A Model-driven Development Approach with Temporal Awareness for Vehicular Embedded Systems

Alessio Bucaioni

Alessio Bucaioni is an industrial Ph.D. student at the School of Innovation, Design and Engineering at Mälardalen University, Sweden and Arcticus Systems AB, Sweden. He holds a Licentiate Degree in Computer Science and Engineering from Mälardalen University.

Alessio’s research interests include, but are not limited to, model driven engineering and software engineering. Currently, Alessio is focusing on model driven development of vehicular embedded systems.
A MODEL-DRIVEN DEVELOPMENT APPROACH WITH TEMPORAL AWARENESS FOR VEHICULAR EMBEDDED SYSTEMS

Alessio Bucaioni

2017

School of Innovation, Design and Engineering
A MODEL-DRIVEN DEVELOPMENT APPROACH WITH
TEMPORAL AWARENESS FOR VEHICULAR EMBEDDED SYSTEMS

Alessio Bucaioni

Akademisk avhandling

som för avläggande av teknologie doktorsexamen i datavetenskap vid
Akademin för innovation, design och teknik kommer att offentligen försvaras
fredagen den 12 januari 2018, 10.15 i Gamma, Mälardalens högskola, Västerås.

Fakultetsopponent: Professor Matthias Tichy, Ulm University
Abstract
Considering the ubiquitousness of software in modern vehicles, its increased value and development cost, an efficient software development became of paramount importance for the vehicular domain. It has been identified that early verification of non-functional properties of vehicular embedded software such as, timing, reliability and safety, is crucial to efficiency. However, early verification of non-functional properties is hard to achieve with traditional software development approaches due to the abstraction and the lack of automation of these methodologies.

This doctoral thesis aims at improving efficiency in vehicular embedded software development by minimising the need for late, expensive and time consuming software modifications with early design changes, identified through timing verification, which usually are cheaper and faster. To this end, we introduce a novel model-driven approach which exploits the interplay of two automotive-specific modelling languages for the representation of functional and execution models and defines a suite of model transformations for their automatic integration.

Starting from a functional model (expressed by means of EAST-ADL), all the execution models (expressed by means of the Rubus Component Model) entailing unique timing configurations are derived. Schedulability analysis selects the set of the feasible execution models with respect to specified timing requirements. Eventually, a reference to the selected execution models along with their analysis results is automatically created in the related functional model to allow the engineer to investigate them.

The main scientific contributions of this doctoral thesis are i) a metamodel definition for the Rubus Component Model, ii) an automatic mechanism for the generation of Rubus models from EAST-ADL, iii) an automatic mechanism for the selection and back-propagation of the analysis results and related Rubus models to design level and iv) a compact notation for visualising the selected Rubus models by means of a single execution model.
To Lorenzo and Gaia
Abstract

Considering the ubiquitousness of software in modern vehicles, its increased value and development cost, an efficient software development became of paramount importance for the vehicular domain. It has been identified that early verification of non functional properties of vehicular embedded software such as, timing, reliability and safety, is crucial to efficiency. However, early verification of non functional properties is hard to achieve with traditional software development approaches due to the abstraction and the lack of automation of these methodologies.

This doctoral thesis aims at improving efficiency in vehicular embedded software development by minimising the need for late, expensive and time consuming software modifications with early design changes, identified through timing verification, which usually are cheaper and faster. To this end, we introduce a novel model-driven approach which exploits the interplay of two automotive-specific modelling languages for the representation of functional and execution models and defines a suite of model transformations for their automatic integration. Starting from a functional model (expressed by means of EAST-ADL), all the execution models (expressed by means of the Rubus Component Model) entailing unique timing configurations are derived. Schedulability analysis selects the set of the feasible execution models with respect to specified timing requirements. Eventually, a reference to the selected execution models along with their analysis results is automatically created in the related functional model to allow the engineer to investigate them.

The main scientific contributions of this doctoral thesis are i) a metamodel definition for the Rubus Component Model, ii) an automatic mechanism for the generation of Rubus models from EAST-ADL, iii) an automatic mechanism for the selection and back-propagation of the analysis results and related Rubus models to design level and iv) a compact notation for visualising the selected Rubus models by means of a single execution model.
Sammanfattning

Eftersom programvaran är allestädes närvarande i fordon är kostnadseffektiv mjukvaruutveckling för fordon avgörande. Det anses att tidig verifiering av icke-funktionella egenskaper hos fordonsmjukvaran, såsom till exempel timing, pålitlighet och säkerhet, som avgörande för kostnadseffektiv mjukvaruutveckling. Emellertid är tidig verifiering av icke-funktionella egenskaper svårt att uppnå med traditionella mjukvaruutvecklingsmetoder på grund av bristande abstraktion och automatisering.

Denna avhandling syftar till att förbättra effektiviteten hos mjukvaruutveckling för fordon genom att ersätta behovet av sena, dyra och tidskrävande mjukvaruändringar med tidiga, billiga och snabba designändringar som drivs av timingverifiering. Vi introducerar en modelldriven metod för utveckling av inbyggd mjukvara för fordon på plattformar med en eller flera kärnor. Metoden guidar utvecklaren till acceptabla lösningar som identifieras genom schemanläggningsanalys.
Acknowledgements

Runners say that you can learn everything you ever wanted to know about yourself in 26.2 miles. In my case, I learnt it in 4 years. Standing by the finish line, I can not think of a more frightening yet terrific experience as the one I am about to finish. However, as cliche as it could sound, I could have not completed this journey without the support of several individuals. The following words are my humble attempt in thanking all of them, knowing that these few words will never suffice the support these people provided me.

First and foremost, I would like to express my deepest gratitude and esteem to my supervisors Mikael Sjödin, Antonio Cicchetti, Federico Ciccozzi and Saad Mubeen whose guidance has been invaluable in my journey. I am especially grateful as they became friends besides being mentors and colleagues. I would like to thank Kurt-Lennart Lundbäck on behalf of Arcticus Systems for providing me with the best workplace a young researcher can wish for. I would like to thank my opponent Professor Matthias Tichy and the examining committee members Dr Henrik Lönn, Associate Professor De-Jiu Chen and Associate Professor Tomas Bures for dedicating me some of their precious time. The workplace at MDH is unique under several aspects. However, the human side of this organisation is just unbeatable. For this, I would like to thank all my friends and co-workers from the department for having provided me with help, inspiration and fun. I will never thank my family enough for their unconditional love. It took me almost thirty years for realising my family is the best and most authentic part of me. I would like to thank Angelika for not giving up on me. I would like to thank my friends, the family I chose, for always standing by my side without never doubting our friendship. I would like to thank my grandfathers Vincenzo and Terzilio, my grandmother Elisa, my aunts Ines and Nunziata and my uncle Pasquale for protecting me from above. Lastly, I would like to thank the One above us all for always answering my prayers and giving me the strength to not throw in the towel.

Alessio Bucaioni
Västerås, January, 2018
List of Publications

Publications Included in this Doctoral Thesis


1The included publications are reformatted to comply with the doctoral thesis printing format.
Additional Publications not Included in the Thesis


# Contents

## I Thesis

### 1 Introduction

1.1 Thesis Contribution ........................................... 5  
1.2 Thesis Outline .............................................. 5  

### 2 Preliminaries

2.1 Embedded Systems ............................................. 7  
2.2 Schedulability Analysis ....................................... 7  
2.3 Model Driven Engineering ..................................... 8  
2.4 EAST-ADL ...................................................... 8  
2.5 Rubus Component Model ....................................... 9  
2.6 Uncertainty ................................................... 10  

### 3 Research Goal, Challenges and Contributions

3.1 Research Goal .................................................. 11  
3.2 Research Challenges .......................................... 12  
3.3 Research Contributions ....................................... 13  
3.4 Papers Contribution .......................................... 22  
3.4.1 Paper A ................................................. 23  
3.4.2 Paper B ................................................. 23  
3.4.3 Paper C ................................................. 24  
3.4.4 Paper D ................................................. 25  
3.4.5 Paper E ................................................. 26  

### 4 Research Methodology and Validation

4.1 Research Methodology ........................................ 27  
4.2 Validation ................................................... 28  

xv
5 Related Works 31

6 Conclusions and Future Works 35

Bibliography 37

II Included Papers 43

7 Paper A:
A Metamodel for the Rubus Component Model: Extensions for Timing and Model Transformation from EAST-ADL 45
7.1 Introduction ........................................ 47
7.2 Background and related work .................. 49
7.2.1 MDE and CBSE in the Automotive Domain .... 49
7.2.2 End-to-end timing models and analyses ...... 52
7.2.3 Paper contributions ............................ 54
7.3 Providing a metamodel for RCM ............... 55
7.4 DL2RCM model transformation ................. 60
7.5 Application to the steer-by-wire system ....... 67
7.6 Evaluation and discussion ....................... 69
7.7 Conclusions and future work .................. 73
Bibliography ........................................ 75

8 Paper B:
Anticipating Implementation-Level Timing Analysis for Driving Design-Level Decisions in EAST-ADL 79
8.1 Introduction ........................................ 81
8.2 Related Work .................................... 82
8.3 A Running Example: the Steer-by-wire System .... 83
8.4 Applying the methodology ...................... 84
8.4.1 Transformation Phase ......................... 85
8.4.2 End-to-end Delay Analysis Phase ............. 89
8.4.3 Filtering and Propagation Phases ............. 90
8.5 Discussion ....................................... 91
8.6 Conclusion ....................................... 92
Bibliography ........................................ 93
## 9 Paper C:
Handling Uncertainty in Automatically Generated Implementation Models in the Automotive Domain  
9.1 Introduction ........................................... 97  
9.2 Background ........................................... 99  
9.3 Motivating Scenario ................................... 100  
9.4 u-Rubus ................................................ 102  
9.5 Discussion ............................................ 105  
9.6 Related Work .......................................... 109  
9.7 Conclusion and Future Work .......................... 114  
Bibliography ............................................ 115  

## 10 Paper D:  
Technology-preserving transition from single-core to multi-core in modelling vehicular systems  
10.1 Introduction .......................................... 121  
10.2 The Rubus Component Model ......................... 123  
10.3 Related Work .......................................... 124  
10.4 Extending Rubus Component Model for Multi-core  
.............................................................. 125  
10.5 Modelling the Brake-by-wire System ................ 129  
10.6 Lesson Learned ........................................ 133  
10.7 Conclusion and Future Work .......................... 134  
Bibliography ............................................ 137  

## 11 Paper E:  
A Model-based Approach for Vehicular Systems  
11.1 Introduction .......................................... 145  
11.2 Background ........................................... 147  
11.2.1 EAST-ADL ......................................... 148  
11.2.2 RCM ................................................ 150  
11.2.3 Timing Analysis .................................... 153  
11.2.4 Paper Contributions in Relation with Authors’ Previous Work  
.............................................................. 155  
11.3 The MoVES Methodology: why? ....................... 156  
11.4 MoVES for Timing .................................... 158  
11.4.1 FDA2RCM ......................................... 160  
11.4.2 HDA2RCM ......................................... 163  
11.4.3 MERGE and A2RCM ............................... 165
11.4.4 ALLOCATION ................................. 168
11.4.5 BP ........................................ 169
11.5 Case Study ................................. 171
11.6 Discussion and Validation ................. 179
11.7 Related Work .............................. 183
   11.7.1 Development of vehicular embedded systems 183
   11.7.2 Development of embedded systems ............. 184
   11.7.3 Support for design-space exploration ............ 185
11.8 Conclusion and Future Work ............... 187
Bibliography ..................................... 189
I

Thesis
Chapter 1

Introduction

In modern vehicles, more than 80% of innovation comes from the use of special purpose computers [1] which comprise of embedded software executing on embedded processors (embedded systems). Considering the growing complexity of embedded systems, their development cost and lead-time, effective software development methodologies is of paramount importance in the vehicular domain [1] [2]. Researchers and practitioners agreed that abstraction and automation, the founding pillars of Model Driven Engineering (MDE), could be game changers towards the achievement of such a goal [3]. MDE is an engineering paradigm which aims at improving the software development using models and model transformations. Models allow to focus on specific aspects of the software using concepts pertaining to the problem domain rather than constructs pertaining to the underlying technology [3]. Model transformations offer automation in the form of model manipulation (e.g., code-generation) [4]. In the last decade, MDE has been increasingly adopted in the vehicular domain bringing the introduction of several domain-specific modelling languages both from industry and academia. Currently, vehicular embedded software can be described by means of functional models from which execution models\(^1\) are derived. Functional models provide a structured representation of the vehicle’s functions in terms of software functions and interaction among them. Often, they are expressed by means of architectural languages such as the Electronics Architecture and Software Technology Architecture Description Language (EAST-ADL) [5]. Execution models enrich functional models with platform-

\(^1\)In the remainder of this thesis we refer to terms execution models and implementation models as synonyms.
and execution- information such as control flows and worst-case execution times and they are used as the base for verification of non functional properties such as timing. Generally, execution models are derived from functional models and expressed by means of domain-specific modelling languages or component models such as the AUTomotive Open System ARchitecture (AUTOSAR) [6] or the Rubus Component Model (RCM) [7]. However, existing approaches to support models integration in the development of vehicular embedded systems are still immature and the translation between functional and execution models is mainly performed manually. This lack of automation makes the translation of execution models cumbersome thus defers the verification of non functional properties to the last stages of the development process when modification on the software can be 40 times more expensive than the same modifications during the design of the software [8]. In this scenario, providing automation to support model integration would enable an early verification of the non functional properties of the vehicular embedded software. This would allow the engineer to take evidence-based decision during the design of the software when modifications generally require less effort and expense than the same modifications on an almost ready-to-deliver software.

In this doctoral thesis, we define a novel model-driven approach for vehicular embedded systems which supports the development and architectural exploration of system-designs with temporal awareness ensured by means of timing analysis. The proposed approach discloses the opportunity of improving the efficiency of the software development of vehicular embedded systems by replacing the need for late, expensive and time consuming software product modifications with early design changes, which are usually cheaper and faster. Starting from a functional model (expressed by means of EAST-ADL), model transformations generate a set of execution models (expressed by means of RCM). As there might be multiple ways to map elements between models, a source functional model can not be univocally translated into a single correspondent execution model [9]. While most of the current model transformations only consider one particular strategy out of the possible alternatives (of which developers have little or no control) [10], in the proposed approach, model transformations derive all the possible execution models entailing meaningful and unique configurations of modelling elements, from a timing perspective. We draw on existing schedulability analysis for evaluating the appropriateness of the generated execution models with respect to the specified timing requirements. Eventually, model transformations create a reference to the selected execution models along with their analysis results for enabling timing-aware design decisions. In order to ease the visualisation of
the selected execution models, we provide the engineer with a compact and intensional notation able to represent all of them by means of a single model. The proposed approach can be generally applied to non functional properties. However, we centred the approach on timing as it is one of the foremost concerns in the development of real-time systems as vehicular systems (let us think, for instance, to an untimely operation of an airbag or the anti-lock braking system which can cause the loss of lives). Moreover, timing-related issues are a perfect example of those usually discovered at late stages of the development and yet with a great impact on the system design.

1.1 Thesis Contribution

The main scientific contributions of this doctoral thesis are the following:

- A metamodel definition for RCM, called RubusMM, comprising modelling elements for representing software, timing constraints, occurrences and events, execution platform and software to hardware allocation.
- A mechanism for the automatic generation of the execution models, expressed using RubusMM, from a set of starting functional models and requirements, expressed using EAST-ADL.
- A mechanism that performs the analysis, selection and back-propagation of the RCM models which meet the specified set of timing requirements.
- A compact notation for visualising the set of the back-propagated RCM models by means of a single RCM model with uncertainty points.

1.2 Thesis Outline

The remainder of this thesis is organised as follows. Chapter 2 introduces the technical concepts used throughout the thesis. Chapter 3 describes the research goals, challenges and contributions of the thesis. Chapter 4 describes the research methodology and validation. Chapter 5 discusses the literature related to the work and contributions in this thesis. Chapter 6 draws conclusions and future directions. The second part of the thesis consists of Chapter 7 through Chapter 11 and describes the research contributions in terms of the included scientific publications.
Chapter 2

Preliminaries

In this section, we introduce the fundamental technical concepts used throughout this thesis.

2.1 Embedded Systems

An embedded system is a special-purpose computer system which is embedded in the system it controls [11]. Often, it interacts with its environment by means of sensors and actuators. Embedded systems are ubiquitous in electronic items, ranging from microwaves ovens to industrial process controllers. In modern vehicles, embedded systems replace or augment most of the vehicle’s mechanical and hydraulic parts and implement many safety features, e.g., anti-lock braking system. Often, embedded systems have to meet real-time requirements as in the case of the collision avoidance systems. In this case, an embedded system is defined as real-time embedded system and it is expected to interact with its environment in a timely manner [12]. That is, its output is only acceptable when it is functionally correct and is delivered within the specified time.

2.2 Schedulability Analysis

Real-time embedded systems require evidences that their output will be delivered at the time that is suitable for the environment they interact with. Schedulability analysis is *a priori* timing analysis technique which provides evidence
on whether each function in the system is going to meet its timing requirements [13]. In this thesis, we leverage a mature schedulability analysis technique called end-to-end response-time and delay analysis [14]. The analysis calculates upper bounds on the end-to-end response times and delays of chains of tasks and messages in the system.

2.3 Model Driven Engineering

Model Driven Engineering is a software engineering paradigm which aims at raising the abstraction of the software development by shifting the focus from code to models [3]. To this end, MDE promotes models and model manipulations as first-class citizens in the development process. Models represent an abstraction of the system and help an expert to focus on system characteristics of interest, while hiding the others [3]. An example could be modelling functional behaviours, while hiding hardware-specific details. Valid models can be specified in accordance to the set of rules and constraints described by so-called metamodels [3]; valid models are said to conform to their respective metamodel. Within MDE, a software system can be developed by means of model manipulations. That is, abstract models are refined into more detailed models, until code is generated. Model manipulations are performed by means of model transformations. Automated model transformations are programs which automatically translate source models into target models while ensuring their conformance to their respective metamodels [4].

2.4 EAST-ADL

EAST-ADL is an architectural description language which captures the essentials of vehicular systems concerning their documentation, design, analysis and synthesis. EAST-ADL is composed of ten different packages, each of which addresses different aspects of these systems and their development. In this doctoral thesis, we leverage concepts from the structure, requirements and timing packages. The structure package provides for the specification of the software architecture in terms of basic elements and interactions among them. The structure package makes use of four abstraction levels which ensure separation of concerns through the development process. The abstraction levels are: vehicle, analysis, design and implementation. Such a separation is only conceptual and some modelling elements span over several abstraction levels. In this doctoral thesis, we specify the functional models by means of the functional
design architecture, hardware design architecture and allocation concepts from
the EAST-ADL design level. The functional design architecture describes how
the software functions interact. The hardware design architecture describes the
physical architecture of the vehicular embedded system. The allocation de-
scribes the mappings between the elements of the functional design architec-
ture and the hardware design architecture. The timing package provides for the
modelling of the timing constraints stemming from the non functional require-
ments. In this doctoral thesis, we use the elements from the timing package for
the specification of the timing events, occurrences and constraints within the
functional models. The requirement package provides the means for describ-
ing the properties of a vehicular embedded system and their verification. In this
doctoral thesis we make use of elements from the requirements package for the
back propagation of the generated execution models and their schedulability
analysis results to the related EAST-ADL model.

2.5 Rubus Component Model

Rubus Component Model is a modelling language for the predictable develop-
ment of resource-constrained embedded real-time systems developed by Arcti-
cus Systems AB\(^1\) in collaboration with Mälardalen University. Currently, it
is used by several OEM, Tier-1 and Tier-2 companies in the vehicular domain
(e.g., Volvo Construction Equipment, BAE Systems Hagglunds, Hoerbiger and
Knorr Bremse) as the modelling language for representing execution models
and in cooperation with architectural languages such as EAST-ADL. Currently,
RCM supports the modelling of software architecture, execution platform, allo-
cation information and timing properties of vehicular embedded systems [15].

Within RCM, the embedded software architecture is modelled by means
of software circuit (SWC) elements and interactions among them. A SWC
encapsulates a software function. SWCs can be grouped in composite ele-
ments called Assemblies. As the main goal of RCM is to support the pre-
dictable development of vehicular embedded systems, timing properties and
constraints are pivotal in the language and they can be specified at different
hierarchical levels. Within RCM, the execution platform is modelled by means
of node, core and partition elements. Allocation information can be specified
among any two elements of the software architecture and execution platform.
In this doctoral thesis, we provide a canonical metamodel definition for RCM,

\(^1\)https://www.arcticus-systems.com
namely RubusMM, as part of our research contribution. Moreover, we employ RubusMM for the specification of execution models.

2.6 Uncertainty

In software engineering, uncertainty is a meta-property caused by the lack of knowledge or unresolved decisions [16]. In this thesis, we adopt a language-centric approach for managing uncertainty, i.e., multiple models, which is able to generate at once and represent the entire solution space of the generated models in the intensive form of a model with uncertainty [9].
Chapter 3

Research Goal, Challenges and Contributions

This chapter discusses research goal, research challenges and research contributions.

3.1 Research Goal

Timing verification is essential and unavoidable for the development of real-time embedded systems such as vehicular embedded systems. However, researches show that timing verification is more efficient when it is performed earlier during the development process as modifications during the last stages of the development can be 40 times more expensive than the same modifications during the design of the software [8]. To this end, we believe that enabling timing verification at design level, by means of integration through model transformations, can improve the efficiency of the software development of vehicular embedded systems. In fact, timing verification results could be used for driving the design process and replacing the need for late, expensive and time-consuming software modifications with earlier design modifications, which are usually cheaper and faster. The overall goal of this research work is to improve the efficiency of the software development of vehicular embedded systems by supporting the development and architectural exploration of system-designs with temporal awareness. More precisely, we aim at providing automation for the generation of execution models, expressed by means
of RubusMM, starting from functional models, expressed by means of EAST-ADL. In addition, we aim at providing an automatic support for the selection of the generated RCM models that meet the specified timing requirements as well as for their back-propagation and visualisation at design level.

### 3.2 Research Challenges

Starting from the research goal, we derived the following research challenges (RCs) and used them as main drivers for the research work presented in this doctoral thesis.

**RC 1. Definition of a metamodel for RCM.** Currently, vehicular embedded software can be described through various modelling languages such as EAST-ADL and RCM. Consequently, MDE seems a natural choice for enabling the automatic integration among the languages. Metamodels and model transformations are the founding pillars of MDE and they serve for regulating the specification of models and for automating their manipulations, respectively. Therefore, in order to enable a full-fledged MDE approach, it is paramount to provide a metamodel definition to all the languages involved in the software development of vehicular embedded systems. Before this research effort, RCM did not have a canonical metamodel specification, but it rather relied on a textual language specification.

The challenge is the definition of a metamodel for RCM comprising modelling elements for representing software and the execution platform architectures, the timing constraints, occurrences and events of the vehicular embedded system and the software to hardware allocation. In particular, the metamodel should be defined bearing in mind backward compatibility with legacy RCM artefacts and should not entail any modification to the Rubus run-time layer.

**RC 2. Definition of a mapping between EAST-ADL and RCM metamodels.** Timing verification is crucial task in the development of vehicular embedded systems. However, it gives meaningful results only when applied on execution models as functional models do not entail detailed, e.g., timing, control and allocation information. One way to leverage timing verification results at design level, is the definition of an automatic and transparent process for the generation of RCM models from EAST-ADL models. However, due to the different levels of abstraction between EAST-ADL and RCM, there might be multiple ways to generate RCM from EAST-ADL models.
3.3 Research Contributions

In this context, the challenge is two-fold. On the one hand, we need to define an automatic process able to generate the RCM models containing all the needed software architecture, timing, control and allocation information. On the other hand, this process should be able to generate all the possible RCM models entailing meaningful and unique timing as well as allocation configurations as opposed to considering only one particular generation strategy [9] [10].

**RC 3. Definition of a mechanism for unveiling the feasible RCM models at design level.** Once the RCM models have been generated and the schedulability analysis performed, the RCM models satisfying the specified timing requirements must be unveiled at design level for enabling timing-aware design decision. Here the challenge is two-fold. On the one hand, the generated RCM models must be compared against the specified timing requirements and back-propagated at design level. On the other hand, it is crucial that all the RCM models satisfying the specified timing requirements are back-propagated at design level and represented in a convenient notation, which highlights the models’ commonalities and differences for aiding possible manual investigations. In fact, at this point, the selection can not be automated and it can only be made by manually investigating the set of selected RCM models considering perhaps additional non functional properties.

3.3 Research Contributions

Early verification of non functional requirements can positively affect the efficiency of the development of vehicular real-time embedded systems. Currently, early verification of non functional requirements is hard to achieve due to the lack of automation supporting models integration and analysis. For instance, let us consider a typical development process as described by the flowchart in Figure 3.1. In this setting, as meaningful non functional analysis (such as timing) must be run on execution models, the engineer is required to create one manually. The non functional analysis of interest is run on the manually created model and the result is verified against the given set of requirements. If the specified non functional requirements are not met, the engineer is has to iterate the process, modify or create a new execution model until a compliant one is found. Since the process of creating and verifying execution models is expensive, it is not leveraged early in the development process for having quick and early feedback on the design level models. To boost early verification, in this thesis we propose a novel model-driven approach for the devel-
development of vehicular real-time embedded systems supporting early verification of non functional properties. Let us consider a development process equipped with the proposed approach as described by the flowchart in Figure 3.2. In this setting, all meaningful execution models are automatically generated from the design model and analysed by means of model transformations, at once. Considering a set of non functional requirements, model transformations are responsible for the selection and back propagation of the best execution model (or set of models), too. Besides relieving the engineer from the manual definition of execution models, the proposed approach enables early verification at design level. In addition, while in the manual process several iterations may be needed to reach a compliant execution model, the proposed approach generates all meaningful execution models and identifies the best one(s) automatically in one single iteration.

As timing requirements are crucial for our domain of interest and timing-related issues are typical problems arising very late in the development process, in this thesis we center the proposed approach on timing. In particular, the
3.3 Research Contributions

Figure 3.2: Development process equipped with the proposed approach

The main contribution of this doctoral thesis is a model-driven approach supporting the development and architectural exploration of system-designs with temporal awareness ensured by means of schedulability analysis. Figure 3.3 provides a graphical representation of the proposed approach.

The proposed approach leverages the interplay of EAST-ADL and RCM as the modelling languages for expressing functional and execution models, respectively, and a suite of 6 model transformations. The first step of the proposed approach is the automatic generation of RCM models representing the software architecture and its timing properties and constraints from an EAST-ADL functional design architecture equipped with EAST-ADL timing constraint modelling elements. Such a generation is entrusted to the FDA2RCM model transformation. As there could be multiple ways of generating RCM models from an EAST-ADL functional design architecture, the FDA2RCM model transformation generates, in a single execution, all the RCM models...
entailing unique timing and control flow information. The second step of the proposed approach is the automatic generation of an RCM model representing the execution platform from an EAST-ADL hardware design architecture and it is performed by the HDA2RCM model transformation. At this point, as RCM describes the execution platform at a different level of abstraction compared to EAST-ADL, manual refinements of the generated RCM model representing the execution platform may be needed in order to, e.g., specify cores and partitions in the case of vehicular embedded systems for multi-core. This is a necessary step as detailed execution platform models are pivotal for the specification of the software allocation information which, in turn, affects schedulability analysis. The next step of the proposed approach merges the generated RCM software and execution platform models to obtain a set of complete RCM models where the software allocation information can be specified. This step is performed by the MERGE model transformation. The specification of the allocation information on the merged RCM models is entrusted to two model transformations, namely A2RCM and ALLOCATION. The former is responsible for translating the allocation information from the EAST-ADL allocation model. The latter is responsible for generating RCM models entailing those allocation configurations that can not be directly derived from the EAST-ADL allocation model as in the case of, e.g., allocation of software to core and partition elements. As there could be multiple unique allocation configurations, the ALLOCATION model transformation generates, in a single execution, all the RCM models entailing unique allocation information. At this point, schedulability analysis is run on the generated RCM models. If none satisfies the set of specified timing requirements, the engineer is notified about the inability of the initial EAST-ADL model to satisfy its timing requirements. Otherwise, the RCM models satisfying the specified timing requirements are propagated back, together with their analysis results, at the design level by the BP model transformation and visualised as a single RCM model with uncertainty. Figure 3.3 provides a breakdown of the main contribution in specific research contributions (RCOs) while Table 3.1 shows the relation between them and the RCs.

**RCO 1 - RubusMM.** This contribution, marked as 1 in Figure 3.3, provides a metamodel definition for RCM as a needed step for enabling integration through model transformations. In fact, RCM was originally thought for providing modelling purposes, but it did not feature any model driven mechanism, i.e., automation in terms of model transformation. RubusMM has been defined through a two-step process. In the first step, we reverse-engineered the RCM specification with the aim of restoring the separation of concerns lost
3.3 Research Contributions

Figure 3.3: Research Contributions
during the evolution of the component model. As a side effect, this allowed us to maximise backward compatibility with legacy RCM artefacts. This activity resulted in the addition of modelling elements such as connectors and threads as well as in the refinement of hierarchical structures. In the second step, we extended RubusMM for the modelling of vehicular embedded systems on single- and multi-core platforms. This extension includes modelling elements for representing the execution platform and the software to hardware allocation information. It is important to note that the extension does not affect backward-compatibility as it does not modify any hierarchical structure. Currently, RubusMM is defined as an Ecore model, within the Eclipse Modeling Framework\(^1\) (EMF), and comprises modelling elements for representing i) the software architecture and timing constraints, occurrences and events, ii) the execution platform and iii) the software to hardware allocation information.

This contribution provides a solution to RC 1. Paper A presents the reverse-engineered version of RubusMM while Paper D presents the extended RubusMM for single- and multi-core.

**RCO 2 - Mechanism for the automatic generation of execution models.** This contribution, marked with 2 in Figure 3.3, provides an automatic mechanism for the generation of RCM models from EAST-ADL models. This is fundamental for enabling timing-aware design decision by integration through model transformation.

This contribution comprises a set of five model transformations namely FDA2RCM, HDA2RCM, MERGE, A2RCM and ALLOCATION. The contribution brought by them is two-fold. On the one hand, they provide automatic generation of RCM models, which are the input for schedulability analysis. On the other hand, they provide generation of all the meaningful RCM models from an initial EAST-ADL model.

The FDA2RCM transformation provides for the generation of RCM models representing the software architecture and its timing constraints from an EAST-ADL functional design architecture equipped with EAST-ADL timing constraints. In a nutshell, it translates the elements of the EAST-ADL functional design architecture to RCM software elements. Additionally, it provides automatic generation of all control flow and timing elements in the RCM models. Since such a translation can produce multiple RCM models entailing unique configurations of control flow and timing elements, FDA2RCM generates, in a single execution, all of them. This is possible thanks to a bidirectional

\(^1\)http://www.eclipse.org
model transformation language, namely the Janus Transformation Language (JTL) [17]. JTL is a constraint-based bidirectional model transformation language specifically tailored to support one-to-many model transformations by generating all the possible models, at once. JTL adopts a Query/View/Transformation (QVT) relation-like syntax [18] and relies on the Answer Set Programming (ASP) [19], which is a declarative programming language based on the answer set (model) semantics of logic programming. The ASP solver, by means of a deductive process, finds and generates in a single execution all the models that are consistent with the transformation rules. For instance, the application of the FDA2RCM transformation to the simplified EAST-ADL functional design architecture depicted in Figure 3.4a produces the four simplified RCM models depicted in Figure 3.4a.

It is worth to mention that JTL supports the specification of logic constraints, which can be used for narrowing the number of generated models and tailoring their generation for specific purposes. In the specific case of the FDA2RCM model transformation, we employed logic constraints for generating RCM models entailing valid configurations of control flow and timing elements, only. For instance, in the case of the simplified EAST-ADL functional design architecture in Figure 3.4a, the logic constraints prevent the generation of the four RCM models where software function Function 1 was not triggered by an independent clock.

The HDA2RCM transformation provides generation of the RCM model representing the execution platform from an EAST-ADL hardware design architecture. It is implemented by means of JTL and translates the EAST-ADL node, connectors and port elements into corresponding elements in RCM. Moreover, in order to conform to the RCM hierarchy of execution platform elements, for each generated RCM Node element, a Core and a Partition element are created, too. In fact, compared to EAST-ADL, RCM describes the execution platform at a different level of abstraction by using core and partition concepts. Please note that, the engineer can still manually refine the generated RCM execution platform model by using the RCM Core and Partition elements.

The MERGE transformation merges the generated RCM models representing software architecture and execution platform for allowing the translation of the allocation information from EAST-ADL. MERGE is implemented as a QVT Operational (QVT-O) transformation which performs a weaving of the RCM models, where the modelling elements of the RCM execution platform model are linked to the System element of the RCM software model through its Node reference. The translation of the allocation information is entrusted to the A2RCM transformation. A2RCM is an in-place transformation writ-
ten in QVT-O and that sets the reference isAllocated of the RCM Allocatable elements starting from the allocation information expressed by means of the EAST-ADL Function Allocation elements.

Due to the different level of abstraction between RCM and EAST-ADL, complete allocation information for the RCM models cannot be directly derived from an EAST-ADL Allocation. In this context, the ALLOCATION transformation provides automation means for the generation of the allocation information in the RCM models when a direct translation from EAST-ADL is
3.3 Research Contributions

not possible. The engineer is required to set which software elements must be allocated to which execution platform elements. (e.g., Assembly to Core, Assembly to Partition, SWC to Core). Based on the engineer choice, the ALLOCATION transformation automatically generates, in a single execution, all the RCM models which entail unique allocation configurations of RCM Allocatable to Allocator elements. Similar to FDA2RCM, this is implemented as a JTL model transformation. It is worth to note that logic constraints can be applied for reducing the number of generated RCM models when, e.g., allocation information is already available.

This contribution provides a solution to RC 2. Paper B provides an initial version of this automation mechanism consisting of the FDA2RCM transformation only. Paper E discusses this contribution in its complete version.

**RCO 3 - Back-propagation of analysis results to design level.** This contribution, marked as 3 in Figure 3.3, enables the selection of the generated RCM models satisfying the specified timing requirements and their back-propagation to design level. This represents the last step in the process of enabling timing-aware design decisions.

The contribution is embodied by BP, an in-place model transformation, which takes as input the generated RCM models, their schedulability analysis results and the set of specified timing requirements. First, BP compares the analysis results with the specified timing requirements and discards those not fulfilling the requirements. Afterwards, it adds to the initial EAST-ADL model the elements from the requirements package for the validation of the software. Finally, it enriches the added elements with the references to the folders containing the selected RCM models and their analysis results. Currently, BP is defined as a QVT-O transformation.

This contribution provides a solution to RC 3. Paper B and Paper E discuss the initial and enhanced version of this contribution, respectively.

**RCO 4 - Compact visualisation of multiple Rubus models.** Multiple RCM models can be selected and back propagated to design level and no further selection can be automated as all selected RCM models have equally good schedulability analysis results. This contribution, marked as 4 in Figure 3.3, provides a mechanism for the compact representation of all these equally good RCM models in terms of their commonalities and distinctions by means of a single RCM model with uncertainty points. The intent is to allow the engineer to deal with the set of selected RCM models as if they were a single model and enable further selection based on, e.g., architectural choices or other relevant...
non functional properties. Such a representation is achieved by employing u-RubusMM, which is a revised version of RubusMM endowed with uncertainty elements. This is done by employing the metamodel-independent technique presented in [9]. More precisely, an automated model transformation defined in u-JTL [9] is responsible for the generation of u-RubusMM starting from RubusMM.

This contribution provides a solution to RC 3. Paper C provides further details about u-RubusMM.

Table 3.1 shows the relation between RCOs and RCs.

<table>
<thead>
<tr>
<th>Research Contributions</th>
<th>RCO 1</th>
<th>RCO 2</th>
<th>RCO 3</th>
<th>RCO 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCO 1</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RCO 2</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RCO 3</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RCO 4</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Table 3.1: Research Contributions in Relation to the Research Challenges

### 3.4 Papers Contribution

This section lists the papers included in the thesis and shows the relations between them and the RCOs, as discussed in Section 3.3, in Table 3.2.

<table>
<thead>
<tr>
<th>Papers</th>
<th>RCO 1</th>
<th>RCO 2</th>
<th>RCO 3</th>
<th>RCO 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: Included papers in Relation to the Research Contributions
3.4 Papers Contribution

3.4.1 Paper A


Abstract – According to the Model-Driven Engineering paradigm, one of the entry requirements when realising a seamless tool chain for the development of software is the definition of metamodels, to regulate the specification of models, and model transformations, for automating manipulations of models. In this context, we present a metamodel definition for the Rubus Component Model, an industrial solution used for the development of vehicular embedded systems. The metamodel includes the definition of structural elements as well as elements for describing timing information. In order to show how, using Model-Driven Engineering, the integration between different modelling levels can be automated, we present a model-to-model transformation between models conforming to EAST-ADL and models described by means of the Rubus Component Model. To validate our solution, we exploit a set of industrial automotive applications to show the applicability of both the Rubus Component Model metamodel and the model transformation.

Status. Published.

Personal Contribution. The research work presented in this paper was done in collaboration with all the authors. However, I was the main contributor and driver. More specifically, I i) reverse-engineered the RCM language, ii) provided a canonical metamodel definition to RCM, called RubusMM and iii) extended RubusMM with the modelling elements for the integration with EAST-ADL.

3.4.2 Paper B

Anticipating Implementation-Level Timing Analysis for Driving Design-Level Decisions in EAST-ADL. Alessio Bucaioni, Antonio Cicchetti, Federico Ciccozzi, Romina Eramo, Saad Mubeen, Mikael Sjödin. 1st International Workshop on Modelling in Automotive Software Engineering (MASE) (acceptance rate: 41%) co-located with the ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS). Ottawa,
Chapter 3. Research Goal, Challenges and Contributions

Canada. September, 2015.

Abstract – The adoption of model-driven engineering in the automotive domain resulted in the standardization of a layered architectural description language, namely EAST-ADL, which provides means for enforcing abstraction and separation of concerns, but no support for automation among its abstraction levels. This support is particularly helpful when manual transitions among levels are tedious and error-prone. This is the case of design and implementation levels. Certain fundamental analyses (e.g., timing), which have a significant impact on design decisions, give precise results only if performed on implementation-level models, which are currently created manually by the developer. Dealing with complex systems, this task becomes soon overwhelming leading to the creation of a subset of models based on the developers experience; relevant implementation-level models may therefore be missed. In this work, we describe means for automation between EAST-ADL design and implementation levels to anticipate end-to-end delay analysis at design level for driving design decisions.

Status. Published.

Personal Contribution. The research work presented in this paper was done in collaboration with all the authors. However, I was the main contributor and driver. More specifically, i) defined the methodology, ii) implemented its composing tasks and iii) applied the solution to the running example.

3.4.3 Paper C


Abstract – Models and model transformations, the two core constituents of Model-Driven Engineering, aid in software development by automating, thus taming, error-proneness of tedious engineering activities. In many cases, the result of these automated activities is an overwhelming amount of information. This is the case of one-to-many model transformations that, e.g. in model-based design-space exploration, can potentially generate a massive amount of
candidate models (i.e., solution space) from one single source model. In our scenario, from one design model we generate a set of possible implementation models on which timing analysis is run. The aim is to find the best model from a timing perspective. However, multiple implementation models can have equally good analysis results. Therefore, the engineer is expected to investigate the solution space for making a final decision, using criteria which fall outside the analysis’ criteria themselves. Since candidate models can be many and very similar to each other, manually finding differences and commonalities is an impractical and error-prone task. In order to provide the engineer with an expressive representation of models’ commonalities and differences, we propose the use of modelling with uncertainty. We achieve this by elevating the solution space to a first-class status, adopting a compact notation capable of representing the solution space by means of a single model with uncertainty. Commonalities and differences are thus represented by means of uncertainty points for the engineer to easily grasp them and consistently make her decision without manually inspecting each model individually.

**Status.** Published.

**Personal Contribution.** The research work presented in this paper was done in collaboration with all the authors. However, I was the main contributor and driver. More specifically, i) provided RubusMM with the uncertainty notation and ii) applied the solution to the running example.

### 3.4.4 Paper D


**Abstract** – The vehicular industry has exploited model-based engineering for design, analysis and develop of single-core vehicular systems. Next generation of autonomous vehicles will require higher computational power, which can only be provided by multi-core platforms. Current model-based solutions and related modelling languages, originally conceived for single-core, can not effectively deal with multi-core specific challenges, such as core-interdependency and allocation of software to hardware. In this paper, we propose an extension
to the Rubus Component Model, core of the Rubus model-based approach, for
the modelling, analysis and development of vehicular systems on multi-core.
Our goal is to provide a lightweight transition of a model-based approach from
single-core to multi-core, without disrupting the current technological assets in
the vehicular domain.

Status. Published.

Personal Contribution. The research work presented in this paper was done
in collaboration with all the authors. However, I was the main contributor and
driver. More specifically, I i) extended RubusMM with the modelling elements
for representing the execution platform and the software to hardware allocation
information and ii) conducted the running example.

3.4.5 Paper E

A Model-based Approach for Vehicular Systems. Alessio Bucaioni, Lorenzo
Addazi, Antonio Cicchetti, Federico Ciccozzi, Romina Eramo, Saad Mubeen,
Mikael Sjödin. MRTC Report MDH-MRTC-321/2017-1-SE. Västerås, Swe-

Abstract – This paper introduces a novel model-based approach for the soft-
ware development of vehicular embedded systems. The proposed approach
discloses the opportunity of improving efficiency of the development process
by providing support to identify viable design solutions with respect to selected
non functional requirements. To this end, it leverages the interplay of two mod-
ellng languages for the vehicular domain whose integration is achieved by a
suite of model transformations. An instantiation of the methodology is dis-
cussed for timing requirements, which are among the most critical ones for
the development of vehicular systems. The applicability of the methodology is
demonstrated as proof of concepts on industrial use cases performed in coop-
eration with our industrial partners.

Status. Under review.

Personal Contribution. The research work presented in this paper was done
in collaboration with all the authors. However, I was the main contributor and
driver. More specifically, I i) defined the methodology and iii) applied the so-
lution to the running example.
Chapter 4

Research Methodology and Validation

This chapter discusses research methodology and validation.

4.1 Research Methodology

Collaborative research between industry and academia is a great example of how research in software engineering is often stimulated by problems arising from the use of software in the real life [20]. In this respect, the research presented in this thesis was conducted in a partnership between Mälardalen University and Arcticus Systems with the collaboration of Volvo Construction Equipment, Saab Avionics Systems and BAE Systems. For this research, we adopted a methodology being an adaptation of the model for technology transfer described in [21]. Figure 4.1 gives a graphical representation of the adopted research methodology. We began by assessing the state-of-the-art, the state-of-the-practice and the industrial demands with the aim of defining a research goal. During these stage, we identified several research challenges connected to the main research goal. For each elicited challenge, we investigated the state-of-the-art and practice with the aim of identifying a possible solution, if none existed. After performing the investigation, we defined a candidate solution. In this stage, the industrial partners played a crucial role as they provided early and quick feedbacks ensuring that the candidate solution was realistic and could fit current practices and industrial needs. The validation of each
candidate solution required three steps. During the academic validation, each solution was evaluated in the university by means of case study performed by researchers. Eventually, we used the finding acquired during the academic validation for refining the existing solutions or defining new research challenges. For instance, this was the case of RubusMM, whose definition described in Paper A was refined in the definition given in Paper D. During the static validation, we presented the candidate solutions to the industrial partners in a series of dedicated meetings and workshops. The aim of this step was to collect feedbacks regarding the usability and scalability of each solution. The feedbacks acquired during the static validation were sued for for refining the existing solutions or defining new research challenges as in the case of the visualisation mechanism described in RCO 4. In fact, the challenge of having an intensional and convenient notation for representing a multitude of models as one model with uncertainty arose only when the generation mechanism described in RCO 2 was able to generate a set of RCM models. Eventually, we performed the dynamic validation by means of industrial projects and experiments.

4.2 Validation

The work presented in this doctoral thesis and its contributions have been evaluated progressively as prescribed by the research methodology in Section 4.1. With respect to RubusMM, we verified its consistency, expressiveness and applicability against several industrial system designs such the Brake-By-Wire
(BBW) [15], Steer-By-Wire (SBW) [22], Intelligent Parking Assist (IPA) [23], etc. Moreover, the industrial partners played a key role in providing feedbacks regarding the its industrial relevance.

Apart from RubusMM, the remaining contributions are implemented by means of model transformations. In this respect, the three validation steps described in the research methodology helped us in discussing some interesting properties of the model transformations, such as syntactic and semantic correctness, complexity, termination and performance [24]. With syntactic correctness, we refer to the ability of a transformation to produce valid target models when executed on valid source models [24]. Such a property holds for the transformations presented in this thesis and we demonstrated it by means of the case studies done during the academic and dynamic validation. With the term semantic correctness, we refer to the ability of a transformation to produce semantically valid target models [24]. Such a property holds for the transformations presented in this thesis and one way we entrusted it was to define the transformations by means of a precise and finite set of rules mapping EAST-ADL to RCM elements without altering, violating or colliding the structural hierarchies of the languages. Moreover, the semantic of the generated RCM models was validated by the practitioners during the static validation and by the leveraged schedulability analysis. We considered two dimensions for the transformations complexity which are the intricacy and the number of the generated RCM models. During the static validation, we conveyed that the generated RCM models have equal complexity of manually defined ones. Although some of the model transformations can theoretically produce multiple RCM models, during the academic and dynamic validation we were able to demonstrate that the transformations always terminate in few seconds and produce only a limited number of RCM models. Moreover, the transformations could be refined by the engineer on the basis of the specific system and the solution space can be reduced by adding constraints that operate on the possible mapping policies.

We believe that the automation introduced by the proposed approach discloses the opportunity to improve the efficiency of the software development process by means reduced need for late modifications on the software. In particular, model transformations allow to cut the development time while ensuring the compliance with the non functional requirements of the vehicular embedded software. Without the proposed approach, in fact, the development would progress incrementally with team of engineers manually defining exe-

---

1Please note that, providing a formal proof on the transformations’ termination is outside the scope of this thesis.
unction models until a suitable one, from a non functional perspective, is found. On the contrary, with the proposed approach, the execution models are automatically generated and non functional requirements verified at once allowing the engineers to focus and reason only on the compliant models. By enabling early verification, the proposed approach discloses the opportunity of reducing late modifications on the vehicular software, which empirical studies showed to be generally more expensive than modifications during the design of the software [8]. In fact, by adopting the proposed approach, the engineer is either notified on the non compliance of the starting EAST-ADL model to the set of the considered non functional requirements or notified with the set of the compliant RCM models with which proceed for the development. In the former scenario, late modifications are prevented while in the latter they are not needed.

In this thesis, given the importance of timing properties during the design and development of vehicular real-time embedded systems, we centred the proposed approach on timing. However, we recognise that further non functional properties such as memory usage, energy efficiency, and so forth, play an important role during the development of these systems. In this respect, it is worth to note that the proposed approach proposed can be instantiated to consider further properties, as long as they are measurable and comparable at the EAST-ADL and RCM levels of detail. Additionally, other properties can be exploited for selecting multiple RCM models having equally good timing performance or can be considered from the initial stages of the generation process of the possible solutions. In both cases, the proposed approach would need to be extended only in terms of specific model transformations for the generation of the related non functional properties of interest.
Chapter 5

Related Works

This doctoral thesis deals with the research problem of supporting a timing-aware model driven development of distributed vehicular embedded systems on single- or multi-core platforms. Hereafter, a number of relevant similar approaches are discussed.

AUTOSAR [6] is an industrial initiative to provide a standardised software architecture for the development of vehicular embedded systems. In the TIMMO and TIMMOUSE projects, AUTOSAR was provided with a timing model [25]. However, the AUTOSAR timing model does not distinguish between the control and the data flows at the application software level, a distinction that is fundamental for providing early timing verification [26]. A framework to specify the end-to-end timing constraints and analyse the corresponding end-to-end delays [27, 28, 29] was proposed in the aforementioned projects too.

There are some works [30] [31] that aim at supporting end-to-end timing analysis of the higher-level abstraction models. These works heavily rely on the availability of legacy models and favour the bottom-up development approach when the existing systems are extended. Unlike our methodology, these works are not applicable to the top-down development approach. Another work [26] refines timing requirements between the design and implementation levels to support the timing analysis at the implementation level. Unlike our methodology, this work performs manual translation of the software architecture from the design level to the implementation level. However, this work can be used complementary to our work by running the timing analysis of the implementation level models that are automatically generated by our methodology.
Often, AUTOSAR is used in cooperation with EAST-ADL as a modelling language for the implementation level. Even if EAST-ADL entails abstraction and separation-of-concerns, there is no specific automation support for interconnecting the abstraction levels in the structure package, e.g., between design and implementation levels. As a consequence, schedulability analysis and their results have to be tackled and tracked back manually by the engineer due to the abstraction gaps between the different levels. These tasks can be time-consuming and error-prone, especially when considering the complexity of modern vehicular systems [32]. On the contrary, in this doctoral thesis we propose to leverage automation through model transformations to keep the consistency between the different abstraction levels. The abstraction gaps naturally introduce multiple choices, which are managed by an appropriate transformation language (JTL). Moreover, AUTOSAR does not provide means for modelling the execution platform [33].

CHESS is a cross-domain framework for the design of component-based embedded systems, including vehicular systems [34]. CHESS features a specific UML profile obtained as a combination of different languages, e.g., MARTE and SysML. The framework provides modelling of embedded software for early analysis, such as dependability and schedulability, as well as for code generation, monitoring and back-propagation. Currently, CHESS does not support the design space generation and does not provide means for representing uncertainty in the development process.

Vehicular embedded systems, especially when considering autonomous driving and networks of vehicles, are often referred to as cyber-physical systems (CPS) [35]. In the recent years, several approaches dealing with CPS development by adopting multi-paradigm modelling techniques and leveraging simulation mechanisms to perform early analysis of systems have been proposed [36, 37]. Although the work presented in this doctoral thesis does not exploit simulation techniques, it does not prevent the use of simulation mechanisms to analyse and select the generated design alternatives with respect to non functional properties of interest.

Given the ubiquity of software, there exists a corpus of literature devoted to the design of embedded systems and posing a special focus to non functional requirements. In this respect, several works are based on the use of general-purpose languages such as UML and the UML profile for MARTE [38] as alternatives to domain-specific languages like, e.g., AUTOSAR and RCM. GASPARD is a MARTE-based framework for the design of parallel embedded systems [39]. It prescribes a workflow made-up of subsequent analyses and refinement steps, from higher to lower abstraction levels. Similar to the pro-
posed approach, the analysis at lower levels are meant to produce feedback for the design activities and the process is automatised by means of model transformations. Compared to our approach, GASPARD seems to tackle different non-functional aspects which can be complementary to timing. Moreover, GASPARD does not provide for the compact representation of the design space.

MARTE is adopted also in [40] to design the high-level architecture of the software system and for the code generation. It uses UML for modelling the software components and MARTE for modelling the hardware and the software to hardware allocations. One major difference with respect the proposed approach is that timing requirements are verified by means of simulations on the generated code and not starting from functional models. In [41], the authors propose a technique to specify tasks and their allocation to cores. The technique is based on MARTE and allows to perform simulation and task allocation optimisation based on the execution whereas the approach proposed in this doctoral thesis statically generates all the possible allocation configurations.
Chapter 6

Conclusions and Future Works

The work presented in this thesis describes a timing-aware model-driven approach for the software development of vehicular embedded systems. In particular, it tackles the problem of guiding the engineer in taking timing-aware decisions at design level when modifications are generally less expensive than modifications at later stages of the development. The overall contribution of this thesis can be broken down into four main research contributions, which are: i) a metamodel definition for RCM, ii) a set of model transformations for the generation of RCM from EAST-ADL models, iii) a mechanism for the automatic selection and back-propagation of the generated RCM models to the design level and iv) a visualisation mechanism for representing the set of the generated RCM models as on RCM models with uncertainty, only.

The formalisation of the RCM metamodel is pivotal for leveraging the proposed approach. The set of model transformations provide automation means for integrating EAST-ADL and RCM, avoiding manual, time-consuming and error-prone activities. In general, the integration between EAST-ADL and RCM requires the generation of a number of RCM models which can rapidly grow exponentially with respect to the number of software components in the software architecture and their allocation. We proposed to solve this by leveraging the properties of a constraint-based transformation language, JTL, to automatically derive all the possible RCM models entailing meaningful and unique timing and allocation configurations. By doing so we could leverage schedulability analysis at design level avoiding the problem of manually iden-
tifying a suitable RCM model, in terms of timing characteristics. In addition to replace the need for late and expensive modifications, the proposed approach also discloses the opportunity of employing expensive resources, such as engineers, more efficiently allowing them to focus only on the design activities exploiting schedulability analysis results without having to investigate nor manually edit execution models. The selection and back-propagation mechanism together with the visualisation mechanism provide a powerful tool for the identification of the best RCM models in terms of timing characteristics thus for taking timing-aware design decisions.

Despite the generation of the RCM models is transparent to the engineer and it can be guided through logic constraints, issues about scalability and performance may remain open when dealing with complex functional models. In this respect, one of the main future investigation direction encompasses the study of a smarter generation process which could reduce the number of the generated RCM models. To this end, one possible solution could be the use of more (domain-)specific logic constraints. Another possible solution to a smarter generation could be to extend the proposed methodology for the optimisation of further system properties, e.g., memory demands. In fact, our experience showes that, besides timing, other relevant system properties need to be dealt with at design level and can be used for pruning the space of the generated RCM models. To summarise, schedulability analysis can represent one step in an exploration chain, where the solution spaces are sequentially investigated based on different system properties [42].
Bibliography


Bibliography


A Model-driven Development Approach with Temporal Awareness for Vehicular Embedded Systems

Alessio Bucaioni

Alessio Bucaioni is an industrial Ph.D. student at the School of Innovation, Design and Engineering at Mälardalen University, Sweden and Arcticus Systems AB, Sweden. He holds a Licentiate Degree in Computer Science and Engineering from Mälardalen University.

Alessio’s research interests include, but are not limited to, model driven engineering and software engineering. Currently, Alessio is focusing on model driven development of vehicular embedded systems.