the model(s) in order to finally have a system with the desired set of EFPs. This is done by performing the process several times until the refined models and the code that is generated from them result in a system with satisfactory set of EFPs. This approach may well be used towards optimizing design models for specific properties. We have applied and validated our proposed round-trip method on the Ericsson's ATM Adaptation Layer 2 (AAL2) subsystem.

C4) Testing of timing properties and runtime model verification: Another challenge with respect to timing properties is how to test them. Testing EFPs, such as timing, can generally be a tricky task. One factor contributing to this issue is that testers need to collect and have necessary information about timing properties of a system so that they can actually then test it. This brings us back to the previous point about the capability to monitor and observe timing properties. To improve testability of a system with respect to its timing properties, we introduce a method and a testing framework that by exploiting monitoring capabilities of the platform enables testing of timing properties. Of course, testing per se does not guarantee absence of errors, but by detecting failures and fixing them it can help to increase our confidence in the correctness of a system. Our suggested testing approach enables automatic generation of Concrete Test Cases (CTC) (i.e., executable test scripts) from abstract ones (ATC: Abstract Test Cases) which are derived from Timed Automata (TA) models, executing them against the target, and determining timing issues and violations. This is done by implementing a parser which reads in ATCs and generates Python scripts based on them. This way, we not only enable testing of timing properties but also do so in an automatic way. We have applied and validated our approach on the Brake-by-Wire (BBW) use-case provided by Volvo.

The result of testing also indicates if there is any deviation between the intended behavior captured by models and the actual behavior of the system at runtime, which can also be useful in verifying architectural deviations and inconsistencies. This is important as models are also used to perform model-based analysis. Therefore, if they do not correctly represent a system's behavior, the analyses that are done based on them may also not be valid for that system.

Our proposed testing framework is covered by papers F and G in the thesis. In the former, the whole testing framework and test case generation methodology are explained with a focus on functional requirements. In the latter, the extensions to also enable testing of timing requirements are described (with a minor difference in how the System Under Test (SUT) code is implemented for mapping states and transitions to code [22], and a slightly different TA model for describing the internal behavior of system components). In our testing approach, timed auomata are used to describe the internal behavior of the components of the target system which is modeled in EAST-ADL [23]. Figure 3.1 shows the EAST-ADL model of the BBW system.

Figure 3.1: The EAST-ADL model of the BBW system.

The TA model of the Anti-lock Braking System (ABS) component in the BBW system, without its timing and clock constraints, is depicted in Figure 3.2.

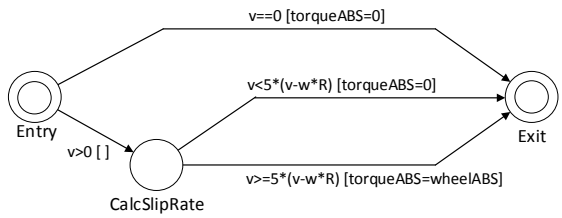

Figure 3.2: The TA description of the ABS function (without timing).

The TA models are analyzed and verified in Uppaal Port which also produces trace information constituting a path in the TA model. The trace information is parsed and transformed into executable test scripts which basically check behavioral conformance by verifying that the same set of states and transitions (or nodes and edges) are taken and visited at runtime and in the same order. To take into account timing properties, the time point at which each state is visited at runtime is timestamped. Doing so, it becomes possible to then check, for instance, how long it has taken to go from one state to another. Having such information as part of the test result reports, comparisons can be done against clock constraints and other timing annotations in the TA model, such as the model shown in Figure 3.3 (in which x represents a clock).

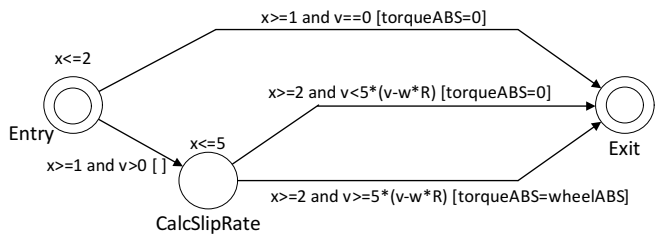

Figure 3.3: The TA description of the ABS function.

Part of a sample internal output (the actual output shown to the user is visualized and represented as HTML) of executing a test script showing the timestamps at each visited state is illustrated in Listing 3.1.

Listing 3.1: Sample internal output from a test script

```
Running tests...
----------------
Checking the behaviour of GlobalBrakeController
Transition 0:
        timestamp = 0:0 (1398176868:107184)
        state = 0
        rpm1 = 8
        rpm2 = 8
        reqTorque = 0
        whlTorque = 0
        v = 0
        rpm3 = 16
        rpm4 = 16
        R = 1

Transition 1:
        timestamp = 0:1 (1398176868:107185)
        state = 1
        rpm1 = 8
        rpm2 = 8
        reqTorque = 0
        whlTorque = 0
        v = 6
        rpm3 = 16
        rpm4 = 16
        R = 1

Transition 2:
        timestamp = 0:4 (1398176868:107188)
        state = 4294967295
        rpm1 = 8
        rpm2 = 8
        reqTorque = 0
        whlTorque = 0
        v = 6
        rpm3 = 16
        rpm4 = 16
        R = 1

....
```

This technique also enables to test other timing properties such as end-to-end response times in the systems. For instance, if there is a requirement on end-to-end response time from the moment that the brake pedal is pressed until the moment that the brake force is applied, the logged timestamp information can be used to calculate the actual end-to-end response time and determine if it is in compliance with the requirement or not. Listing 3.2 shows an excerpt of an Abstract Test Case (ATC) generated from TA models, including a sample requirement on end-to-end response time (specified as 'EndtoEnd = 3').

Listing 3.2: A sample ATC for BBW system

```
##############################################
# [TestCaseSpecification]
# atc1
#
# [RequirementSpecification]
# [Functional]
# E<> C2.VehicleSpeed_kmph_FL#==0 or C2.VehicleSpeed_kmph_FL#<5*(C2.
    VehicleSpeed_kmph_FL#-C2.WheelSpeed_rpm_FL#*C2.R/2) and C2.ABSTorque_FL
    #==0
#
# [RequirementSpecification]
# [Characteristics]
# EndtoEnd = 3
#
# [Purpose]
# -
#
# [Description]
# If VehicleSpeedIn == 0 or slip rate > ABSSlipRateThreshhold, then
    ABSBrakeTorqueOut shall be set to 0Nm.
#
# [PassCriteria]
# PASS if parameter values in the return signal are the same as the expected
     values.
#
# [Environment]
# Local host
#
# [Prerequisites]
# N/A
#
# [AbstractTestCaseDateGenerated]
# 06/05/2014
#
# [GeneratingToolAndVersion]
# UPPAAL PORT v0.48
#
##############################################
[SHORTNAME=atc1]
[ENDTOEND=3]

State:
( GlobalBrakeController.idle ABSFL.idle )
GlobalBrakeController.x=0 ABSFL.x=0 GlobalBrakeController.rpm1=0
      GlobalBrakeController.rpm2=0 GlobalBrakeController.reqTorque=0
      GlobalBrakeController.whlTorque=0 GlobalBrakeController.v=0
      GlobalBrakeController.rpm3=16 GlobalBrakeController.rpm4=16
      GlobalBrakeController.R=1 ABSFL.w=0 ABSFL.wheelABS=0 ABSFL.torqueABS=0
      ABSFL.v=0 ABSFL.R=1

Transitions:
  GlobalBrakeController.idle->GlobalBrakeController.Entry { reqTorque := 0,
       rpm1 := 8, rpm2 := 8, x := 0 }

State:
( GlobalBrakeController.Entry ABSFL.idle )
GlobalBrakeController.x=0 ABSFL.x=0 GlobalBrakeController.rpm1=8
      GlobalBrakeController.rpm2=8 GlobalBrakeController.reqTorque=0
      GlobalBrakeController.whlTorque=0 GlobalBrakeController.v=0
      GlobalBrakeController.rpm3=16 GlobalBrakeController.rpm4=16
      GlobalBrakeController.R=1 ABSFL.w=0 ABSFL.wheelABS=0 ABSFL.torqueABS=0
      ABSFL.v=0 ABSFL.R=1

Delay: 2

State:
( GlobalBrakeController.Entry ABSFL.idle)
GlobalBrakeController.x=2 ABSFL.x=2 GlobalBrakeController.rpm1=8
      GlobalBrakeController.rpm2=8 GlobalBrakeController.reqTorque=0
      GlobalBrakeController.whlTorque=0 GlobalBrakeController.v=0
      GlobalBrakeController.rpm3=16 GlobalBrakeController.rpm4=16
      GlobalBrakeController.R=1 ABSFL.w=0 ABSFL.wheelABS=0 ABSFL.torqueABS=0
      ABSFL.v=0 ABSFL.R=1

...
```

Our testing approach also provides for other interesting features such as defect localization. It means that when a problem is identified in the system, it can be observed in the test result report on the transition between which two states a deviation from the expected behavior has occurred. This way, it provides hints and information about the vicinity of a problem and helps with determining where the root cause of a problem could be in the system.

3.1 Overview of the Included Papers

The main contributions of this thesis are organized and included as a set of published papers as mentioned below. Other papers which can strengthen the contributions of the thesis, but are not included here, were mentioned at the beginning of the thesis; some of which, such as [19, 20], were briefly discussed and cited in the thesis.

- Paper A: Model-Based Trade-off Analysis of Non-Functional Requirements: An Automated UML-Based Approach

  Abstract: One common goal followed by software engineers is to deliver a product which satisfies the requirements of different stakeholders. Software requirements are generally categorized into functional and Non-Functional Requirements (NFRs). While NFRs may not be the main focus in developing some applications, there are systems and domains where the satisfaction of NFRs is even critical and one of the main factors which can determine the success or failure of the delivered product, notably in embedded systems. While the satisfaction of functional requirements can be decomposed and determined locally, NFRs are interconnected and have impacts on each other. For this reason, they cannot be considered in isolation and a careful balance and trade-off among them needs to be established. We provide a generic model-based approach to evaluate the satisfaction of NFRs taking into account their mutual impacts and dependencies. By providing indicators regarding the satisfaction level of NFRs in the system, the approach enables to compare different system design models and also identify parts of the system which can be good candidates for modification in order to achieve better satisfaction levels.

Contribution: I have been the initiator and main author of the paper.

- Paper B: Managing Timing Implications of Security Aspects in Model-Driven Development of Real-Time Embedded Systems

  Abstract: Considering security as an afterthought and adding security aspects to a system late in the development process has now been realized to be an inefficient and bad approach to security. The trend is to bring security considerations as early as possible in the design of systems. This is especially critical in certain domains such as real-time embedded systems. Due to different constraints and resource limitations that these systems have, the costs and implications of security features should be carefully evaluated in order to find appropriate ones which respect the constraints of the system. Model-Driven Development (MDD) and Component-Based Development (CBD) are two software engineering disciplines which help to cope with the increasing complexity of real-time embedded systems. While CBD enables the reuse of functionality and analysis results by building systems out of already existing components, MDD helps to increase the abstraction level, perform analysis at earlier phases of development, and also promotes automatic code generation. By using these approaches and including security aspects in the design models, it becomes possible to consider security from early phases of development and also identify the implications of security features. Timing issues are one of the most important factors for successful design of real-time embedded systems. In this paper, we provide an approach using MDD and CBD methods to make it easier for system designers to include security aspects in the design of systems and identify and manage their timing implications and costs. Among different security mechanisms to satisfy security requirements, our focus in this paper is mainly on using encryption and decryption algorithms and consideration of their timing costs to design secure systems.

  Contribution: I have been the initiator and main author of the paper.

- Paper C: Monitoring Capabilities of Schedulers in Model-Driven Development of Real-Time Systems

  Abstract: Model-driven development has the potential to reduce

the design complexity of real-time embedded systems by increasing the abstraction level, enabling analysis at earlier phases of development, and automatic generation of code from the models. In this context, capabilities of schedulers as part of the underlying platform play an important role. They can affect the complexity of code generators and how the model is implemented on the platform. Also, the way a scheduler monitors the timing behaviors of tasks and schedules them can facilitate the extraction of runtime information. This information can then be used as feedback to the original model in order to identify parts of the model that may need to be re-designed and modified. This is especially important in order to achieve round-trip support for model-driven development of real-time systems. In this paper, we describe our work in providing such monitoring features by introducing a second layer scheduler on top of the OSE real-time operating system's scheduler. The goal is to extend the monitoring capabilities of the scheduler without modifying the kernel. The approach can also contribute to the predictability of applications by bringing more awareness to the scheduler about the type of real-time tasks (i.e., periodic, sporadic, and aperiodic) that are to be scheduled and the information that should be monitored and logged for each type.

Contribution: I have been the initiator and main author of the paper. The implementation was majorly done by Naveed Ul Mustafa.

- Paper D: An Automated Round-trip Support Towards Deployment Assessment in Component-based Embedded Systems

  Abstract: Synergies between model-driven and component-based software engineering have been indicated as promising to mitigate complexity in development of embedded systems. In this work we evaluate the usefulness of a model-driven round-trip approach to aid deployment optimization in the development of embedded component-based systems. The round-trip approach is composed of the following steps: modelling the system, generation of full code from the models, execution and monitoring the code execution, and finally back-propagation of monitored values to the models. We illustrate the usefulness of the round-trip approach exploiting an industrial case-study from the telecom-domain. We use a code-generator that can realise different deployment strategies, as well as special monitoring code injected into the generated code, and

monitoring primitives defined at operating system level. Given this infrastructure we can evaluate extra-functional properties of the system and thus compare different deployment strategies.

Contribution: I have been the second author of the paper, responsible for writing and implementing the monitoring framework part.

- Paper E: Towards Accurate Monitoring of Extra-Functional Properties in Real-Time Embedded Systems

  Abstract: Management and preservation of Extra-Functional Properties (EFPs) is critical in real-time embedded systems to ensure their correct behavior. Deviation of these properties, such as timing and memory usage, from their acceptable and valid values can impair the functionality of the system. In this regard, monitoring is an important means to investigate the state of the system and identify such violations. The monitoring result is also used to make adaptation and re-configuration decisions in the system as well. Most of the works related to monitoring EFPs are based on the assumption that monitoring results accurately represent the true state of the system at the monitoring request time point. In some systems this assumption can be safe and valid. However, if in a system the value of an EFP changes frequently, the result of monitoring may not accurately represent the state of the system at the time point when the monitoring request has been issued. The consequences of such inaccuracies can be critical in certain systems and applications. In this paper, we mainly introduce and discuss this practical problem and also provide a solution to improve the monitoring accuracy of EFPs.

  Contribution: I have been the initiator and main author of the paper.

- Paper F: A Model-Based Testing Framework for Automotive Embedded Systems

  Abstract: Architectural models, such as those described in the EAST-ADL language, represent convenient abstractions to reason about automotive embedded software systems. To enjoy the fully-fledged advantages of reasoning, EAST-ADL models could benefit from a component-aware analysis framework that provides, ideally, both verification and model-based test-case generation capabilities. While different verification techniques have been developed

for architectural models, only a few target EAST-ADL. In this paper, we present a methodology for code validation, starting from EAST-ADL artifacts. The methodology relies on: (i) automated model-based test-case generation for functional requirements criteria based on the EAST-ADL model extended with timed automata semantics, and (ii) validation of system implementation by generating Python test scripts based on the abstract test-cases, which represent concrete test-cases that are executable on the system implementation. We apply our methodology to analyze the ABS function implementation of a Brake-by-Wire system prototype.

Contribution: I have been the second author in this paper, responsible for the parts regarding generation of executable test scripts (called as Concrete Test Case), and their execution on the platform and producing test result reports. I also participated and contributed to the development of the overall testing methodology.

- Paper G: Testing of Timing Properties in Real-Time Systems: Verifying Clock Constraints

  Abstract: Ensuring that timing constraints in a real-time system are satisfied and met is of utmost importance. There are different static analysis methods that are introduced to statically evaluate the correctness of such systems in terms of timing properties, such as schedulability analysis techniques. Regardless of the fact that some of these techniques might be too pessimistic or hard to apply in practice, there are also situations that can still occur at runtime resulting in the violation of timing properties and thus invalidation of the static analyses' results. Therefore, it is important to be able to test the runtime behavior of a real-time system with respect to its timing properties. In this paper, we introduce an approach for testing the timing properties of real-time systems focusing on their internal clock constraints. For this purpose, test cases are generated from timed automata models that describe the timing behavior of real-time tasks. The ultimate goal is to verify that the actual timing behavior of the system at runtime matches the timed automata models. This is achieved by tracking and time-measuring of state transitions at runtime.

  Contribution: I have been the initiator and main author of the paper.

Table 3.1 shows how the papers cover research goals and contributions of the thesis.

Thesis Paper	Contribution	Research Goal
A	C1	G1
B	C2	G2
C	C3 & C4	G3
D	C3	G2
E	C3 & C4	G3
F	C4	G3
G	C4	G3

Table 3.1: Mapping of papers to the research goals and contributions of the thesis

Chapter 4

Related Work

Property preservation: The importance of property preservation is acknowledged and discussed in different contexts in building software systems. In [24], the authors confirm that "the model can only be an approximation of the implementation w.r.t. the timing behavior. It is difficult to guarantee that the issuing time of an event in the implementation is exactly the same as that in the model". Based on this observation, in [24], they provide an approach and demonstrate that the real-time properties of the implemented system can be predicted from the properties of its (timed state/action sequences) model, when the time deviation is bounded. As an extension of this work, in [25] the authors introduce an approach for strengthening property preservation between model and implementation by imposing urgency on the execution of observable actions (than the execution of unobservable ones). They define the notion of distance "as a metric to express the strength of observable property preservation between model and implementation" [25]. Using this metric, they show that by applying the aforementioned approach and executing observable actions before unobservable ones, a smaller distance to the model than any other implementation of the same model can be obtained. From this aspect, their work can also be relevant and applicable for the issue of accuracy in monitoring EFPs that we have discussed in this thesis and the technique we introduced to tackle it [26].

One scenario where deviation of system properties can occur is in performing model transformations (horizontal or vertical). In this thesis we did not focus on property preserving model transformations and

consider it as a future work to complement the contributions of the thesis. There are, however, various works in the literature that discuss and provide solutions for this problem and for verifying the correctness of transformations. In [27], Vallecillo et al. discuss the importance of model transformation correctness and the issues related to specification and testing of transformations. They introduce the concept of *tract* as a generalization of model transformation contracts and a mechanism to specify and capture the expected behavior of a model transformation. Using tracts, they then generate test cases and perform testing of model transformations in a black-box fashion. Based on the concept of tracts, in [28], a static and white-box fault localization method is presented which helps to identify problematic rules in model transformations. In [29], an investigation on techniques based on finite model theory is done to show the use of algebraic co-limits (from category theory) in preservation of certain logical properties (consistency related) in model merging transformations. REFINER [30] is a tool consisting of a set of techniques for verification of behavioral transformations of formal models of concurrent systems. The tool can analyze transformations to determine if semantics of the input model and also given safety and liveness properties are preserved or not. Another work that can be consulted for semantic preserving model transformations is [31] by Mathias Hülsbusch et al. in which a direct bisimulation proof and borrowed context technique are used and compared as two different ways in order to show semantic preservation for a transformation.

NFR Framework: One of the fundamental works in addressing NFRs in development of systems, identifying their impacts and conflicts, and performing trade-off analysis on them, is the NFR Framework [32]. In this framework, NFRs are represented as *softgoals* which are to be *satisficed*. The notion of softgoal is used as a looser notion of goals to indicate the absence of a clear-cut criterion for satisfaction of non-functional requirements [33]. Similarly, the term satisfice is used to indicate that there is sufficient positive evidence and little negative evidence for the satisfaction of an NFR. An NFR is considered *unsatisficeable* when the opposite of the above condition holds. Development techniques for achieving NFRs are defined as *operationalization* which include (but are not limited to) operations, functions and data. Besides NFR softgoals and operationalizating softgoals, NFR framework introduces *claim softgoals* which convey the rationale and argument for or

against a design decision. In refining and decomposing softgoals, the relationships between them are established in the form of 'AND' and 'OR' contributions. An 'AND' contribution means that all the sub-softgoals are needed in order to achieve a softgoal at a higher level in the hierarchy. The structure that is produced as the result of the decomposition and refinement process is called Softgoal Interdependency Graph (SIG). The NFR UML profile that we introduced in this thesis is inspired by the NFR Framework and the concept of *feature* in our NFR profile is similar to what NFR Framework calls *operationalization*. One major difference, though, is that NFR Framework can only support a qualitative form of analysis of NFRs while our profile along with its automated analysis mechanism enables to do so in a quantitative manner. Moreover, the concepts we have suggested as part of the NFR Profile enable to capture more detailed information for each NFR in the system, such as the rationale behind having a requirement as well as numerical properties such as deviation indicator value that are calculated as the result of automated analysis of NFRs.

MARTE: The UML profile for Modeling and Analysis of Real-Time and Embedded Systems (MARTE) [34] is one of the recent and major efforts on modeling real-time embedded systems and their extra-functional properties. It was introduced as the successor of UML profile for Schedulability, Performance, and Time (SPT). MARTE includes concepts and semantics for UML-based description of real-time and embedded systems. The core concepts in MARTE are categorized in two parts: modeling and analysis. The intent in the analysis part is not to define new analysis techniques, but to provide a framework to annotate models with the necessary extra-functional properties and information in order to support different kinds of analyses in the real-time domain, particularly performance and schedulability. One of the main characteristics of MARTE is that it provides a common way to model both the hardware and software aspects of real-time systems. This improves the communication between developers and helps to include both hardware and software characteristics in making predication and analysis of the systems. In this thesis, we did not deal with how EFPs can be specified and represented, for which MARTE can serve as one solution. There is, however, the potential to use and gain from the capabilities of MARTE in our suggested solutions. For instance, the concepts for modeling EFPs in MARTE can be used together with our NFR profile to annotate and

include in the model of NFRs the EFPs of different *feature* elements that are considered for satisfying the NFRs.

Real-Time Specification for Java (RTSJ): Many of the operating systems and also programming languages today provide support for measuring the CPU time that a runnable entity (i.e., thread, etc.) consumes to perform its function. However, the monitoring facilities and event handling mechanisms provided by these platforms are not usually integrated with their scheduling facilities [35]. As a result, the platform cannot enforce and ensure real-time properties of threads such as their allowed execution times and deadlines. This can lead to the violation of the result of timing analyses performed before runtime. Real-Time Specification for Java (RTSJ) is an attempt to integrate scheduling of threads with the execution time monitoring facilities and enforce execution budgets on them. By monitoring the execution times of threads and comparing them with the specified timing requirements, it can detect and ensure that a thread which is about to exceed its execution time budget does not impair the execution of other tasks as expected and predicted through schedulability analysis [35].

The concept of the second layer scheduler that we introduced and implemented (in paper C) to enable detailed monitoring of real-time tasks is similar to the main idea behind RTSJ. Our approach, however, provides more types of monitoring information such as on periodicity of tasks, deadline misses, execution time overruns. It also generates extensive log information which can be analyzed for different purposes such as to detect that a task is getting closer and closer to missing its deadline in each execution and take some measures to prevent it from missing its deadline before it occurs. In addition, our solution enables to configure and use different scheduling policies in the scheduler.

Ada Ravenscar: Ravenscar profile for Ada which was introduced in the 8th International Real-Time Ada Workshop (IRTAW) [36] is a subset of the tasking model in Ada. It is defined to provide the level of predictability and determinism that is needed in safety-critical hard real-time systems by removing constructs that contain non-deterministic behavior and implementation dependancies [37, 38]. It offers features that cover the programming needs of statically defined real-time systems. Ravenscar profile enables specification of a safety-critical system in a way to be certifiable to the highest integrity levels. Such a verifiable system

is achieved by:

- providing a task concurrency model for real-time systems along with several concurrency-related checks,
- enforcement of constraints that are required to enable and preserve results of schedulability analysis and other types of static analysis techniques,
- defining bounds on memory usage and allowing only a static task set (number of tasks are fixed),
- enabling creation of small and highly efficient runtime implementations (by reducing the size and complexity of the required runtime system) [37].

These features facilitate temporal verification of Ravenscar programs and gaining confidence in its behavior. In [39], several mechanisms in Ada Ravenscar profile that contribute to the preservation of timing properties, such as WCET and period/MIAT, are introduced and discussed.

CHESS project: The CHESS project [40] which stands for Composition with Guarantees for High-integrity Embedded Software Components Assembly aims to provide model-based solutions for addressing extra-functional concerns in embedded systems while guaranteeing correctness of the system at runtime. This is done by modeling systems using a cross-domain UML profile called CHESS ML which adopts from the modeling concepts of MARTE [34] (for modeling of extra-functional properties) and SysML [41] (for high-level modeling of requirements) UML profiles, and also extends some of the concepts defined in these profiles to cover the modeling needs of telecommunication, space, railway, and to some extent automotive domains. In CHESS, Model-Driven Architecture (MDA) methodology that is recommended by Object Management Group (OMG) is used [42]. Different types of analyses are performed at different levels and throughout the transformation chain for code generation to ensure correctness of the design. The idea here is to provide a read-only Platform-Specific Model (PSM) to the user, and also make the manual editing of the code unnecessary. This is important in order to maintain design consistency. As a consequence, the results of the analyses are propagated back to the Platform-Independent Model

(PIM), so that the user can identify parts of the model that need to be modified in order to achieve the desired behavior.

CHESS project is basically defined as an extension of the ASSERT (Automated proof-based System and Software Engineering for Real-Time systems) [43] project following two main goals: to establish correctness as early as possible and to actively preserve it throughout the whole development process down to deployment and execution. Towards these goals, ASSERT adopted an MDE methodology targeting high-integrity software systems that are used in space domain and was based on a DSL. CHESS extended the approach covering other types of systems from telecommunication and railway (and partially automotive) domains. The modeling language that was used in CHESS, called CHESSML, was based on UML profiling approach consisting of concepts from UML, SysML and MARTE. Some of the contributions of this thesis have been formulated in the scope of the CHESS project (mentioned in the respective papers).

Others: In [44] by Sentilles, a framework for management of extra-functional properties in component models of embedded systems is offered. The framework enables specification and attachment of multi-valued and context-aware EFPs to architectural elements of component models. In this work, it is also highlighted and emphasized that different values for an EFP can exist which originate from different sources. Moreover, by gaining more information and knowledge about a system, refined and more accurate values for an EFP can be considered along the development of the system. As part of this work, ProCom component model is also introduced. In summary, it focuses mainly on capturing and representation of EFPs in component models of embedded systems.

The work done by Ciccozzi in [45] is a model-driven approach towards achieving preservation of properties. It introduces model-driven techniques such as full-code generation from (analyzed) system models and also a back propagation mechanism to facilitate preservation of properties. The back propagation mechanism enables to establish a feedback loop between monitoring results at runtime and system models. This way, it helps to inform designers about actual values of properties at runtime versus their expected values specified in the model. This work assumes that the platform for which code is generated has necessary mechanisms for monitoring, enforcement and basically preservation of EFPs. Moreover, it does not deal with how balance between EFPs can

be established, except its suggested round-trip support (which is common between that work and this thesis) which can be used as a means towards establishing such a balance. In that work however, the focus in the round-trip solution is on the model-based techniques to propagate back monitored information to the design models, while in this thesis, we focused on the monitoring part and the role of the platform in the round-trip chain.

REMES which is a Resource Model for Embedded Systems is introduced and applied in [46]. REMES enables modeling and reasoning about the functional and extra-functional behaviors of a system and its components. It introduces a formal language (a state-machine based behavioral language) for modeling resources. The main intention with REMES is to express resource usage and perform usage analysis. In a REMES model, resources are considered as global quantities of finite size. The analysis of resources is based on multi-priced timed automata and enables to solve, for instance, feasibility, trade-off and optimal/worst-case resource analysis problems [47].

An example of works that deal with EFPs at the hardware and chip level is [48] by Hatami which provides a method for analysis and prediction of hardware related EFPs at early design phases. It takes into account various hardware parameters such as voltage, frequency, and a few others to determine chip level EFPs such as dynamic energy for a transistor. In [49] by Sapienza, an approach is provided for making decisions based on EFPs for optimal partitioning of an embedded system into software and hardware parts.

Chapter 5

Conclusion and Future Directions

In this thesis, we addressed the important issue of preservation of EFPs in embedded systems. We started by discussing the relationship between NFRs and EFPs which is necessary in understanding how constraints over EFPs form and where they originate from. It is based on such a relationship that it then becomes possible to talk about if the EFPs of a system element or component is acceptable in the context of that system or not. In embedded systems where such EFP constraints can be determinant in success or failure of the product, ensuring that those constraints are not violated and EFPs are preserved within their constraints is crucial.

For modeling NFRs and performing trade-off analysis among them, we introduced a UML profile which enables capturing of NFRs in a generic way and automatic analysis of their trade-offs using model transformation techniques. In this step, mutual impacts and conflicts among NFRs are understood and they can then be balanced to achieve an acceptable overall satisfaction level for all the NFRs before continuing with the rest of the development process. As the next step, to build a system which respects the constraints defined over its EFPs, we introduced an approach for balancing security versus timing constraints which enables to produce a component model of the system including added security features while respecting and operating within its timing constraints. Although in the approach we only focused on security and timing prop-

erties, the suggested solution has the potential to be adapted and used for other EFPs as well.

As mentioned before, the ultimate goal in designing a software product is that the system performs as expected at runtime and when it is in operation. However, regardless of which development method is used and the amount of analysis performed, situations may still occur at runtime that lead to the violation of EFPs. To monitor for such cases for timing properties, we suggested the second layer scheduler concept. It basically adds to the execution platform the necessary mechanism for monitoring of timing events. Of course, adding monitoring features has its own overhead. In terms of the second layer scheduler, the solution we have suggested adds the monitoring capabilities without modifying the kernel of the OS, which is particularly interesting in cases where, for instance, the source code is not available, legacy systems, or when more flexibility and portability is desired. By including the monitoring features as part of the OS kernel and scheduler, the overhead can be reduced to some extent. In the context of model driven development, we also demonstrated how the collected monitoring information about EFPs can be used to propagate back to the models in order to identify inconsistencies. The models can then be modified and new code can be generated until the desired set of EFPs value at runtime is achieved. Such a round-trip support is particularly useful in optimizing the system with respect to EFPs, which deserves a separate study and investigation as a future work.

When a system is built, the next step is to test it before shipping it to customers. For testing timing properties, we also introduced a model-based testing framework which enables testing the actual behavior of the system at runtime against its desired behavior captured in the form of timed automata models. This was achieved in our approach by automatic generation and execution of test cases and use of timestamps at runtime for each state transition. As a future extension, the approach may also be well modified and used for testing of other EFPs such as memory usage, if memory constraints are also captured as part of the model. For instance, analogous to a timed automata model, if in a state machine model the information on allowed memory usage in each state is specified, the approach can easily be modified to check the memory consumption at runtime at each state transition and verify if it matches the specified constraints in the model. One point to note here with respect to testing in general is that testing and passing test cases does

not necessarily guarantee the absence of errors and bugs in a system, but serves as a means to gain more confidence in the quality of the product that is developed.

In this thesis, we also investigated the issue of the accuracy of collected monitoring information and demonstrated a technique for mitigating this issue. Accuracy plays an important role, for example, when the collected information is used to make decisions such as for runtime adaptation. If the information is not accurate (or fresh and up-to-date), it can lead to taking wrong decisions, which in turn might have drastic consequences, for instance, in safety critical systems such as a pacemaker. In the thesis, mainly single core platforms were considered and it would be interesting as another future work to investigate monitoring and testing of EFPs in multicore systems which would have their unique challenges. For instance, there could be a requirement on end-to-end response time of a set of tasks which are allocated and run on different cores. Extending the monitoring and testing methods introduced in this work to such scenarios would be another future direction and extension of this work.

As mentioned in the thesis, an EFP may well be refined and translated into one or more fine-grained EFPs at lower abstraction levels. The refinement and granularity level at which preservation of an EFP is applied can depend on several factors such as need and interest to preserve an EFP at a certain level and also monitoring feasibility at that level. For instance, if there is a composite component with a constraint on its maximum execution time, preservation of its execution time property may be enforced at the level of the composite component itself or at the level of its child components. It should, however, be noted that in the latter case, property preservation can be more restrictive, which might actually be necessary in certain systems and contexts. On the other hand, in the former case there should be a mechanism to be able to monitor the execution time of the composite component or derive it from the execution times of its child components which are feasible to monitor.

The solutions introduced in this thesis contribute to preserving EFPs at different abstraction levels and development phases, particularly in the context of model-based development. Therefore, together they can form and be part of a methodology and act as a set of design techniques for building a system with property preservation considerations and support, from the requirements analysis phase down to its deployment on the

platform, and execution. Application of such a methodology as a whole and its validation against one single system and use-case is planned as another future work. Overall, we believe that the solutions proposed in this thesis can help and serve as a set of means in building embedded systems with better quality assurance.

Bibliography

[1] Martin Glinz. On Non-Functional Requirements. In *15th IEEE International Requirements Engineering Conference*, pages 21–26, New Delhi, India, October 2007.

[2] S. Heath. Embedded Systems Design. EDN series for design engineers, ISBN: 9780750655460. Newnes, 2003.

[3] Thomas Henzinger and Joseph Sifakis. The Embedded Systems Design Challenge. In Jayadev Misra, Tobias Nipkow, and Emil Sekerinski, editors, *FM 2006: Formal Methods*, volume 4085 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2006.

[4] Mehrdad Saadatmand, Antonio Cicchetti, and Mikael Sjödin. Toward Model-Based Trade-off Analysis of Non-Functional Requirements. In *38th Euromicro Conference on Software Engineering and Advanced Applications(SEAA)*, September 2012.

[5] Mehrdad Saadatmand. *Satisfying Non-Functional Requirements in Model-Driven Development of Real-Time Embedded Systems, Licentiate Thesis.* Number 150. Mälardalen University, May 2012.

[6] Mehrdad Saadatmand, Antonio Cicchetti, and Mikael Sjödin. UML-Based Modeling of Non-Functional Requirements in Telecommunication Systems. In *The Sixth International Conference on Software Engineering Advances (ICSEA 2011)*, Barcelona, Spain, October 2011.

[7] Lawrence Chung and Julio Cesar Prado Leite. Conceptual Modeling: Foundations and Applications. chapter On Non-Functional

Requirements in Software Engineering, pages 363–379. Springer-Verlag, Berlin, Heidelberg, 2009.

[8] Luiz Marcio Cysneiros and Julio Cesar Sampaio do Prado Leite. Non-functional requirements: From elicitation to conceptual models. In *IEEE Transactions on Software Engineering*, volume 30, pages 328–350, 2004.

[9] Bran Selic. The Pragmatics of Model-Driven Development. *IEEE Software*, 20:19–25, September 2003.

[10] M. Torngren, DeJiu Chen, and I. Crnkovic. Component-based vs. model-based development: a comparison in the context of vehicular embedded systems. In *Software Engineering and Advanced Applications, 2005. 31st EUROMICRO Conference on*, pages 432 – 440, aug.-3 sept. 2005.

[11] B.W. Boehm and P.N. Papaccio. Understanding and controlling software costs. *Software Engineering, IEEE Transactions on*, 14(10):1462–1477, 1988.

[12] Barry Boehm and Victor R. Basili. Software Defect Reduction Top 10 List. *Computer*, 34(1):135–137, January 2001.

[13] G. Tassey. The economic impacts of inadequate infrastructure for software testing. Technical report, National Institute of Standards and Technology, May, 2002.

[14] O. Florescu, Jinfeng Huang, J. Voeten, and H. Corporaal. Strengthening Property Preservation in Concurrent Real-Time Systems. In *Embedded and Real-Time Computing Systems and Applications, 2006. Proceedings. 12th IEEE International Conference on*, pages 106–109, 2006.

[15] Oana Florescu, Jinfeng Huang, Jeroen Voeten, and Henk Corporaal. Towards Stronger Property Preservation in Real-Time Systems Synthesis (Technical Report). `http://repository.tue.nl/710999`, 2006.

[16] S.E. Chodrow, F. Jahanian, and M. Donner. Run-time monitoring of real-time systems. In *Real-Time Systems Symposium, 1991. Proceedings., Twelfth*, pages 74 –83, dec 1991.

[17] Mehrdad Saadatmand, Antonio Cicchetti, and Mikael Sjödin. Design of adaptive security mechanisms for real-time embedded systems. In *Proceedings of the 4th international conference on Engineering Secure Software and Systems*, ESSoS'12, pages 121–134, Eindhoven, The Netherlands, 2012. Springer-Verlag.

[18] Schrödinger's cat Experiment. `http://www.http://en.wikipedia.org/wiki/Schr%C3%B6dinger%27s_cat`, Accessed: December 2014.

[19] Mehrdad Saadatmand and Mikael Sjodin. On Combining Model-Based Analysis and Testing. In *Information Technology: New Generations (ITNG), 2013 Tenth International Conference on*, pages 260–266, Las Vegas, NV, USA, April 2013.

[20] Mehrdad Saadatmand and Sahar Tahvili. A Fuzzy Decision Support Approach for Model-Based Tradeoff Analysis of Non-Functional Requirements. In *12th International Conference on Information Technology : New Generations (ITNG)*, Las Vegas, Nevada, USA, April 2015.

[21] Srivaths Ravi, Anand Raghunathan, Paul Kocher, and Sunil Hattangady. Security in Embedded Systems: Design Challenges. *ACM Trans. Embed. Comput. Syst.*, 3(3):461–491, August 2004.

[22] Mehrdad Saadatmand and Antonio Cicchetti. Mapping of State Machines to Code: Potentials and Challenges. In *The Ninth International Conference on Software Engineering Advances (ICSEA)*, pages 247–251, Nice, France, October 2014.

[23] EAST-ADL Specification. `http://www.atesst.org`, Accessed: December 2014.

[24] Jinfeng Huang, Jeroen Voeten, and Marc Geilen. Real-time Property Preservation in Concurrent Real-time Systems. In *In: Proc. of 10th International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA)*, 2004.

[25] O. Florescu, Jinfeng Huang, J. Voeten, and H. Corporaal. Strengthening Property Preservation in Concurrent Real-Time Systems. In *Embedded and Real-Time Computing Systems and Applications, 2006. Proceedings. 12th IEEE International Conference on*, pages 106–109, 2006.

[26] Mehrdad Saadatmand and Mikael Sjodin. Towards Accurate Monitoring of Extra-Functional Properties in Real-Time Embedded Systems. In *Software Engineering Conference (APSEC), 2012 19th Asia-Pacific*, pages 338–341, Dec 2012.

[27] Antonio Vallecillo, Martin Gogolla, Loli Burgueño, Manuel Wimmer, and Lars Hamann. Formal Specification and Testing of Model Transformations. In Marco Bernardo, Vittorio Cortellessa, and Alfonso Pierantonio, editors, *Formal Methods for Model-Driven Engineering*, volume 7320 of *Lecture Notes in Computer Science*, pages 399–437. Springer Berlin Heidelberg, 2012.

[28] L. Burgueno, J. Troya, M. Wimmer, and A. Vallecillo. Static Fault Localization in Model Transformations. *Software Engineering, IEEE Transactions on*, PP(99):1–1, 2015.

[29] Mehrdad Sabetzadeh, Shiva Nejati, Marsha Chechik, and Steve Easterbrook. Reasoning about Consistency in Model Merging. In *Proceedings of 3rd Workshop on Living With Inconsistency in Software Development (Co-located with ASE2010)*, Antwerp, Belgium, September 2010.

[30] Anton Wijs and Luc Engelen. REFINER: Towards Formal Verification of Model Transformations. In JuliaM. Badger and KristinYvonne Rozier, editors, *NASA Formal Methods*, volume 8430 of *Lecture Notes in Computer Science*, pages 258–263. Springer International Publishing, 2014.

[31] Mathias Hülsbusch, Barbara König, Arend Rensink, Maria Semenyak, Christian Soltenborn, and Heike Wehrheim. Showing Full Semantics Preservation in Model Transformation - A Comparison of Techniques. In Dominique Méry and Stephan Merz, editors, *Integrated Formal Methods*, volume 6396 of *Lecture Notes in Computer Science*, pages 183–198. Springer Berlin Heidelberg, 2010.

[32] Lawrence Chung, Brian A. Nixon, Eric Yu, and John Mylopoulos. *Non-Functional Requirements in Software Engineering*, volume 5 of *International Series in Software Engineering*. Springer, 1999.

[33] John Mylopoulos, Lawrence Chung, and Eric Yu. From object-oriented to goal-oriented requirements analysis. *Commun. ACM*, 42:31–37, January 1999.

[34] MARTE specification. `http://www.omgmarte.org`, Accessed: December 2014.

[35] Andy J. Wellings, Gregory Bollella, Peter C. Dibble, and David Holmes. Cost Enforcement and Deadline Monitoring in the Real-Time Specification for Java. In *7th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)*, pages 78–85. IEEE Computer Society, 12-14 May 2004.

[36] Lars Asplund, B. Johnson, Kristina Lundqvist, and Alan Burns. Session Summary: The Ravenscar Profile and Implementation Issues. ACM Press, July 1999.

[37] Alan Burns, Brian Dobbing, and Tullio Vardanega. Guide for the use of the Ada Ravenscar Profile in high integrity systems. *Ada Lett.*, XXIV:1–74, June 2004.

[38] Kristina Lundqvist and Lars Asplund. A Ravenscar-Compliant Run-time Kernel for Safety-Critical Systems*. *Real-Time Systems*, 24:29–54, January 2003.

[39] Enrico Mezzetti, Marco Panunzio, and Tullio Vardanega. Preservation of Timing Properties with the Ada Ravenscar Profile. In Jorge Real and Tullio Vardanega, editors, *Reliable Software Technologies – Ada-Europe 2010*, volume 6106 of *Lecture Notes in Computer Science*, pages 153–166. Springer Berlin Heidelberg, 2010.

[40] CHESS Project: Composition with Guarantees for High-integrity Embedded Software Components Assembly. `http://chess-project.ning.com/`, Accessed: December 2014.

[41] OMG SysML Specifcation. `http://www.sysml.org/specs.htm`, Accessed: December 2014.

[42] Model-Driven Architecture (MDA). `http://www.omg.org/mda/`, Accessed: December 2014.

[43] ASSERT Project: Automated proof-based System and Software Engineering for Real-Time systems. `http://http://cordis.europa.eu/projects/rcn/71564_en.html/`, Accessed: December 2014.

[44] Séverine Sentilles. *Managing Extra-Functional Properties in Component-Based Development of Embedded Systems.* PhD thesis, Mälardalen University, Västerås, Sweden, June 2012.

[45] Federico Ciccozzi. *From models to code and back: A round-trip approach for model-driven engineering of embedded systems.* PhD thesis, Mälardalen University, Västerås, Sweden, January 2014.

[46] Aneta Vulgarakis. *A Resource-Aware Framework for Designing Predictable Component-Based Embedded Systems.* PhD thesis, Mälardalen University, June 2012.

[47] C. Seceleanu, A. Vulgarakis, and P. Pettersson. REMES: A Resource Model for Embedded Systems. In *Engineering of Complex Computer Systems, 2009 14th IEEE International Conference on*, pages 84–94, 2009.

[48] Nadereh Hatami Mazinani. *Multi-level analysis of non-functional properties (PhD Thesis).* PhD thesis, Universität Stuttgart, Holzgartenstr. 16, 70174 Stuttgart, 2014.

[49] Gaetana Sapienza. *Multiple Property-Based Partitioning for Embedded Applications, Licentiate Thesis.* Number 176. Mälardalen University, May 2014.