A Resource-Aware Component Model for Embedded Systems

by Aneta Vulgarakis
Mälardalen University
aneta.vulgarakis@mdh.se
December 15, 2008

Licentiate Proposal

Main Supervisor: *Ivica Crnković*
Assistant Supervisors: *Paul Pettersson*
Cristina Seceleanu

Mälardalen University

conducted at:
Mälardalen University
School of Innovation, Design and Technique
The PROGRESS Centre for Predictable Embedded Software Systems
Västerås, Sweden

Abstract

Component-based development has proved to be a promising approach for reusability and managing complexity, but to date has not been extensively used for handling the requirements from embedded systems domain (real-time constraints, resource limitation etc). It is based on component models i.e. specification of components and their interaction. Most of the existing component models tailored for embedded systems only cover a small set of the embedded systems' requirements. This paper is a licentiate proposal that glances through the work that has been done on building a resource-aware component model for embedded systems. In this work we have proposed a two-layered component model - ProCom for design and development of embedded systems and a modeling framework - REMES for modeling and reasoning of components' and systems' behavior that includes relevant resource types for embedded systems.

Contents

1 Background and Motivation

Embedded systems are present everywhere around us, in our MP3 players, digital watches, mobile phones, cars, nuclear power plants, satellites and so on. An embedded system is a special-purpose computer system that forms part of a larger system dedicated to perform a specific task often with real-time computing constraints. The software of such systems is growing and becoming more complex, thus their development requires new methods to cope with these needs. A promising approach to handle the complexity, introduce structure and abstractions, lies in the adaptation of a Component-Based Development (CBD) approach. The basic rationale for the field of CBD [8, 22] is the idea of constructing systems by reusing existing components, in much the same way as standard components are used in electronics or mechanics: integrated circuits, switches, etc.

In contrast to the unsteady nature of software, the available resources that embedded systems use (computational power, energy, memory, and hardware components such as buses, input/output ports, etc.) are limited in capacity, expensive and (usually) not extensible during the system's lifetime [14, 15, 16, 11, 9, 12, 3, 20]. Since resource efficiency is of a high importance in embedded systems there is a need for resource consumption modeling and analysis during the whole system lifecyle. This should include both early stage resource-usage analysis based on abstract system models and platform abstractions and later stage analysis when components are grouped into tasks and mapped onto hardware units.

There is a broad range of potential formalizations of component models, but most designers describe the function, and very few describe the behavior of their components as a relation between a set of inputs and a set of outputs. This relation is often informally defined, not logically connected by a set of precise relationships and even expressed in a natural language [10]. To enable formal analysis a component model needs to be formally defined. In order to reason about the resource consumption, which is important for embedded systems, once the component model is defined, it should be enriched with behavioral and resource usage information such as resource types and constraints.

One potential formal verification technique is model checking. Model-based verification techniques are based on models describing the possible system behavior in a mathematically precise and unambiguous manner. They allow for desired behavioral properties of a given system to be verified on the basis of a suitable model of the system through brute-force inspection of all states of the model [5]. Main advantage of the model checking approach comes from the fact that it is completely automatic and that it provides counterexamples if a model fails to satisfy a property. These counterexamples can later help in the debugging process of the model.

This paper is a licentiate proposal that outlines the research that has been conducted on developing a resource-aware component model for embedded systems within the frames of the PROGRESS project [13]. The research was divided into two parts:

1. developing a component model that will fulfill the requirements coming from the embedded systems domain
2. developing a model for describing component's and system's functional and extra functional behavior (such as timed behavior and resource consumption)

The rest of the proposal is organized as follows: in section 2, the related work is surveyed. In section 3, we formulate the research problem, define the research questions, summarize the contribution of the thesis and present the research methodology. In section 4 the planned contents of the thesis are outlined and finally in section 5 a time plan of completing the thesis is given.

2 Related Work

2.1 Component Models for Embedded Systems

The state of the art on component models suitable for development of embedded systems covers a considerable number of component models. Most of the component models that specifically target embedded systems focus primarily on "small" components that offer concrete functionality. Some of these component models were developed to target a particular domain in embedded systems such as: automation systems, vehicular systems or consumer electronics.

Our component model - ProCom (conducted as a part of the PROGRESS approach) was inspired by some of the previous works on SaveCCM [1] and Rubus component model [4]. SaveCCM is a component model specifically designed for embedded control applications in the automotive domain. ProCom has inherited some concepts from SaveCCM, such as emphasis on reusability and safety thanks to the strong restrictions on the proposed semantics and syntax, ability to analyze timed behavior and separation of data and control-flow. Rubus component model is intended for small resource-constrained embedded systems and timing aspects are the only extra-functional properties considered (such as release-time, deadline, worst-case execution-time and period-time). ProCom has been influenced by the time and event-triggering features of Rubus.

Koala [23] is a component model targeting embedded systems for consumer electronics. The requirements on safety and reliability in these kind of systems are not as high like in the vehicular domain. Therefore, real-time concerns are not taken into consideration, apparat from some aspects of static resource consumption (such as memory). Robocop [17] component model is an extension of the Koala component model. A main addition to Koala was the introduction of models associated with components and the aim of covering all the aspects of the CBD process. The models can have different form (e.g. textual or binary) and can model different properties related to the component.

PECOS [19] is a component model for small reactive embedded systems in the automation domain. PECOS considers extra-functional properties (e.g. timing and memory usage) in order to enable analysis. It allows specification of extra-functional properties, but does not provide support for analysis of these properties neither on the component nor on the system level.

Pin [18] component model is used as a basis in prediction-enabled technologies (PECTs). Pin does not provide support for system design and does not allow hierarchical component nesting.

2.2 Resource Modelling and Analysis for Embedded Systems

In [24], we have classified the related work on modeling and analyzing resources in embedded systems into three categories.

First, research has been dedicated to prediction of code-level resource consumption in component assemblies. In Koala [11] and Robocop [9] component models static memory estimation has been performed on concrete, reduced-sized applications and scenarios in which the instantiated components in a composition are known before run-time. In real-world applications, the set of components may dynamically change, so the estimation will only hold for a specific snapshot of components instantiated at that moment. More abstract description on expected resource usage may be needed for not-yet implemented components or for guiding the selection of already developed components from the repository.

Second, some UML-based attempts [12, 3] to tackle the analysis of embedded resources have been carried out. Nevertheless graphical and intuitive, these approaches lack a formal description that could enable thorough formal analysis.

Third, higher-level formal approaches [14, 15, 16] cover a family of process-algebraic formalisms established to unify formal modeling and analysis of embedded systems resources. The family is based on discrete-time process algebra of communicating shared resources (ACSR), that extends classical process algebra with the notion of a resource. Although, the ACSR framework is theoretically rich for resource modeling, the verification is independent of the design stage. Alternative formal approach, based on timed abstract state machines [20] considers resources as simple annotations, in the form of real-valued variable assignments. Consequently the framework can not support trade-off analysis of conflicting resource requirements.

3 Research Description

In this research, we have studied the state of the art on component models and resource modeling and analysis for embedded system. Also we have summarized the component model requirements coming from the embedded systems domain. According to these studies a two-layered component model - ProCom and a model for describing component's functional and extra functional behavior (such as timing and resources) -

REMES, have been developed. Additionally, for analysis purposes of resource consumption, REMES models have been semantically translated into Priced Timed Automata [6, 2].

3.1 Goal of the Research

The goal of this thesis is:

To incorporate the relevant resource types for embedded systems in a resource-aware formal component model amenable for automated formal reasoning and development of embedded systems.

3.2 Research Questions

The indispensable characteristic of CBD are the component models. A component model should define standards for properties that individual components must satisfy and methods, and possibly mechanisms, for composing components [7]. Nowadays, there is a wide spectrum of component models with different characteristics. This makes it difficult to properly understand the component-based principles. For this reason there is a need for a framework where different component models will be classified and compared. From this we state the first question:

Question 1. *What are the common characteristics and differences between existing component models?*

One of the central concepts in CBD, apart from component models are component technologies. Component technologies put component models into practise, in a sense that a particular component technology provides tools that enable development and deployment of systems that adhere to a corresponding component model.

CBD is an established approach for development of desktop and e-business applications, but this is not yet the case for embedded applications. An important reason for the limited success of the CBD in embedded systems domain is the inability of commercially available component technologies to provide solutions that meet typical embedded system requirements. In [25] it is discussed about which domain specific requirements a component technology targeting embedded system development should be aware of. Accordingly, embedded systems seek design architectures that can execute particular application's functions (protocols, signal processing, user interface, etc.) while meeting all domain requirements such as reliability, resource-efficiency, predictability etc. Embedded system designers should also strive to characterize both hardware and software with performance, resource consumption, and size requirements. Furthermore, in order to properly analyze embedded systems, designers should create a hierarchy of models that will alow them to reason about timed behavior, resource consumption and so on, without going down to to the instruction level. Also, embedded system developers must verify that applications meet their functional and extra-functional specification. All these requirements should be reflected in the component model. Thus a second question is raised:

Question 2. *How can a component model be designed to fulfill the requirements coming from the embedded systems domain?*

As already stated in section 1 one of the main characteristics of embedded systems is the restriction of available resources. In section 2.2 we have summarized the current state of the art on resource modeling and analysis for embedded systems. The diversity of approaches existing in the aforementioned literature indicate the complexity of gathering all relevant resource types for embedded systems in one uniformed theory. This calls for a fresh look on resource-aware design methods, based on the experience from the existing modeling approaches. In order to properly specify and analyze embedded systems, a modeling formalism should be developed that will incorporate the notation of a resource as a first-class entity. In addition, essential to any resource modeling technique is an extensive classification of resource classes and their characteristics. So, a number of resource class definitions that are useful for modeling embedded systems should be considered. Accordingly, the third research question can be formulated:

Question 3. *How can embedded system resources be classified and modeled?*

Component technologies for embedded systems should support development of systems with high dependability assurance. To guarantee that the combination of components fits on a particular target platform, an estimation of the consumed resources for that combination is necessary. As complexity grows, and dependability requirements get stronger, testing can no longer provide sufficient dependability assurance. Formal analysis of functionality, timeliness and resource usage are significant supplements to testing. The analysis techniques should provide algorithms for computing worst-case and best-case timing resource consumption. The increase in performance requirements for embedded systems calls for simultaneous efficient use of several resources (such as processors and memory). As such, for a given resource model, we may have more than one property to verify concurrently, and we want to know whether it is possible to satisfy all of them. Accordingly, the model should be able to perform trade-off analysis between apparently conflicting resource requirements. Taking all this into consideration, the forth research question which includes two sub-questions can be stated:

Question 4. *How can a component model be formalized to support uniformed analysis of functional and extra-functional behavior (such as timing and resource consumption)?*

In this research we focus on consumption of resources.

Question 4A. *How to predict the worst-case and best-case resource consumption of a component-based system?*

Question 4B. *How to carry out trade-off analysis of consumption of different resources?*

3.3 Contribution of the Thesis

The thesis will include six research papers which are related to each other according to Figure 1.

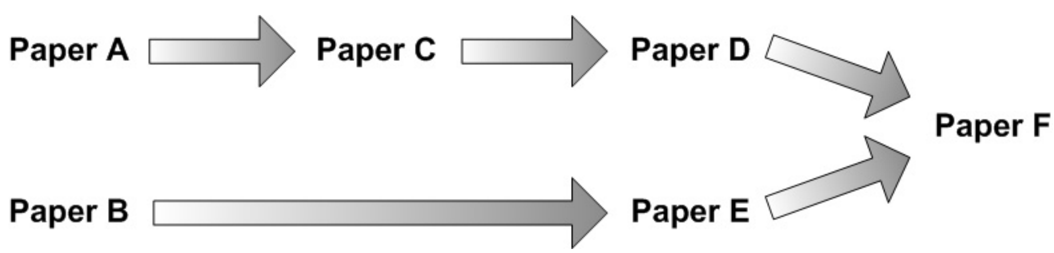

Figure 1: Relation of the six papers that will be included in the licentiate thesis.

The first paper, paper A gives an answer to question 1. It is a survey paper, where a four dimensional framework (lifecycle, constructs, extra-functional properties and domains) for classification and comparison of component models is introduced. It also gives a recommended set of characteristics that a model should fulfill in order to be considered as a component model. We have also mapped our classification framework on a significant number of component models and compared their characteristics.
Paper C and paper D answer the second research question by presenting the requirements placed on the ProCom component model developed to target embedded systems and the ProCom component model itself, respectively. A component model for embedded systems should fulfill the following requirements:

- complexity management

- coverage of both early-designed components (with very vague specification) and deployable components (with very concrete semantics)
- deal with different types of components with respect to their size, functionality and semantics
- provide support for different kind of analysis (functional behavior, timed behavior, resource usage etc.)
- manage the strong coupling between the components, the system and the targeted platform, while still distinguishing between the development of reusable components and a specific system.

ProCom takes into account these requirements and uses the concept of reusable components through the whole development process, from early design to deployment. It is structured in two layers (ProSys and ProSave) which address the different concerns that arise when modelling on one hand loosely coupled subsystems, and on the other hand small parts of control functionality of an individual subsystem.

Paper B gives a partial answer on question 3. It presents the current state of the art on resource modeling and analysis of embedded systems build out of components. In addition, in this paper we describe our own view of how to model and analyze embedded resources during the whole system lifecycle.

Paper E and paper F give an answer to question 4. In particular, paper E gives an answer as well to the third question and to the sub-questions 4A and 4B. It proposes a modeling framework REMES and associated analysis techniques for performing quantitative analysis such as best-case / worst-case resource consumption and trade-off analysis. REMES is tailored for embedded systems, but it is also suitable for reactive systems. It covers both discrete (such as memory) and continuous (such as power) abstract resources characterized further by the way they are consumed and released, and by whether they can be refereed to, or not. REMES can model a number of generic resources (memory, cpu, power, busses, access to external devices, etc) and the classification of the resources presented in paper E is not tied to any particular formal semantical presentation. We also present how REMES can be semantically translated into Priced Timed Automata for resource consumption analysis and we illustrate our approach on a case study. As such, REMES presents an intermediate model that narrows the gap between very abstract architectural modeling and very concrete semantical analysis models. REMES will be further used for describing the behavior of ProSave components and ProSys subsystems in paper F.

The overall illustration of how the included papers A-F contribute to the research questions is depicted in Table1.

Table 1: Illustration of how the included papers A-F contribute to the research questions

Research Questions / Papers	Q1	Q2	Q3	Q4	Q4A	Q4B
A	X					
B			X			
C		X				
D		X				
E			X	X	X	X
F				X		

3.4 Research Methodology

Different research methods are suitable for different settings, and similarly different validation techniques are suitable for different types of results. The methodology that has been used in this research is deductive and is based on the research steps presented in [21]. The main activities are:

1. Identification of the research problem from real-world software engineering.
2. Transferring the problem to a research setting, and defining the research questions. In this stage the research problem is often refined and narrowed down.
3. Analysis of the current state of the art addressing the research questions.

4. Answering the research questions and presenting the research results. This stage includes several iterations steps such as observations, discussions, analysis and refinement of the research results.
5. Validating whether the research results adequately answer the research questions. This can be performed in several different ways, e.g., by formal proofs, by performing case-studies, by implementation of a prototype, by describing experiences etc.
6. Validating whether the research results are feasible for the real-world software engineering problem.

Following the abovementioned activities to a great extent, we have initially defined the research problem, as stated in section 3.1. Second, from the research problem we have identified the research questions presented in section 3.2. Later we have conducted a thorough investigation of the current state of the art addressing the research questions. This investigation has resulted in writing two papers: paper A and paper B. Further in papers C, D and E (and as planned in paper F) we have presented our research results which are summarized in section 3.3. Since the research covered in this licentiate proposal is not mature enough, a complete validation of the results has not been performed yet, but the research results have been exemplified on "research examples" presented in papers A, D, and E. Accordingly, in paper A the classification framework was demonstrated on a considerable number of component models. In paper D we have exemplified the ProCom component model on an electronic stability control system of a car. Further in paper E, we have performed a small case study demonstrating the principles of our resource modeling and analysis approach. The case study has been conducted on an abstracted version of the internal design of a temperature control system for a heat producing reactor.

3.5 Included Papers

More details about the included papers are as follows:

Paper A. "A Classification Framework for Component Models". Ivica Crnković, Séverine Sentilles, Aneta Vulgarakis, Michel Chaudron. Submitted to IEEE Transactions on Software Engineering

Abstract: *The essence of component-based software engineering is embodied in component models. Component models specify the properties of components and the mechanism of component compositions. In last decade a rapid growth, a plethora of different component models has been developed, using different technologies, having different aims, and using different principles. This has resulted in a number of models and technologies which have many similarities, but also principal differences, and in a lot cases unclear concepts. Component-based development has not succeeded in providing standard principles, as for example object-oriented development. In order to increase the understanding of the concepts, and to easier differentiate component models, this paper provides a Component Model Classification Framework which identifies and discusses the basic principles of component models. Further the paper classifies a certain number of component models using this framework.*

This paper was written with equal contribution from the first three authors. I was responsible mainly for the lifecycle section and shared the responsibility with Séverine Sentilles for collecting information, analysing and classifying in tables the included component models.

Paper B. "Embedded Systems Resources: Views on Modeling and Analysis". Aneta Vulgarakis, Cristina Seceleanu. Proceedings of COMPSAC, the 1st IEEE International Workshop On Component-Based Design Of Resource-Constrained Systems Software and Applications Conference (CORCS), Turku, Finland, July, 2008.

Abstract: *The conflicting requirements of real-time embedded systems, e.g. minimizing memory usage while still ensuring that all deadlines are met at run-time, require rigorous analysis of the system's resource consumption, starting at early design stages. In this paper, we glance through several representative frameworks that model and estimate resource usage of embedded systems, pointing out advantages and limitations. In the end, we describe our own view on how to model and carry out formal analysis of* *embedded resources, along with developing the system.*

This paper was written with equal contribution from all the authors. I was specifically working on the code-level and UML- based resource modeling and analysis.

Paper C. ”A Component Model Family for Vehicular Embedded Systems”. Tomáš Bureš, Jan Carlson, Séverine Sentilles, Aneta Vulgarakis. Proceedings of the 3rd International Conference on Software Engineering Advances (ICSEA), IEEE, Sliema, Malta, October, 2008.

Abstract: *In this paper we propose to use components for managing the increasing complexity in modern vehicular systems. Compared to other approaches, the distinguishing feature of our work is using and benefiting from components throughout the development process from early design to development and deployment, and an explicit separation of concerns at different levels of granularity. Based on the elaboration of the specifics of vehicular systems (resource constraints, real-time requirements, hard demands on reliability), the paper identifies concerns that need to be addressed by a component model for this domain, and describes a realization of such a component model.*

This paper was written with equal contribution from all the authors. I took part in the discussions and contributed with writing and improving parts of the paper, particulary the related work section.

Paper D. ”A Component Model for Control-Intensive Distributed Embedded Systems”. Séverine Sentilles, Aneta Vulgarakis, Tomáš Bureš, Jan Carlson, Ivica Crnković. Proceedings of the 11th International Symposium on Component-Based Software Engineering (CBSE2008), Karlsruhe, Germany, October, 2008.

Abstract: *In this paper we focus on design of a class of distributed embedded systems that primarily perform real-time controlling tasks. We propose a two-layer component model for design and development of such embedded systems with the aim of using component-based development for decreasing the complexity in design and providing a ground for analyzing them and predict their properties, such as resource consumption and timing behavior. The two-layer model is used to efficiently cope with different design paradigms on different abstraction levels. The model is illustrated by an example from the vehicular domain.*

This paper was written with equal contribution from all the authors. The ProCom component model that we describe in this paper was developed in several iteration steps resulting from the conducted discussions between the authors.

Paper E. ”REMES: A Resource Model for Embedded Systems”. Cristina Seceleanu, Aneta Vulgarakis, Paul Pettersson. Submitted to RTAS 2009 (Real-Time and Embedded Technology and Applications Symposium)

Abstract: *In this paper, we introduce the model* REMES *for formal modeling and analysis of embedded resources such as storage, power, communication, and computation. The model is annotated with both discrete and continuous resources. It is in fact a state-machine based behavioral language with support for hierarchal modeling, continuous time, and a notion of explicit entry and exit points, making it suitable for component-based modeling. The analysis of* REMES*-based systems is centered around a weighted sum in which the variables represent the amounts of consumed resources. We describe a number of important resource related analysis problems, including feasibility, trade-off, and optimal resource-utilization analysis. To formalize these problems, and to provide a basis for formal analysis, we show how to analyze* REMES *models using the framework of priced timed automata and weighted CTL. To illustrate the approach, we describe a case study in which it has been applied to model and analyze resource usage of a temperature control system.*

This paper was written with equal contribution from all the authors. I particularly worked on the classification of the resources and specified, modeled in REMES and analyzed in UPPAAL CORA the TCS system presented as a case study in the paper.

Paper F. Semantics of ProCom based on REMES. (to be done)

I will be the main driver and principal author of this paper.

4 Thesis Outline

The proposed title of the thesis is "A Resource-Aware Component Model for Embedded Systems". The thesis will be a collection of six articles presented in 3.5, together with an introduction, a related work chapter, a contribution and summarizing discussions and a conclusion. The following outlines the chapters of the thesis:

4.1 Introduction

The introduction will present general background on embedded systems, component-based software engineering and general discussion on why are resources essential in this type of systems.

4.2 Research Summary

This chapter will contain a summary of the research methodology, questions, and contributions relevant to the thesis.

4.2.1 Research Methodology

The applied research methodology will be presented similar to section 3.4.

4.2.2 Problem Description

In this section the research problem will be described.

4.2.3 Research Questions

The research questions presented in 3.2 will be introduced.

4.2.4 Contribution

The contribution of the thesis will be clearly presented and mapped according to the research questions. This section will be a summary of the findings and conclusions from the included papers.

4.3 Related Work

This chapter will contain the current state of the art on component models for embedded systems and resource modeling and analysis in embedded systems. It will result in some duplication of information presented in papers A and B. Here, we will also position our research view according to the current state of the art.

4.4 Conclusions and Future Work

This chapter will summarize the thesis and propose directions for future research.

4.5 Included Papers

This chapter will include the six papers presented in 3.5.

5 Progress and Time Plan

The progress of the current research is:

Publications. Papers B, C and D have been accepted to international workshops or conferences. Paper A has been accepted to a local conference on Software Engineering Research and Practice in Sweden, however it has been extended and submitted to the journal of IEEE Transactions on Software Engineering (TSE). If it is not accepted for publication in TSE, it will be included in the thesis as a technical report. Paper E has been submitted to Real-Time and Embedded Technology and Applications Symposium and paper F is yet to be written. We plan to submit paper F to 35th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA - CBSE and SPPI track) in March 2009. If not accepted to EUROMICRO SEAA, it can be improved and submitted to Conference on Model Driven Engineering Languages and Systems (MODELS) 2009 or to Conference on Automated Software Engineering (ASE) 2009.

Courses. The requirement for the licentiate degree is at least 30 points from courses. This requirement is fulfilled, and a summary of finished and pending courses is shown below

Courses	**Credits**	**Status**
System Software	5	Completed
Distributed Computer Systems	5	Completed
Research Methodology for Natural Sciences and Engineering	5	Completed
PROGRESS: techniques and technologies	5	Completed
Advanced CBSE	5	Completed
Formal languages, automata and theory of computation	3	Completed
Research planning	3	Completed
Software Engineering	3	Completed
PhD school - Introduction course for research students	3	Completed
Modeling and Verification of Real Time Systems	5	Ongoing
Total Credits	**42**	

The goal is to have the licentiate seminar in June 2009. To complete the thesis, the main remaining activities are:

Writing and compiling the thesis. As described in section 3.5, five out of six of the papers have been written and published or submitted. Paper F is planned to be submitted to EUROMICRO SEAA. The other part of the thesis will be written based on the thesis contents as outlined in section 4, including an introduction, related work, problem description, conclusion and future work.

A preliminary time plan is as follows:

Number	Task	Start	Finish	Duration	Q4 08	Q1 09			Q2 09			Q3 09
					dec	jan	feb	mar	apr	maj	jun	jul
1	Thesis proposal presentaton	2008-12-15	2008-12-15	1d								
2	Paper F submitted to SEAA (CBSE & SPPI track)	2009-01-15	2009-03-16	52d								
3	Write first draft of the thesis	2009-01-15	2009-03-10	47d								
4	Complete thesis and send it to the opponent	2009-03-10	2009-04-10	28d								
5	Finish camera – ready copy of the thesis	2009-04-10	2009-05-09	26d								
6	Licentiate seminarium	2009-06-10	2009-06-10	1d								

Potential opponents are Tiziana Margaria (University Potsdam), Antonia Bertolino (Institute of Information Science and Technologies in Pisa) and Bengt Jonsson (Uppsala University).

References

[1] Mikael Åkerholm, Jan Carlson, Johan Fredriksson, Hans Hansson, John Håkansson, Anders Möller, Paul Pettersson, and Massimo Tivoli. The save approach to component-based development of vehicular systems. *Journal of Systems and Software*, 80(5):655–667, May 2007.

[2] Rajeev Alur. Optimal paths in weighted timed automata. In *In HSCC01: Hybrid Systems: Computation and Control*, pages 49–62. Springer, 2001.

[3] Hany H. Ammar, Vittorio Cortellessa, and Alaa Ibrahim. Modeling resources in a uml-based simulative environment. In *AICCSA*, pages 405–410, 2001.

[4] Arcticus Systems. Rubus Software Components. `www.arcticus-systems.com`.

[5] Christel Baier and Joost P. Katoen. *Principles of Model Checking - The MIT Press*.

[6] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. Minimum-Cost Reachability for Priced Timed Automata. In *Proceedings of the 4th International Workshop on Hybris Systems: Computation and Control*, number 2034 in Lecture Notes in Computer Sciences, pages 147–161. Springer–Verlag, 2001.

[7] Michel Chaudron and Ivica Crnkovic. Compnent-based software engineering. In Hans van Vliet, editor, *Software Engineering: Principles and Practice, 3rd Edition*, pages 605–628. Wiley, May 2008.

[8] Ivica Crnkovic and Magnus Larsson. *Building Reliable Component-Based Software Systems*. Artech House publisher, 2002.

[9] Merijn de Jonge, Johan Muskens, and Michel Chaudron. Scenario-based prediction of run-time resource consumption in component-based software systems. In *Proceedings of the 6th ICSE Workshop on Component-based Software Engineering (CBSE6)*, pages 19–24. IEEE, 2003.

[10] S. Edwards, L. Lavagno, E.A. Lee, and A. Sangiovanni-Vincentelli. Design of embedded systems: formal models, validation, and synthesis. *Proceedings of the IEEE*, 85(3):366–390, March 1987.

[11] Alexandre V. Fioukov, Evgeni M. Eskenazi, Dieter K. Hammer, and Michel R. V. Chaudron. Evaluation of static properties for component-based architectures. In *EUROMICRO*, pages 33–39, 2002.

[12] Object Management Group. Uml profile for schedulability, perfomance and time specification. *Version 1.1, formal / 05-01-02*, 2005.

[13] Hans Hansson, Mikael Nolin, and Thomas Nolte. Beating the automotive code complexity challenge. In *National Workshop on High-Confidence Automotive Cyber-Physical Systems*, Troy, Michigan, USA, April 2008.

[14] Insup Lee, Jin-Young Choi, Hee-Hwan Kwak, Anna Philippou, and Oleg Sokolsky. A family of resource-bound real-time process algebras. In *FORTE*, pages 443–458, 2001.

[15] Insup Lee, Anna Philippou, and Oleg Sokolsky. A general resource framework for real-time systems. In *RISSEF*, pages 234–248, 2002.

[16] Insup Lee, Anna Philippou, and Oleg Sokolsky. Resources in process algebra. *J. Log. Algebr. Program.*, 72(1):98–122, 2007.

[17] H. Maaskant. *A Robust Component Model for Consumer Electronic Products*, volume 3 of *Philips Research*, pages 167–192. Springer, 2005.

[18] Gabriel A. Moreno. Creating custom containers with generative techniques. In *GPCE '06: Proceedings of the 5th international conference on Generative programming and component engineering*, pages 29–38. ACM, 2006.

[19] Oscar Nierstrasz, Gabriela Arévalo, Stéphane Ducasse, Roel Wuyts, Andrew P. Black, Peter O. Müller, Christian Zeidler, Thomas Genssler, and Reinier van den Born. A component model for field devices. In *Proc. of the 1st Int. IFIP/ACM Working Conference on Component Deployment*, pages 200–209. Springer, 2002.

[20] Martin Ouimet, Kristina Lundqvist, and Mikael Nolin. The timed abstract state machine language: An executable specification language for reactive real-time systems. In *Proceedings of the 15th International Conference on Real-Time and Network Systems*, 2007.

[21] Mary M. Shaw. The coming-of-age of software architecture research. In *Proceedings of the 23rd International Conference on Software Engineering, ICSE 2001, Toronto, Ontario, Canada*, pages 656–664a. IEEE Computer Society.

[22] Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming*. ACM Press and Addison-Wesley, New York, NY, 1998.

[23] R. van Ommering, F. van der Linden, and J. Kramer. The Koala component model for consumer electronics software. In *IEEE Computer*, pages 78–85. IEEE, March 2000.

[24] Aneta Vulgarakis and Cristina Seceleanu. Embedded systems resources: Views on modeling and analysis. In *COMPSAC*, pages 1321–1328, 2008.

[25] Wayne Wolf. Embedded computing - what is embedded computing? *IEEE Computer*, 35(1):136–137, 2002.