# Embedded Systems Development: From Functional Models to Implementations

Alberto Sangiovanni-Vincentelli, Haibo Zeng, Marco Di Natale
and Peter Marwedel

Alberto Sangiovanni-Vincentelli
UC Berkeley, Berkeley, USA, e-mail: alberto@eecs.berkeley.edu

Haibo Zeng
McGill University, Montreal, Canada, e-mail: haibo.zeng@mcgill.ca

Marco Di Natale
Scuola Superiore Sant'Anna, Pisa, Italy, e-mail: marco.dinatale@sssup.it

Peter Marwedel
Technische Universität Dortmund, Dortmund, Germany, e-mail: peter.marwedel@tu-dortmund.de

# Contents

**Part III Component-Based Design and Real-Time Components**

**Part IV Timing Analysis and Time-Based Synthesis**



**List of Figures**

# Preface

This book is an edited collection of contributions in selected topics related to embedded systems modeling, analysis and synthesis. Most contributions are extended versions of papers that originally appeared throughout several workshops organized in the context of the Embedded Systems Week and Real-Time Systems Symposium in the last months of 2011. The workshops targeted topics and challenges related to the use of models for the design, analysis, and synthesis of Embedded Systems. Problems and solutions were discussed for different stages in the development process and apply to the system-level view, as well as to the design, analysis and synthesis of components and subsystems and the behaviors therein. These workshops were the *WSS, Workshop on Software Synthesis*, the *TiMoBD, Time Analysis and Model-Based Design*, and the *SOMRES, Workshop on Synthesis and Optimization Methods for Real-Time Embedded Systems*.

As workshop organizers and editors of this book, we believe that these are very special times for embedded and cyber-physical systems researchers and developers. A time of opportunity, because of the need and emergence of feature-rich, complex, distributed systems and the need to tame their complexity in new ways, including the adoption of model-based development, new analysis methods and design synthesis techniques, and true component-based development, in which functional and platform assemblies are correct-by-construction.

This book collects contributions on different topics, including system and software models, innovative architectures (including OS and resource managers), formal methods, model checking and analysis techniques, software synthesis, system optimization and real-time networks, with the ambitious objective of providing useful insights and innovative ideas on how to solve very complex problems throughout the entire (model-based) development cycle. Contrary to other books on the subject, we attempt at reconciling the two communities of Model-Based and Model-Driven Design, which often operate in independent ways, with only a few fortunate exceptions.

Regardless of the workshop organization, the selected paper have been organized according to their topics and divided in chapters that better fit the stages in the development process rather than an abstract classification based, for example, on lan-

guages, algorithmic solutions or methods. The intended audience includes of course the general community of embedded systems researchers, but we believe several topics and contributions should be also of interest for developers, tool vendors and development process experts. Several contributions are provided by industry developers and researchers and refer to upcoming commercial products, methods and tools. The applicability of most other results is demonstrated by use cases and/or project experiences.

We would like to acknowledge all authors, the workshop audiences that provided constructive feedback and interesting discussion, which eventually found their way into improved and new content and the assistant editors at Springer.

Berkeley, Montreal, Pisa, Dortmund,
March 2013

*Alberto Sangiovanni-Vincentelli,*
*Haibo Zeng,*
*Marco Di Natale,*
*Peter Marwedel*

# Chapter 8
# Contract-Based Reasoning for Component Systems with Rich Interactions

Susanne Graf, Roberto Passerone and Sophie Quinton

**Abstract** In this paper we propose a rule unifying circular and non-circular assume-guarantee reasoning and show its interest for contract-based design and verification. Our work was motivated by the need to combine, in the top-down methodology of the FP7 SPEEDS project, partial tool chains for two component frameworks derived from the HRC model and using different refinement relations. While the L0 framework is based on a simple trace-based representation of behaviors and uses set operations for defining refinement, the more elaborated L1 framework offers the possibility to build systems of components with complex interactions. Our approach in L1 is based on circular reasoning and results in a method for checking contract dominance which does not require the explicit composition of contracts. In order to formally relate results obtained in L0 and L1, we provide a definition of the minimal concepts required by a consistent contract theory and propose abstract definitions which smoothly encompass hierarchical components. Finally, using our relaxed rule for circular reasoning, we show how to use together the L0 and L1 refinement relations and as a result their respective tool chains.

Susanne Graf
VERIMAG / CNRS, 2 avenue de Vignate, 38610 Gières, France.
e-mail: susanne.graf@imag.fr

Roberto Passerone
DISI / University of Trento, via Sommarive 5, 38123 Trento, Italy.
e-mail: roberto.passerone@unitn.it

Sophie Quinton
IDA / TU Braunschweig, Hans-Sommer-Straße 66, 38106 Braunschweig, Germany.
e-mail: quinton@ida.ing.tu-bs.de

## 8.1 Introduction

Contract and interface frameworks are emerging as the formalism of choice for system designs that require large and distributed teams, or where the supply chain is complex [242, 64, 65]. This style of specification is typically employed for top-down design of systems of components, where the system under design is built by a sequence of decomposition and verification steps. In this paper we present and study some distinctive features of contract theories for frameworks in which the interaction between components is "rich", i.e., more complex than the usual input/output (I/O) communication. One such component framework is BIP [26] which allows multi-party synchronizations scheduled according to priorities. In addition, we show how to combine results obtained using different contract refinement relations.

Our work has its practical motivation in the component framework HRC [29, 32, 64, 65] (standing for Heterogeneous Rich Components) defined in the FP7 IP project SPEEDS [255], which has been reused in the FP7 STREP project COMBEST [60] and the ARTEMIS project CESAR [51]. The HRC model defines component properties in terms of extended transition systems and provides several composition models, ranging from low-level semantic composition to composition frameworks underlying the design tools used by system designers. More precisely, HRC is organized around two abstraction levels called L0 and L1 and describing respectively the *core level* and the *analysis tool level* of HRC [210]. That is, L0 determines the expressive power of the entire model and there exist translations from L1 models to L0. On the other hand, L1 extends the core model with concepts such as coordination mechanisms — the rich interactions mentioned in the title. Analysis tools can then take advantage of these additional concepts to make system descriptions more concise and therefore verification more efficient.

Our objective is to allow combined use of synchronous tools like Simulink [269] for L0 and synchronization-based tools like BIP for L1, which have complementary strengths. For example, Simulink is very convenient for modeling physical dynamic systems or streaming applications. In contrast BIP, which encompasses rich interactions, is well adapted for describing the dynamic behavior of sets of components depending on available resources for memory, energy, communication bandwidth etc. We are interested in this paper in the relation between the L0 and L1 contract frameworks as we want to use verification results established in L1 for further reasoning within L0. The presence of rich interactions in L1 makes contract composition problematic and leads us to focus instead on *circular reasoning*, which allows a component and its environment to be refined concurrently — each one relying on the abstract description of its context — and entails an interesting rule for proving *dominance*, i.e., refinement between contracts. In order to relate L0 and L1, we define a generic contract framework that uses abstract composition operators and thus encompasses a variety of interaction models, including those for L0 and L1. Finally, we show how to use a relaxed rule for circular reasoning to combine partial tool chains for both frameworks into a complete tool chain for our methodology.

To the best of our knowledge, this is the first time that a rule combining different refinement relations is proposed and used to unify two contract frameworks.

While circular reasoning has been extensively studied, e.g. in [6, 173], existing work focuses on finding sufficient conditions for soundness of circular reasoning while we focus on how to use circular reasoning in a contract-based methodology. Non-circular assume-guarantee reasoning is also a topic of intense research focused on finding a decomposition of the system that satisfies the strong condition imposed on at least one of its components [59]. Finally, our contract frameworks are related to interface automata [2]. Since de Alfaro and Henzinger's seminal paper many contract and interface theories have been developed for numerous frameworks (see e.g. [153, 282, 71, 229, 231, 230] to name just a few). However these theories focus on composition of contracts while we strive to avoid that and furthermore they do not handle rich interactions. Examples include [154, 230] based on modal I/O automata and [282] defining relational interfaces for capturing functional dependencies between inputs and outputs of an interface. Preliminary versions of our contract framework appeared in [224, 109] but did not address the question of combining results obtained for different refinements.

This paper is structured as follows: Section 8.2 describes our design and verification methodology as well as generic definitions of component and contract framework. It then discusses sufficient reasoning rules for establishing dominance without composing contracts. Section 8.3 presents how the proposed approach is applied to the L0 and L1 frameworks. In particular it shows how their different satisfaction relations may be used together using *relaxed circular reasoning* and discusses practical consequences of this result. Section 8.4 concludes the paper. The proofs of all theorems presented in this paper are presented in [102].

## 8.2 Design Methodology

Our methodology is based on an abstract notion of component. We characterize a component $K$ by its *interface* defined as a set $\mathscr{P}$ of *ports* which describe what can be observed by its environment. We suppose given a global set of ports *Ports*, which all sets of ports in the following are subsets of. In addition, components are also characterized by their *behavior* . At this level of abstraction, we are not concerned with how behaviors are represented and develop our methodology independently of the particular formalism employed. Interactions (potentially complex) between components are expressed using the concept of *glue* operator [254] . A glue defines how the ports of different components are connected and the kind of synchronization and data exchange that may take place. We denote the composition of two components $K_1$ and $K_2$ through a glue $gl$ as $gl\{K_1, K_2\}$. The glue must be defined on the union of the ports $\mathscr{P}_1$ and $\mathscr{P}_2$ of the components.

In order to separate the implementation phase of a component from its integration into the system under design, we use *contracts* [32, 30, 224] . A contract for a component $K$ describes the interface $\mathscr{P}$ of $K$, the interaction between $K$ and its environment $E$, the expected behavior of $E$, called the *assumption* $A$ of the contract, and the expected behavior of $K$, called the *guarantee* $G$. Assumptions and guarantees

are in turn expressed as components, defining the interface and the behavior that are considered acceptable from the environment and from the component. Thus, formally, a contract $\mathscr{C}$ for an interface $\mathscr{P}$ is a triple $(A, gl, G)$, where $gl$ is a glue operator on $\mathscr{P} \cup \mathscr{P}_A$ for some $\mathscr{P}_A$ disjoint from $\mathscr{P}$; the assumption $A$ is a component with interface $\mathscr{P}_A$; and the guarantee $G$ is a component with interface $\mathscr{P}$. Note that the interface of the environment is implicitly defined by $gl$. Graphically, we represent contracts as in Figure 8.1.

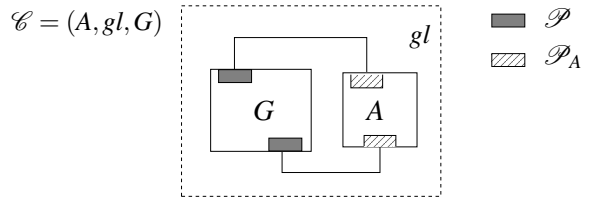


Fig. 8.1: A contract $(A, gl, G)$ for an interface $\mathscr{P}$

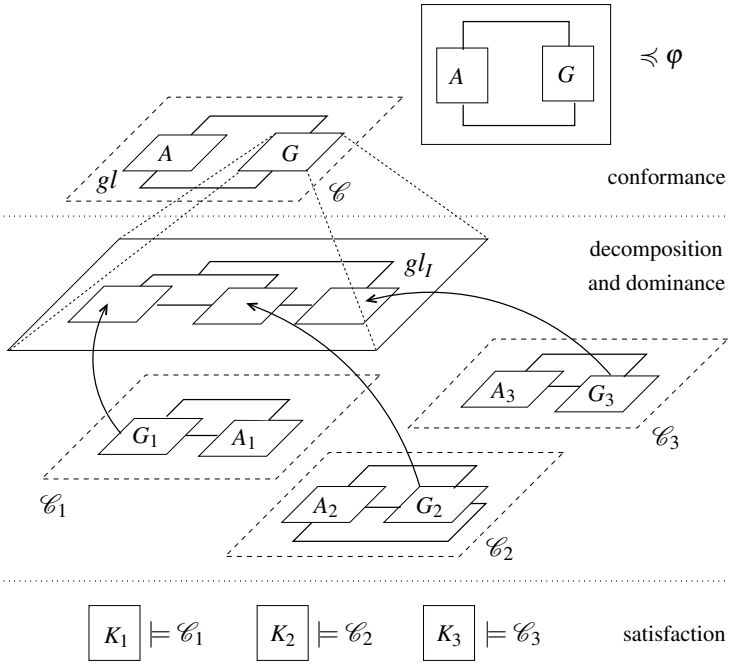


Fig. 8.2: Proof of $gl\{A, gl_I\{K_1, K_2, K_3\}\} \preccurlyeq \varphi$

From a macroscopic point of view, we adopt a top-down design and verification methodology (see Figure 8.2) in which global requirements are pushed progressively from the top-level system to the low-level atomic components. As usual, this is just a convenient representation; in real life, the final picture is always obtained in several iterations alternatively going up and down the hierarchy [213]. While the refinement relation between a specification and an implementation is at the core of component-based design, in contract-based design refinement takes different forms depending on whether it relates a system to a specification, two contracts or an implementation to a contract. In this paper we use a methodology which divides the design and verification process into three steps corresponding to these three forms of refinement.

We assume that the system $K$ under construction has to realize a global requirement $\varphi$ together with an environment on which we may have some knowledge, expressed by a property $A$. Both $\varphi$ and $A$ are expressed w.r.t. the interface $\mathscr{P}$ of $K$. We proceed as follows: (1) define a *contract* $\mathscr{C} = (A, gl, G)$ for $\mathscr{P}$ such that $gl\{A, G\}$ *conforms* to $\varphi$; (2) decompose $K$ as subcomponents $K_i$ connected through a glue operator $gl_I$ and provide a contract $\mathscr{C}_i$ for each of them; possibly iterate this step if needed; (3) prove that whenever a set of implementations $K_i$ *satisfy* their contracts $\mathscr{C}_i$, then their composition satisfies the top-level contract $\mathscr{C}$ (*dominance*) — and thus guarantee $\varphi$; (4) provide such implementations.

The correctness proof for a particular system is therefore split into 3 phases: *conformance* (denoted $\preccurlyeq$) of the system defined by the top-level contract $\mathscr{C}$ to $\varphi$; *dominance* of $\mathscr{C}$ by the composition of the set of contracts $\{\mathscr{C}_i\}$ through $gl_I$; and *satisfaction* (denoted $\models$) of each $\mathscr{C}_i$ by the corresponding implementation $K_i$. Thus, *conformance* relates closed systems, *dominance* relates contracts, while *satisfaction* relates components to contracts.

The assumption of $\mathscr{C}_1$ is represented as one component $A_1$ while in the actual system $K_1$ will be used in the context of three components, namely $K_2$, $K_3$ and $A$. Thus, we need to relate the actual glues $gl$ and $gl_I$ to the glue $gl_1$ of $\mathscr{C}_1$. In other words, we need a glue $gl_{E_1}$ to compose $K_2$, $K_3$ and $A$ as well as an operation $\circ$ on glues such that $gl \circ gl_I = gl_1 \circ gl_{E_1}$. In most cases, $\circ$ cannot simply be composition of functions and has to involve some *flattening* of the system.

### 8.2.1 Contract Framework

To summarize, we consider a component framework that smoothly supports complex composition operators and hierarchical components. The elements of the component framework are as follows:

**Definition 1 (Component framework).** A *component framework* is defined by a tuple $(\mathscr{K}, GL, \circ, \cong)$ where:

- $\mathscr{K}$ is a set of *components*. Each component $K \in \mathscr{K}$ has as *interface* a set of *ports*, denoted $\mathscr{P}_K$ and subset of our global set of ports *Ports*.

- $GL$ is a set of *glues*. A glue is a partial function $2^{\mathscr{K}} \longrightarrow \mathscr{K}$ transforming a set of components into a new composite component. Each $gl \in GL$ is defined on a set of ports $S_{gl}$, called *support set*, and defines a new interface $\mathscr{P}_{gl}$ for the new component, called *exported interface*. $K = gl(\{K_1, \ldots, K_n\})$ is defined if $K_1, \ldots, K_n \in \mathscr{K}$ have disjoint interfaces, $S_{gl} = \bigcup_{i=1}^{n} \mathscr{P}_{K_i}$ and $\mathscr{P}_K = \mathscr{P}_{gl}$.
- $\circ$ is a partial operator on $GL$, called *flattening*, to compose glues. $gl \circ gl'$ is defined if $\mathscr{P}_{gl'} \subseteq S_{gl}$. Its support set is $S_{gl} \backslash \mathscr{P}_{gl'} \cup S_{gl'}$ and its interface is $\mathscr{P}_{gl}$.
- $\cong \subseteq \mathscr{K} \times \mathscr{K}$ is an equivalence relation between components.

We simplify our notation by writing $gl\{K_1, \ldots, K_n\}$ instead of $gl(\{K_1, \ldots, K_n\})$. The equivalence relation $\cong$ is typically used for relating composite components with their semantics given as an atomic component. More importantly, $\circ$ must be coherent with $\cong$ in the sense that $gl\{gl'\{\mathscr{K}_1\}, \mathscr{K}_2\} \cong (gl \circ gl')\{\mathscr{K}_1 \cup \mathscr{K}_2\}$ for any sets of components $\mathscr{K}_i$ such that all terms are defined.

After formalizing generic properties required from a component framework, we now define the relations used in the methodology for dealing with contracts. Satisfaction is usually considered as a derived relation and chosen as the weakest relation implying conformance and preserved by composition. We loosen the coupling between satisfaction and conformance to obtain later stronger reasoning schemata for dominance. Furthermore, we propose a representation of satisfaction as a set of *refinement under context* relations denoted $\sqsubseteq_{A,gl}$ and such that $K \sqsubseteq_{A,gl} G$ iff $K \models (A, gl, G)$.

**Definition 2 (Contract framework).** A *contract framework* is defined by a tuple $(\mathscr{K}, GL, \circ, \cong, \preccurlyeq, \models)$ where:

- $(\mathscr{K}, GL, \circ, \cong)$ is a component framework.
- $\preccurlyeq \subseteq \mathscr{K} \times \mathscr{K}$ is a preorder called *conformance* relating components having the same interface.
- $\models$ is a relation called *satisfaction* between components and contracts s.t.: the relations $\sqsubseteq_{A,gl}$ defined by $K \sqsubseteq_{A,gl} G$ iff $K \models (A, gl, G)$ are preorders; and, if $K \models (A, gl, G)$ then $gl\{A, K\} \preccurlyeq gl\{A, G\}$.

Our definition of satisfaction emphasizes the fact that $\models$ can be seen as a set of refinement relations where $K \sqsubseteq_{A,gl} G$ means that $K$ refines $G$ in the context of $A$ and $gl$. The condition which relates satisfaction and conformance ensures that the actual system $gl\{A, K\}$ will conform to the global requirement $\varphi$ discussed in the methodology because $\preccurlyeq$ is transitive and $gl\{A, G\} \preccurlyeq \varphi$.

*Example 1.* Typical notions of conformance for labeled transition systems are *trace inclusion* and its structural counterpart *simulation*. For these, satisfaction is usually defined as the weakest relation implying conformance.

$$K \models (A, gl, G) \triangleq gl\{K, A\} \preccurlyeq gl\{G, A\}$$

Dominance is a key notion for reasoning about contracts rather than using refinement between components. Proving that a contract $\mathscr{C}$ dominates $\mathscr{C}'$ means showing

that every component satisfying $\mathscr{C}$ also satisfies $\mathscr{C}'$.[1] However, a dominance check involves in general not just a pair of contracts: a typical situation would be the one depicted in Figure 8.2, where a set of contracts $\{\mathscr{C}_i\}_{i=1}^n$ are attached to disjoint interfaces $\{\mathscr{P}_i\}_{i=1}^n$. Besides, a glue $gl_I$ is defined on $P = \bigcup_{i=1}^n \mathscr{P}_i$ and a contract $\mathscr{C}$ is given for $P$. In this context, a set of contracts $\{\mathscr{C}_i\}_{i=1}^n$ dominates a contract $\mathscr{C}$ w.r.t. a glue $gl_I$ if any set of components satisfying contracts $\mathscr{C}_i$, when composed using $gl_I$, makes a component satisfying $\mathscr{C}$.

**Definition 3 (Dominance).** Let $\mathscr{C}$ be a contract on $\mathscr{P}$, $\{\mathscr{C}_i\}_{i=1}^n$ a set of contracts on $\mathscr{P}_i$ and $gl_I$ a glue such that $S_{gl_I} = \bigcup_{i=1}^n \mathscr{P}_i$ and $\mathscr{P} = \mathscr{P}_{gl_I}$. Then $\{\mathscr{C}_i\}_{i=1}^n$ *dominates* $\mathscr{C}$ *with respect to* $gl_I$ iff for all components $\{K_i\}_{i=1}^n$:

$$(\forall i : K_i \models \mathscr{C}_i) \implies gl_I\{K_1, \ldots, K_n\} \models \mathscr{C}$$

Note that this formal definition of dominance does not help establishing dominance in practice because looking at all possible components satisfying a contract is not realistic. What we need is a sufficient condition that refers to assumptions and guarantees, rather than components. One such condition is when the composition of the low-level guarantees $G_i$ satisfies the top-level contract $\mathscr{C}$ and furthermore each low-level assumption $A_i$ is discharged by the abstraction of its environment defined by the guarantees of the other components. Formally:

$$\begin{cases} gl_I\{G_1, \ldots, G_n\} \models \mathscr{C} \\ \forall i : gl_{E_i}\{A, G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_n\} \models \mathscr{C}_i^{-1} \end{cases} \tag{8.1}$$

where for any contract $\mathscr{C}_i = (A_i, gl_i, G_i)$ we use the notation $\mathscr{C}_i^{-1}$ to denote the contract $(G_i, gl_i, A_i)$.

In the next subsection, we provide two rules which indeed make the previous condition sufficient for establishing dominance: one is similar to circular assume-guarantee reasoning and the other one deals with preservation of satisfaction by composition. This result is particularly significant because one can check dominance while avoiding composition of contracts, which is impossible in the general case and leads to state explosion in most concrete contract frameworks.

### *8.2.2 Reasoning within a Contract Framework*

We use here the representation of satisfaction as a set of refinement under context relations $\sqsubseteq_{A,gl}$ where $K \sqsubseteq_{A,gl} G$ if and only if $K \models (A, gl, G)$. The usual non-circular assume-guarantee rule reads as follows in our context:

[1] One may also need to ensure that the assumptions of the low-level contracts are indeed satisfied in the actual system. This is achieved by strengthening the definition with:

$$\forall E \text{ on } \mathscr{P}_A, \text{ if } E \models (G', gl', A') \text{ then } E \models (G, gl, A)$$

$$K \sqsubseteq_{A,gl} G \wedge E \sqsubseteq A \implies K \sqsubseteq_{E,gl} G$$

where $E \sqsubseteq A$ denotes that for any component $G$ and $gl$ such that $\sqsubseteq_{G,gl}$ is defined $E \sqsubseteq_{G,gl} A$. This rule relates the behavior of $K$, when composed with the abstract environment $A$, to the behavior of $K$, when composed with its actual environment $E$. However it is quite limited as it imposes a very strong condition on $E$. Hence the following rule which is commonly referred to as *circular reasoning*.

$$K \sqsubseteq_{A,gl} G \wedge E \sqsubseteq_{G,gl} A \implies K \sqsubseteq_{E,gl} G$$

Note that $E$ and $K$ may symmetrically rely on each other. For a given contract framework, this property can be proven by an induction based on the semantics of composition and refinement. Unfortunately, circular reasoning is not sound in general. In particular it does not hold for parallel composition with synchronizations (as in Petri nets or process algebras) or instantaneous mutual dependencies between inputs and outputs (as in synchronous formalisms). The following example illustrates one possible reason for the non validity of circular reasoning[2].

*Example 2.* Consider a contract framework where components are labeled transition systems and composition is strong synchronization between corresponding labels and interleaving of others denoted $\|$. Define conformance as simulation and satisfaction as the usual relation defined in Example 1. The circular reasoning rule translates into: if $K \| A$ is simulated by $G \| A$ and $E \| G$ is simulated by $A \| G$ then $K \| E$ is simulated by $G \| E$. In the example of Figure 8.3, both $G$ and $A$ forbid a synchronization between $b_K$ and $b_E$ from occurring. This allows their respective refinements according to $\sqsubseteq^{\preccurlyeq}$, namely $K$ and $E$, to offer respectively $b_K$ and $b_E$, since they can rely on respectively $G$ and $A$ to forbid their actual occurrence. But obviously, the composition $K \| E$ now allows a synchronization between $b_K$ and $b_E$.

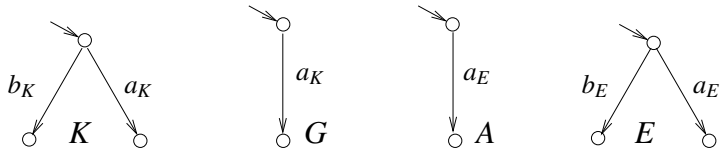


Fig. 8.3: $K \| A \preccurlyeq G \| A$ and $E \| G \preccurlyeq A \| G$ but $K \| E \not\preccurlyeq G \| E$.

Note that this satisfaction relation can be strengthened to obtain a more restrictive relation for which circular reasoning is sound. This is the approach taken for the L1 contract framework in Section 8.3.2, where we need circular reasoning to avoid composition of contracts.

[2] Note that non-determinism is another reason here for the non validity of circular reasoning.

A second rule which is used for compositional reasoning in most frameworks is: if $I \sqsubseteq S$, then $I \parallel E \sqsubseteq S \parallel E$.It states that if an implementation $I$ refines its specification $S$ then it refines it in any environment $E$. The equivalent of this rule for satisfaction is more complex as refinement here relates closed systems.

**Definition 4.** Satisfaction $\models$ is preserved by composition iff for any component $E$, $gl$ such that $S_{gl} = \mathscr{P}_E \cup \mathscr{P}$ for some $\mathscr{P}$ such that $\mathscr{P} \cap \mathscr{P}_E = \emptyset$ and $gl_E, E_1, E_2$ such that $E = gl_E\{E_1, E_2\}$, the following holds for any components $I$, $S$ on $\mathscr{P}$:

$$I \sqsubseteq_{E,gl} S \implies gl_1\{I, E_1\} \sqsubseteq_{E_2,gl_2} gl_1\{S, E_1\}$$

where $gl_1$ and $gl_2$ are such that $gl \circ gl_E = gl_2 \circ gl_1$.

We now have the ingredients to formalize our sufficient condition for dominance. This condition reduces a dominance proof to a set of satisfaction checks, one for the refinement between the guarantees and $n$ for discharging individual assumptions.

**Theorem 1.** *Suppose that circular reasoning is sound and satisfaction is preserved by composition. If* $\forall i\ \exists gl_{E_i} :\ gl \circ gl_I = gl_i \circ gl_{E_i}$ *then to prove that* $\{\mathscr{C}_i\}_{i=1}^n$ *dominates* $\mathscr{C}$ *w.r.t. gl, it is sufficient to prove that condition* (1) *holds.*

## 8.3 Circular Reasoning in Practice

In this section, we show how the results presented in the previous section have been applied within the SPEEDS project: we define two contract frameworks, called L0 and L1, and show how to combine them.

### *8.3.1 The L0 Framework*

A component $K$ with interface $\mathscr{P}_K$ at level L0 of HRC is defined as a set of behaviors in the form of traces, or runs, over $\mathscr{P}_K$. The behaviors correspond to the history of values seen at the ports of the component for each particular behavior. For instance, these histories could be the traces generated by a labeled transition system (LTS). Composition is defined as a composite that retains only the matching behaviors of the components. If the ports of the two components have the same names, composition at the level of trace sets boils down to a simple intersection of the sets of behaviors. Because in our framework components must have disjoint sets of ports under composition, we must introduce glues, or connectors, as explicit components that establish a synchronous relation between the histories of connected ports. The collection of these simple connectors forms the glues $gl \in GL$ of our framework at the L0 level.

We can model a glue as an extra component $K_{gl}$, whose set of ports includes all the ports of the components involved in the composition. This component has as

set of behaviors all the identity traces. Composition can then be taken as the intersection of the sets of behaviors of the components, together with the glue. To make this work, we must also equalize the ports of all trace sets using inverse projection $\mathbf{proj}^{-1}_{\mathscr{P}_i,\mathscr{P}}$, which extends behaviors over $\mathscr{P}_1$ with the appropriate additional ports of $\mathscr{P}$. If we denote the interface of the composite as $\mathscr{P}_{gl}$, and if $\mathscr{K} = \{K_1, \ldots, K_n\}$ is a set of components such that $\mathscr{P}_1, \ldots, \mathscr{P}_n$ are pairwise disjoint, then a glue $gl$ for $\mathscr{K}$ is a component $K_{gl}$ defined on the ports $\mathscr{P} = \mathscr{P}_{gl} \cup (\bigcup_{i=1}^{n} \mathscr{P}_i)$, and:

$$\begin{aligned} K &= gl\{K_1, \ldots, K_n\} \\ &= \mathbf{proj}_{p_{gl},\mathscr{P}} \left( K_{gl} \cap \mathbf{proj}^{-1}_{\mathscr{P}_1,\mathscr{P}}(K_1) \cap \cdots \cap \mathbf{proj}^{-1}_{\mathscr{P}_n,\mathscr{P}}(K_n) \right) \end{aligned}$$

The definition of $\circ$ is straightforward: since glues are themselves components, their composition follows the same principle as component composition. Finally, the $\cong$ relation on $\mathscr{K}$ is taken as equality of sets of traces.

In the L0 model there exists a unique maximal component satisfying a contract $\mathscr{C}$, namely $M_{\mathscr{C}} = G \cup \neg A$, where $\neg$ denotes the operation of complementation on the set of all behaviors over ports $\mathscr{P}_A$. A contract $\mathscr{C} = (A, G)$ is in *canonical form* when $G = M_{\mathscr{C}}$. Every contract has an equivalent contract in canonical form, which is obtained by replacing $G$ with $M_{\mathscr{C}}$. The operation of computing a canonical form is well defined, since the maximal implementation is unique, and it is idempotent. It is easy to show that $K \models \mathscr{C}$ if and only if $K \subseteq M_{\mathscr{C}}$.

The L0 contract framework has strong compositional properties, which derive from its simple definition and operators [30]. The theory, however, depends on the effectiveness of certain operators, complementation in particular, which are necessary for the computation of canonical forms. While the complete theory can be formulated without the use of canonical forms, complementation remains fundamental in the definition of contract composition, which is at the basis of system construction. Circular reasoning is not sound for contracts which are *not* in canonical form (Example 2 is a counter-example in that case). This is a limitation of the L0 framework, since working with canonical forms could prove computationally hard.

### *8.3.2 The L1 Framework*

L1 composition is based on *interactions*, which involve non-empty sets of ports. An interaction is defined by the components that synchronize when it takes place and the ports through which these components synchronize. Interactions are structured into connectors which are used as a mechanism for *encapsulation*: only these connectors appear at the interface of a composite component. This enables to abstract the behavior of a component in a black-box manner, by describing which connector is triggered but not exactly which interaction takes place. Furthermore L1 is expressive enough to encompass synchronous systems.

**Definition 5.** An *atomic component* on a interface $\mathscr{P}$ is defined by an LTS $K = (Q, q^0, 2^{\mathscr{P}}, \longrightarrow)$, where $Q$ is a set of states, $q^0$ an initial state and $\longrightarrow \subseteq Q \times 2^{\mathscr{P}} \times Q$ is a transition relation.

Note that atomic components are labeled by sets of ports rather than ports because we allow several ports of a component to be triggered at the same time.

**Definition 6.** An *interaction* is a non-empty set of ports. A *connector* $\gamma$ is defined by a set of ports $S_\gamma$ called the *support set* of $\gamma$, a port $p_\gamma$ called its *exported port* and a set $\mathscr{I}(\gamma)$ of interactions in $S_\gamma$.

The notions of support set and exported port are illustrated in Figure 8.4, where connectors relate in a composition a set of inner ports (of the subcomponents) to an outer port (of the composite component). One should keep in mind that a connector $\gamma$, and thus the exported port $p_\gamma$, represents a set of interactions rather than a single interaction.

Typical connectors represent rendezvous (only one interaction, equal to the support set), broadcast (all the interactions containing a specific port called *trigger*) and also mutual exclusion (some interactions but not their union).

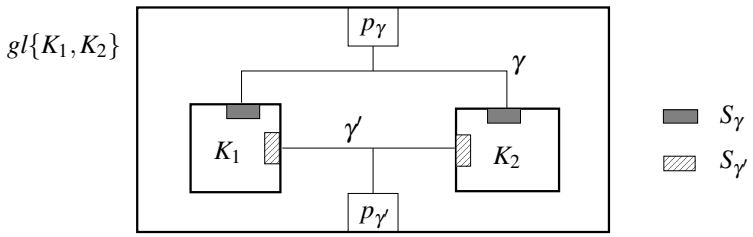


Fig. 8.4: The role of connectors in a composition

We now define glues as sets of connectors which may be used together in order to compose components.

**Definition 7.** A glue $gl$ on a support set $S_{gl}$ is a set of connectors with distinct exported ports and with support sets included in $S_{gl}$.

A glue $gl$ defines as exported interface $\mathscr{P}_{gl}$ the set $\{p_\gamma \mid \gamma \in gl\}$. Besides, $\mathscr{I}(gl)$ denotes the set of all interactions of the connectors in $gl$, i.e.: $\mathscr{I}(gl) = \bigcup_{\gamma \in gl} \mathscr{I}(\gamma)$. In Figure 8.4, $gl$ is composed of connectors $\gamma$ and $\gamma'$ and defines a composite component denoted $gl\{K_1, K_2\}$.

**Definition 8.** A *component* is either an atomic component or it is inductively defined as the composition of a set of components $\{K_i\}_{i=1}^n$ with disjoint interfaces $\{\mathscr{P}_i\}_{i=1}^n$ using a glue $gl$ on $\mathscr{P} = \bigcup_{i=1}^n \mathscr{P}_i$. Such a composition is called a *composite component* on $\mathscr{P}_{gl}$ and it is denoted $gl\{K_i\}_{i=1}^n$.

So far, we have defined components and glues. Glues can be composed so as to allow flattening of components. Such a composition requires to handle *hierarchical connectors* built by merging connectors defined at different levels of hierarchy. The definition of the operator $\circ$ used for this purpose is omitted here and can be found in [102]. Connectors whose exported ports and support sets are not related are called *disjoint* and need not be composed. The operator $\circ$ is then easily extended to glues: the composition $gl \circ gl'$ of two glues $gl$ and $gl'$ is obtained from $gl \cup gl'$ by inductively composing all connectors which are not disjoint.

We can now formally define the flattened form of a component. This in turn will allow us to provide an equivalence relation between components based on the semantics of their flattened form. A component is called *flat* if it is atomic or of the form $gl\{K_1,\ldots,K_n\}$, where all $K_i$ are atomic components. A component that is not flat is called *hierarchical*. A hierarchical component $K$ is of the form $gl\{K_1,\ldots,K_n\}$ such that at least one $K_i$ is composite. Thus, such a $K$ can be represented as $gl\{gl'\{\mathscr{K}^1\},\mathscr{K}^2\}$, where $\mathscr{K}^1$ and $\mathscr{K}^2$ are sets of components.

**Definition 9.** The *flattened form* of a component $K$ is denoted $\mathit{flat}(K)$ and defined inductively as:

- if $K$ is a flat component, then $\mathit{flat}(K)$ is equal to $K$.
- otherwise, $K$ is of the form $gl\{gl'\{\mathscr{K}^1\},\mathscr{K}^2\}$, and then $\mathit{flat}(K)$ is the flattened form of $(gl \circ gl')\{\mathscr{K}^1 \cup \mathscr{K}^2\}$.

**Definition 10.** The *semantics* $[\![K]\!]$ of a flat component $K = gl\{K_1,\ldots,K_n\}$ is defined as $(Q, q^0, \mathscr{I}(gl), \longrightarrow)$, where $Q = \prod_{i=1}^{n} Q_i$, $q^0 = (q_1^0,\ldots,q_n^0)$ and $\longrightarrow$ is such that: given two states $q^1 = (q_1^1,\ldots,q_n^1)$ and $q^2 = (q_1^2,\ldots,q_n^2)$ in $Q$ and an interaction $\alpha \in \mathscr{I}(gl)$, $q^1 \xrightarrow{\alpha} q^2$ if and only if $\forall i, q_i^1 \xrightarrow{\alpha_i}_i q_i^2$, where $\alpha_i = \alpha \cap \mathscr{P}_i$.

We use the convention that $\forall q : q \xrightarrow{\emptyset} q$ so components not involved in an interaction do not move. Thus the semantics of a flat component is obtained as the composition of its constituting LTS where labels are synchronized according to the interactions of $\mathscr{I}(gl)$.

We then define equivalence $\cong$ as follows: two components are equivalent if their flattened forms have the same semantics. Note that in practice one would prefer to define the semantics of a hierarchical component as a function of the semantics of its constituting components. In presence of encapsulation this requires to distinguish between closed and open systems and thus to provide two different semantics. Details can be found in [102].

We now have the ingredients for defining the L1 component framework and we focus on its contract framework.

**Definition 11.** $K_1 \preccurlyeq^{L1} K_2$ if and only if $[\![K_1]\!]$ is simulated by $[\![K_2]\!]$.

Thus L1-conformance is identical to L0-conformance for components without non-observable non-determinism, and otherwise stronger. Note that in verification tools, in order to check trace inclusion efficiently, one will generally check simulation anyway. Satisfaction is defined as follows.

**Definition 12.** A component $K$ *satisfies* a contract $\mathscr{C} = (A, gl, G)$ for $\mathscr{P}_K$, denoted $K \models^{L1} (A, gl, G)$, if and only if:

$$\begin{cases} gl\{K, A_{det}\} \preccurlyeq^{L1} gl\{G, A_{det}\} \\ (q_K, q_A)\mathscr{R}(q_G, q'_A) \wedge \exists q'_K : q_K \xrightarrow{\alpha}_K q'_K \implies \exists q'_G : q_G \xrightarrow{\alpha}_G q'_G \end{cases}$$

where $A_{det}$ is the determinization of $A$, $\mathscr{R}$ is the relation on states proving that $gl\{K, A_{det}\} \preccurlyeq^{L1} gl\{G, A_{det}\}$ and $\alpha \in 2^{\mathscr{P}_K}$ is such that $\exists \alpha' \in \mathscr{I}(gl) : \alpha \subseteq \alpha'$.

Thus $\models^{L1}$ strengthens the satisfaction relation used in the L0 framework by: 1) determinizing $A$; 2) requiring every transition of $K$ to have a counterpart in each related state of $G$ — unless it is structurally forbidden by $gl$ — but the target states of the transition need to be related only if the environment allows this transition. As a consequence, $\models^{L1}$ allows circular reasoning.

### *8.3.3 Relaxed Circular Reasoning*

We have presented in the previous sections two contract frameworks developed in the SPEEDS project. We show now how we use their respective tool chains together. Unifying the L0 and L1 component frameworks is quite straightforward. Nevertheless, we have introduced two different notions of satisfaction: $\models^{L0}$ and $\models^{L1}$ where the second one is strictly stronger than the first one. To combine results based on L0 and L1, we propose a rule called *relaxed circular reasoning* for two (possibly different) refinement relations:

$$K \sqsubseteq^1_{A,gl} G \wedge E \sqsubseteq^2_{G,gl} A \implies K \sqsubseteq^1_{E,gl} G$$

This rule generalizes circular and non-circular reasoning by not restricting $\sqsubseteq^2_{G,gl}$ to refinement under context $\sqsubseteq^1_{G,gl}$ or refinement in any context $\sqsubseteq^1$. Depending on which relation is the most restrictive it can be used in two different ways:

1. If the first relation allows circular reasoning and is stronger than the second one (i.e. $K \sqsubseteq^1_{A,gl} G \implies K \sqsubseteq^2_{A,gl} G$) then our new rule relaxes circular reasoning by requiring $E \sqsubseteq^2_{G,gl} A$ rather than $E \sqsubseteq^1_{G,gl} A$.
2. Symmetrically, if the first relation does not allow circular reasoning and refinement in any context $\sqsubseteq^1$ is stronger than the second one then this rule relaxes non circular reasoning by requiring $E \sqsubseteq^2_{G,gl} A$ rather than $E \sqsubseteq^1 A$.

Interestingly, relaxed circular reasoning can be used both ways for L0- and L1-satisfaction. First it leads to a relaxed sufficient condition for dominance in L1.

**Theorem 2.** *$K \sqsubseteq^{L1}_{A,gl} G \wedge E \sqsubseteq^{L0}_{G,gl} A$ implies $K \sqsubseteq^{L1}_{E,gl} G$.*

**Theorem 3.** *If $\forall i \exists gl_{E_i} : gl \circ gl_I = gl_i \circ gl_{E_i}$ the following is sufficient to prove that $\mathscr{C}$ dominates $\{\mathscr{C}_i\}_{i=1..n}$ w.r.t. $gl$:*

$$\begin{cases} gl_I\{G_1,\ldots,G_n\} \models^{L1} \mathscr{C} \\ \forall i : gl_{E_i}\{A, G_1,\ldots,G_{i-1},G_{i+1},\ldots,G_n\} \models^{L0} \mathscr{C}_i^{-1} \end{cases}$$

In that case, checking that contracts $\{\mathscr{C}_i\}_{i=1}^n$ L1-dominate a contract $\mathscr{C}$ requires one L1-satisfaction check and $n$ L0-satisfaction checks. This is particularly interesting since checking L0-satisfaction may be achieved by using other tools or approaches (that may not need circular reasoning). Moreover, dominance can be established more often as L1-satisfaction is stronger than L0-satisfaction. Second:

**Theorem 4.** *$K \sqsubseteq^{L0}_{A,gl} G \wedge E \sqsubseteq^{L1}_{G,gl} A$ implies $K \sqsubseteq^{L0}_{E,gl} G$.*

This result made it possible in SPEEDS to incorporate results from tools checking L0-satisfaction with results obtained through L1-dominance (implemented by a set of L1-satisfaction checks), thus building a complete tool chain.

## 8.4 Conclusion and Future Work

The work presented in this paper has been motivated by the necessity of combining contract-based verification tools and corresponding results for two component frameworks L0 and L1 defined in the context of the European SPEEDS project. In particular, we were interested in using dominance results established in L1 — and which cannot be obtained using the L0 refinement relation — for further reasoning in L0. To that purpose, we have presented an abstract notion of contract framework for a given component framework that defines three different notions of refinement, that is, conformance, dominance and satisfaction. We show how to derive these notions from refinement of closed systems and refinement under context and we provide a methodology for compositional and hierarchical verification of global properties.

We have studied circular reasoning as a powerful means for proving dominance. As circular reasoning does not always hold for usual notions of refinement, we provide proof rules for dominance relying on a relaxed notion of circular reasoning based on two notions of refinement. We have then shown that our abstract framework is general enough to represent both L0 and L1 as specific instances and proved that the L0 and L1 refinement relations satisfy the condition for relaxed circular reasoning.

This approach was applied to only simple case studies in the SPEEDS project and should therefore rather be seen as a proof of concept. The practical relevance of such an approach is that it opens up ways of connecting tools that work at different levels of abstraction, and relate their results to prove stronger properties. In addition, our results relax the requirements on the tools, since circular reasoning would not be needed at the L0 level.

**Acknowledgements** This work was supported in part by the EU projects COMBEST (n. 215543) and ArtistDesign (n. 214373).

# References

References for all chapters


1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data Structures and Algorithms (1983)
2. Alfaro, L., Henzinger, T.A.: Interface automata. In: Proc. of ESEC/SIGSOFT FSE'01, pp. 109–120. ACM Press (2001)
3. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2), 183–235 (1994). DOI http://dx.doi.org/10.1016/0304-3975(94)90010-8
4. Alur, R., Etessami, K., Torre, S.L., Peled, D.: Parametric Temporal Logic for "Model Measuring". ACM Transactions on Computational Logic **2**(3), 388–407 (2001)
5. Alur, R., Henzinger, T.: Real-time Logics: Complexity and Expressiveness. Information and Computation **104**(1), 35–77 (1993)
6. Alur, R., Henzinger, T.A.: Reactive modules. Formal Methods in System Design **15**(1), 7–48 (1999)
7. Andersson, B.: Global static-priority preemptive multiprocessor scheduling with utilization bound 38%. In: Principles of Distributed Systems. Springer Berlin / Heidelberg (2008)
8. Andrade, H., Correll, J., Ekbal, A., Ghosal, A., Kim, D., Kornerup, J., Limaye, R., Prasad, A., Ravindran, K., Tran, T.N., Trimborn, M., Wang, G., Wong, I., Yang, G.: From Streaming Models to FPGA Implementations. In: Proc. of the Conference for Engineering of Reconfigurable Systems and Algorithms (ERSA-IS). Las Vegas, USA (2012)
9. Andrade, H.A., Kovner, S.: Software Synthesis from Dataflow Models for G and LabVIEW. In: Proc. of the IEEE Asilomar Conference on Signals, Systems, and Computers, pp. 1705–1709 (1998)
10. Anssi, S., Albers, K., Dörfel, M., Gérard, S.: ChronVAL/ChronSIM: A Tool Suite for Timing Analysis of Automotive Applications. In: Proceedings of the Conference on Embedded Real-time Software and Systems (ERTS 2012) (2012)
11. Anssi, S., Tucci-Pergiovanni, S., Mraidha, C., Albinet, A., Terrier, F., Gerard, S.: Completing EAST-ADL2 with MARTE for Enabling Scheduling Analysis for Automotive Applications. In: Conference ERTS (2010)
12. Apvrille, L., Courtiat, J.P., Lohr, C., de Saqui-Sannes, P.: Turtle: a real-time uml profile supported by a formal validation toolkit. Software Engineering, IEEE Transactions on **30**(7), 473–487 (2004). DOI 10.1109/TSE.2004.34. URL http://labsoc.comelec.enst.fr/turtle/ttool.html
13. aquintos GmbH: Preevision. http://www.aquintos.com
14. Arcticus Systems AB: Arcticus Systems home page. Http://www.arcticus-systems.com
15. Audsley, N., Burns, A., Richardson, M.F., Wellings, A.J.: Hard real-time scheduling: The deadline-monotonic approach. In: Workshop on Real-Time Operating Systems and Software (1991)
16. Audsley, N., Dd, Y.: Optimal priority assignment and feasibility of static priority tasks with arbitrary start times (1991)
17. AUTOSAR Consortium: AUTomotive Open System ARchitecture. http://www.autosar.org

18. AUTOSAR Consortium: AUTOSAR Technical Overview, Version 2.2.2. AUTOSAR – AUTomotive Open System ARchitecture, Release 3.1, The AUTOSAR Consortium, Aug., 2008. http://autosar.org
19. BAE Systems: BAE Systems Hägglunds. Http://www.baesystems.com/hagglunds
20. Balarin, F., Chiodo, M., Giusto, P., Hsieh, H., Jurecska, A., Lavagno, L., Passerone, C., Sangiovanni-Vincentelli, A., Sentovich, E., Suzuki, K., Tabbara, B. (eds.): Hardware-Software Co-design of Embedded Systems: The POLIS Approach. Kluwer Academic Publishers, Norwell, MA, USA (1997)
21. Balarin, F., Watanabe, Y., Hsieh, H., Lavagno, L., Passerone, C., Sangiovanni-Vincentelli, A.L.: Metropolis: An integrated electronic system design environment. IEEE Computer **36**(4), 45–52 (2003)
22. Balsamo, S., Di Marco, A., Inverardi, P., Simeoni, M.: Model-based performance prediction in software development: A survey. IEEE Trans. Softw. Eng. **30**(5), 295–310 (2004). DOI 10.1109/TSE.2004.9. URL http://dx.doi.org/10.1109/TSE.2004.9
23. Bartolini, C., Bertolino, A., De Angelis, G., Lipari, G.: A uml profile and a methodology for real-time systems design. In: Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications, EUROMICRO '06, pp. 108–117. IEEE Computer Society, Washington, DC, USA (2006). DOI 10.1109/EUROMICRO.2006.14. URL http://dx.doi.org/10.1109/EUROMICRO.2006.14
24. Bartolini, C., Lipari, G., DiNatale, M.: From functional blocks to the synthesis of the architectural model in embedded real-time applications. In: Proceedings of the 11th IEEE Real Time on Embedded Technology and Applications Symposium, RTAS '05, pp. 458–467. IEEE Computer Society, Washington, DC, USA (2005). DOI 10.1109/RTAS.2005.24. URL http://dx.doi.org/10.1109/RTAS.2005.24
25. Bartolini, C., Lipari, G., Natale, M.D.: From Functional Blocks to the Synthesis of the Architectural Model in Embedded Real-time Applications. In: IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 458–467 (2005)
26. Basu, A., Bozga, M., Sifakis, J.: Modeling heterogeneous real-time components in BIP. In: Proc. of SEFM'06, pp. 3–12. IEEE Computer Society (2006)
27. Behrmann, G., David, A., Larsen, K.G.: A tutorial on UPPAAL. In: M. Bernardo, F. Corradini (eds.) Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004, no. 3185 in LNCS, pp. 200–236. Springer–Verlag (2004)
28. Bengtsson, J., Yi, W.: Timed Automata: Semantics, Algorithms and Tools. In: Lectures on Concurrency and Petri Nets, *LNCS*, vol. 3098, pp. 87–124. Springer Berlin / Heidelberg (2004)
29. Benveniste, A., Caillaud, B., Ferrari, A., Mangeruca, L., Passerone, R., Sofronis, C.: Multiple viewpoint contract-based specification and design. In: F.S. de Boer, M.M. Bonsangue, S. Graf, Willem-Paul de Roever (eds.) Formal Methods for Components and Objects, $6^{th}$ International Symposium (FMCO 2007), Amsterdam, The Netherlands, October 24–26, 2007, Revised Papers, *Lecture Notes in Computer Science*, vol. 5382, pp. 200–225. Springer Verlag Berlin Heidelberg (2008). DOI 10.1007/978-3-540-92188-2
30. Benveniste, A., Caillaud, B., Passerone, R.: A generic model of contracts for embedded systems. Rapport de recherche 6214, Institut National de Recherche en Informatique et en Automatique (2007)
31. Benveniste, A., Caspi, P., Edwards, S.A., Halbwachs, N., Guernic, P.L., Robert, Simone, D.: The synchronous languages 12 years later. In: Proceedings of The IEEE, pp. 64–83 (2003)
32. Benvenuti, L., Ferrari, A., Mangeruca, L., Mazzi, E., Passerone, R., Sofronis, C.: A contract-based formalism for the specification of heterogeneous systems. In: Proceedings of the Forum on Specification, Verification and Design Languages (FDL08), pp. 142–147. Stuttgart, Germany (2008). DOI 10.1109/FDL.2008.4641436
33. Berg, H., Brunelli, C., Lücking, U.: Analyzing Models of Computation for Software Defined Radio Applications. In: International Symposium on System-on-Chip (SOC), pp. 1–4. Tampere, Finland (2008)

34. Berry, G., Gonthier, G.: The esterel synchronous programming language: design, semantics, implementation. Sci. Comput. Program. **19**(2), 87–152 (1992)
35. Berthomieu, B., Diaz, M.: Modeling and verification of time dependent systems using time petri nets. Software Engineering, IEEE Transactions on **17**(3), 259–273 (1991)
36. Bertogna, M., Cirinei, M., Lipari, G.: New schedulability tests for real-time task sets scheduled by deadline monotonic on multiprocessors. In: Principles of Distributed Systems. Springer Berlin / Heidelberg (2006)
37. Bhattacharya, B., Bhattacharyya, S.: Parameterized Dataflow Modeling for DSP Systems. IEEE Trans. on Signal Processing **49**(10), 2408 –2421 (2001)
38. Bhattacharyya, S.S., Deprettere, E., Leupers, R., Takala, J. (eds.): Handbook of Signal Processing Systems. Springer (2010)
39. Bhattacharyya, S.S., Murthy, P.K., Lee, E.A.: Software Synthesis from Dataflow Graphs. Kluwer Academic Press, Norwell, MA (1996)
40. Bhattacharyya, S.S., Murthy, P.K., Lee, E.A.: APGAN and RPMC: Complementary heuristics for translating DSP block diagrams into efficient software implementations. Journal of Design Automation for Embedded Systems **2**(1), 33–60 (1997)
41. Bilsen, G., Engels, M., Lauwereins, R., Peperstraete, J.: Cyclo-static data flow. In: IEEE Intl. Conf. Acoustics, Speech, and Signal Processing, vol. 5, pp. 3255–3258 (1995)
42. Bilsen, G., Engels, M., Lauwereins, R., Peperstraete, J.A.: Cyclo-static dataflow. IEEE Transactions on Signal Processing **44**(2), 397–408 (1996)
43. Blickle, T., Teich, J., Thiele, L.: System-Level Synthesis Using Evolutionary Algorithms. Design Automation for Embedded Systems **3**(1), 23–62 (1998)
44. Blossom, E.: GNU radio: tools for exploring the radio frequency spectrum. Linux Journal (2004)
45. Boussinot, F., De Simone, R.: The esterel language. Proceedings of the IEEE **79**(9), 1293–1304 (1991)
46. Bozga, M., et. al.: The IF Toolset. In: Formal Methods for the Design of Real-Time Systems, Lecture Notes in Computer Science, pp. 237-267, volume 3185, 2004, Springer
47. Broy, M., Feilkas, M., Herrmannsdoerfer, M., Merenda, S., Ratiu, D.: Seamless Model-Based Development- From Isolated Tools to Integrated Model Engineering Environments. Proceedings of the IEEE **98**(4), 526 –545 (2010)
48. Cameron, J.: An Overview of JSD. IEEE Transactions on Software Engineering **12**(2), 938–949 (1986)
49. Caspi, P., Pilaud, D., Halbwachs, N., Plaice, J.A.: Lustre: a declarative language for real-time programming. In: Proceedings of the 14th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, POPL '87, pp. 178–188. ACM (1987)
50. Cervin, A., Henriksson, D., Lincoln, B., Eker, J., Arzen, K.: How does control timing affect performance? analysis and simulation of timing using jitterbug and truetime. Control Systems, IEEE **23**(3), 16–30 (June)
51. CESAR Consortium: Home page. http://www.cesarproject.eu/
52. Chakraborty, S., Kunzli, S., Thiele, L.: A General Framework for Analysing System Properties in Platform-based Embedded System Designs. In: Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2003), pp. 190–195 (2003)
53. Chapin, J., Lum, V., Muir, S.: Experiences implementing gsm in rdl (the vanu radio description language trade;). In: Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE, vol. 1, pp. 213–217 vol.1 (2001). DOI 10.1109/MILCOM.2001.985792
54. Chapiro, D.M.: Globally-Asynchronous Locally-Synchronous Systems. Ph.D. thesis, Stanford Univ., CA. (1984)
55. Che, W., Panda, A., Chatha, K.: Compilation of stream programs for multicore processors that incorporate scratchpad memories. In: Design, Automation Test in Europe Conference Exhibition (DATE), 2010, pp. 1118–1123 (2010)
56. Chen, R., Sgroi, M., Lavagno, L., Martin, G., Sangiovanni-Vincentelli, A., Rabaey, J.: Uml and platform-based design pp. 107–126 (2003)

57. Cheriyan, J., Mehlhorn, K.: Algorithms for Dense Graphs and Networks on the Random Access Computer. Algorithmica **15**(6), 521–549 (1996)
58. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. The MIT Press, Cambridge, MA, USA (2000)
59. Cobleigh, J.M., Avrunin, G.S., Clarke, L.A.: Breaking up is hard to do: An evaluation of automated assume-guarantee reasoning. ACM Trans. Softw. Eng. Methodol. **17**(2) (2008)
60. COMBEST Consortium: Home page. http://www.combest.eu
61. Crnkovic, I., Larsson, M.: Building Reliable Component-Based Software Systems. Artech House, Inc., Norwood, MA, USA (2002)
62. Cucinotta, T., Palopoli, L.: QoS Control for Pipelines of Tasks using Multiple Resources. IEEE Trans. on Computers **59**, 416–430 (2010)
63. Cuenot, P., Frey, P., Johansson, R., Lönn, H., Reiser, M.O., Servat, D., Tavakoli Kolagari, R., Chen, D.: Developing Automotive Products Using the EAST-ADL2, an AUTOSAR Compliant Architecture Description Language. In: Proceedings of the 4th European Congress ERTS (Embedded Real Time Software) (2008)
64. Damm, W.: Controlling speculative design processes using rich component models. In: Proc. of ACSD'05, pp. 118–119. IEEE Computer Society (2005)
65. Damm, W., Votintseva, A., Metzner, A., Josko, B., Peikenkamp, T., Böde, E.: Boosting reuse of embedded automotive applications through rich components. In: Proc. of Foundations of Interface Technologies (FIT) (2005)
66. Davare, A., Densmore, D., Meyerowitz, T., Pinto, A., Sangiovanni-Vincentelli, A., Yang, G., Zeng, H., Zhu, Q.: A next-generation design framework for platform-based design. In: DVCon 2007 (2007)
67. Davare, A., Zhu, Q., Natale, M.D., Pinello, C., Kanajan, S., Sangiovanni-vincentelli, A.L.: Period Optimization for Hard Real-time Distributed Automotive Systems. In: Design Automation Conference, pp. 278–283 (2007). DOI 10.1109/DAC.2007.375172
68. Davis, R., Burns, A.: Optimal priority assignment for aperiodic tasks with firm deadlines in fixed priority pre-emptive systems. Information Processing Letters **53**, 249 – 254 (1995)
69. Davis, R.I., Burns, A.: Priority assignment for global fixed priority pre-emptive scheduling in multiprocessor real-time systems. In: Real-Time Systems Symp. (RTSS) (2009)
70. Davis, R.I., Burns, A., Bril, R.J., Lukkien, J.J.: Controller area network (can) schedulability analysis: Refuted, revisited and revised. Real-Time Syst. **35**, 239–272 (2007)
71. Delahaye, B., Caillaud, B., Legay, A.: Probabilistic contracts: A compositional reasoning methodology for the design of stochastic systems. In: Proc. of ACSD'10, pp. 223–232 (2010)
72. Dennis, J.: Data flow supercomputers. Computer **13**(11), 48–56 (1980). DOI 10.1109/MC.1980.1653418
73. Dixit, M., Ramesh, S., Dasgupta, P.: Some Results on Parametric Temporal Logic. Information Processng Letters **111**(3), 994–998 (2011)
74. Dixit, M.G.: Formal methods for early time-budgeting in component based embedded control systems. Ph.D. thesis, IIT Kharagpur (2012)
75. Dixit, M.G., Dasgupta, P., Ramesh, S.: Taming the Component Timing: A CBD Methodology for Component based Embedded Systems. In: Design Automation and Test in Europe (DATE) (2009)
76. Douglass, B.: Real Time UML. Pearson Education (2009)
77. Douglass, B.P.: Real Time UML: Advances in the UML for Real-Time Systems (3rd Edition). Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA (2004)
78. EAST-ADL Overview. http://www.atesst.org/home/liblocal/docs/ConceptPresentations/01_EAST-ADL_OverviewandStructure.pdf
79. Eclipse Modeling Framework. http://www.eclipse.org
80. Eclipse Foundation: Papyrus. http://www.eclipse.org/modeling/mdt/papyrus/
81. Edwards, M., Green, P.: The Implementation of Synchronous Dataflow Graphs Using Reconfigurable Hardware. In: Proc. of FPL '00, pp. 739–748 (2000)
82. Eker, J., Janneck, J.W., Lee, E.A., Liu, J., Liu, X., Ludvig, J., Neuendorffer, S., Sachs, S., Xiong, Y.: Taming Heterogeneity - The Ptolemy Approach. Proc. of the IEEE **91**(1), 127–144 (2003)

83. Emerson, A.E., Mok, A., Sistla, A., Srinivasan, J.: Quantitative Temporal Reasoning. In: Computer Aided Verification, pp. 136–145 (1994)
84. Emerson, A.E., Trefler, R.: Parametric Quantitative Temporal Reasoning. In: IEEE Symposium on Logic in Computer Science, pp. 336–343 (1999)
85. Enoiu, E.P., Marinescu, R., Seceleanu, C., Pettersson, P.: ViTAL : A Verification Tool for EAST-ADL Models using UPPAAL PORT. In: Proceedings of the 17th IEEE International Conference on Engineering of Complex Computer Systems. IEEE Computer Society Press (2012)
86. Esterel Technologies: SCADE suite. http://www.esterel-technologies.com/products/scade-suite/
87. Evidence: Home page. http://evidence.eu.com
88. Feiertag, N., Richter, K., Nordlander, J., Jonsson, J.: A Compositional Framework for End-to-End Path Delay Calculation of Automotive Systems under Different Path Semantics. In: Workshop on Compositional Theory and Technology for Real-Time Embedded Systems (CRTS) (2008)
89. Feng, L., Chen, D.J., Lönn, H., Törngren, M.: Verifying System Behaviors in EAST-ADL2 with the SPIN Model Checker. In: Mechatronics and Automation (ICMA), 2010 International Conference on, pp. 144 –149 (2010)
90. FMCSA: Forward Collision Warning Systems (CWS) (2005). http://www.fmcsa.dot.gov/facts-research/research-technology/report/forward-collision-warning-systems.htm
91. France, R., Ghosh, S., Dinh-Trong, T., Solberg, A.: Model-driven development using uml 2.0: promises and pitfalls. Computer **39**(2), 59–66 (2006). DOI 10.1109/MC.2006.65
92. García, J.G., Harbour, M.G.: Optimized priority assignment for tasks and messages in distributed hard real-time systems. In: Workshop on Parallel and Distributed Real-Time Systems (1995)
93. Garner, G., Gelter, A., Teener, M.: New simulation and test results for IEEE 802.1as timing performance. In: International Symposium on Precision Clock Synchronization for Measurement, Control and Communication, pp. 1–7 (2009)
94. George, L., Rivierre, N., Spuri, M.: Preemptive and Non-Preemptive Real-Time UniProcessor Scheduling. Research Report RR-2966, INRIA (1996)
95. Ghattas, R., Dean, A.G.: Preemption threshold scheduling: Stack optimality, enhancements and analysis. In: Proceedings of the 13th IEEE Real Time and Embedded Technology and Applications Symposium, RTAS '07, pp. 147–157. IEEE Computer Society, Washington, DC, USA (2007)
96. Ghosal, A., Limaye, R., Ravindran, K., Tripakis, S., Prasad, A., Wang, G., Tran, T.N., Andrade, H.: Static Dataflow with Access Patterns: Semantics and Analysis. In: Proc. of the 49th Annual Design Automation Conference, DAC '12, pp. 656–663. ACM, New York, NY, USA (2012)
97. Girault, A., Lee, B., Lee, E.A.: Hierarchical Finite State Machines with Multiple Concurrency Models. IEEE Transactions on Computer-Aided Design **18**(6), 742–760 (1999)
98. Glaß, M., Herrscher, D., Meier, H., Piastowski, M., Schoo, P.: SEIS – Security in Embedded IP-based Systems. ATZelektronik worldwide **1**, 36–40 (2010)
99. Glaß, M., Lukasiewycz, M., Teich, J., Bordoloi, U., Chakraborty, S.: Designing heterogeneous ECU networks via compact architecture encoding and hybrid timing analysis. In: Proceedings of Design Automation Conference (DAC), pp. 43–46 (2009)
100. Gomaa, H.: Designing Concurrent, Distributed, and Real-Time Applications with Uml, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2000)
101. Gomma, H.: A Software Design Method for Real-time Systems. Communications ACM **27**(9), 938–949 (1984)
102. Graf, S., Passerone, R., Quinton, S.: Contract-based reasoning for component systems with complex interactions. Research report TR-2010-12, VERIMAG (2010 updated 2013)
103. Graf, S., Russ, T., Glaß, M., Teich, J.: Considering MOST150 during virtual prototyping of automotive E/E architectures. In: Proceedings of the Automotive meets Electronics (AmE), pp. 116–121 (2012)

104. Graf, S., Streubühr, M., Glaß, M., Teich, J.: Analyzing automotive networks using virtual prototypes. In: Proceedings of the Automotive meets Electronics (AmE), pp. 10–15 (2011)
105. Green, P., Essa, S.: Integrating the synchronous dataflow model with uml. In: Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings, vol. 1, pp. 736–737 Vol.1 (2004). DOI 10.1109/DATE.2004.1268954
106. Gssler, G.: Prometheus - A Compositional Modeling Tool for Real-Time Systems. In: Workshop on Real-Time Tools (RT-TOOLS), 2001 (2001)
107. Gu, R., Janneck, J.W., Raulet, M., Bhattacharyya, S.S.: Exploiting Statically Schedulable Regions in Dataflow Programs. In: Proc. of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '09, pp. 565–568. IEEE Computer Society, Washington, DC, USA (2009)
108. Gu, Z., He, Z.: Real-time scheduling techniques for implementation synthesis from component-based software models. In: Proceedings of the 8th international conference on Component-Based Software Engineering, CBSE'05, pp. 235–250. Springer-Verlag, Berlin, Heidelberg (2005)
109. Hafaiedh, I.B., Graf, S., Quinton, S.: Reasoning about safety and progress using contracts. In: Proc. of ICFEM'10, pp. 436–451 (2010)
110. Halbwachs, N., Caspi, P., Raymond, P., Pilaud, D.: The synchronous dataflow programming language lustre. In: Proceedings of the IEEE, pp. 1305–1320 (1991)
111. Hamann, A., Jersak, M., Richter, K., Ernst, R.: Design Space Exploration and System Optimization with SymTA/S - Symbolic Timing Analysis for Systems. In: IEEE Real-Time Systems Symposium, pp. 469 – 478 (2004)
112. Hamann, A., Jersak, M., Richter, K., Ernst, R.: A framework for modular analysis and exploration of heterogeneous embedded systems. Real-Time Syst. **33**, 101–137 (2006)
113. Hamann, A., Racu, R., Ernst, R.: Multi-dimensional robustness optimization in heterogeneous distributed embedded systems. In: Proceedings of the 13th IEEE Real Time and Embedded Technology and Applications Symposium, RTAS '07, pp. 269–280. IEEE Computer Society, Washington, DC, USA (2007)
114. Hänninen, K.: Efficient memory utilization in resource constrained real-time systems. Ph.D. thesis, Mälardalen University, Sweden (2008)
115. Hänninen et.al., K.: The Rubus Component Model for Resource Constrained Real-Time Systems. In: 3rd IEEE International Symposium on Industrial Embedded Systems (2008)
116. Harel, D.: Statecharts: A visual formalism for complex systems. Sci. Comput. Program. **8**(3), 231–274 (1987)
117. Hatley, D., Pirabhai, I.: Strategies for Real-time System Specificaiton. Dorset House (1988)
118. Heinecke, H., et al.: AUTOSAR – Current results and preparations for exploitation. In: Proceedings of the 7th Euroforum Conference, EUROFORUM '06 (2006)
119. Hekkala, A., Harjula, I., Panaitopol, D., Rautio, T., Pacalet, R.: Cooperative spectrum sensing study using welch periodogram. In: Telecommunications (ConTEL), Proceedings of the 2011 11th International Conference on, pp. 67–74 (2011)
120. Hendriks, M., Verhoef, M.: Timed Automata Based Analysis of Embedded System Architectures. In: Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International, p. 8 pp. (2006)
121. Henia, R., Hamann, A., Jersak, M., Racu, R., Richter, K., Ernst, R.: System level performance analysis - the SymTA/S approach. Computers and Digital Techniques, IEE Proc. - **152**, 148–166 (2005). DOI 10.1049/ip-cdt:20045088
122. Henzinger, T.A., Sifakis, J.: The Embedded Systems Design Challenge. In: Proceedings of the 14th International Symposium on Formal Methods (FM), Lecture Notes in Computer Science, pp. 1–15. Springer (2006)
123. Hong, S., Chantem, T., Hu, X.S.: Meeting end-to-end deadlines through distributed local deadline assignments. In: Real-Time Systems Symp. (RTSS) (2011)
124. Horstmannshoff, J., Meyr, H.: Optimized System Synthesis of Complex RT Level Building Blocks from Multirate Dataflow Graphs. In: Proc. of ISSS '99, pp. 38–43 (1999)

125. Hsu, C.J., Pino, J.L., Hu, F.J.: A mixed-mode vector-based dataflow approach for modeling and simulating lte physical layer. In: Proc. of the 47th Design Automation Conference, DAC '10, pp. 18–23. ACM, New York, NY, USA (2010)
126. Imtiaz, J., Jasperneite, J., Han, L.: A performance study of Ethernet Audio Video Bridging (AVB) for industrial real-time communication. In: IEEE Conference on Emerging & Technologies Factory Automation (ETFA), pp. 1–8 (2009)
127. The Institute of Electrical and Electronics Engineers, Inc.: Local and metropolitan area networks – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications, IEEE Std 802.3, 2000 edn. (2000)
128. The Institute of Electrical and Electronics Engineers, Inc.: IEEE Standard for Local and metropolitan area networks – Virtual Bridged Local Area Networks, IEEE Std 802.1qav-2009 edn. (2009)
129. The Institute of Electrical and Electronics Engineers, Inc.: IEEE Standard for Local and metropolitan area networks–Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks (2011)
130. Jair Gonzalez-Pina, R.A.B.R.P.: Diplodocusdf, a domain-specific modelling language for software defined radio applications. In: Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on, vol. 1, pp. 213–217 vol.1 (2012)
131. Janneck, J., Miller, I., Parlour, D., Roquier, G., Wipliez, M., Raulet, M.: Synthesizing Hardware from Dataflow Programs: An MPEG-4 Simple Profile Decoder Case Study. In: IEEE Workshop on Signal Processing Systems, pp. 287–292 (2008)
132. Janneck, J.W.: Open Dataflow (OpenDF). URL http://www.opendf.org/
133. Jin, D., Levy, D.C.: An approach to schedulability analysis of uml-based real-time systems design. In: Proceedings of the 3rd international workshop on Software and performance, WOSP '02, pp. 243–250. ACM, New York, NY, USA (2002). DOI 10.1145/584369.584409. URL http://doi.acm.org/10.1145/584369.584409
134. Jonsson, B., Perathoner, S., Thiele, L., Yi, W.: Cyclic Dependencies in Modular Performance Analysis. In: Proceedings of the 8th ACM International Conference on Embedded software (EMSOFT 2008), pp. 179–188 (2008)
135. Jonsson, J., Shin, K.G.: Robust adaptive metrics for deadline assignment in distributed hard real-time systems. Real-Time Systems **23**, 239–271 (2002)
136. Jung, H., Lee, K., Ha, S.: Optimized RTL Code Generation from Coarse-Grain Dataflow Specification for Fast HW/SW Cosynthesis. J. Signal Process. Syst. **52**(1), 13–34 (2008)
137. Kahn, G.: The semantics of a simple language for parallel programming. In: IFIP Cong. (1974)
138. Kang, E.Y., Schobbens, P.Y., Pettersson, P.: Verifying Functional Behaviors of Automotive Products in EAST-ADL2 Using UPPAAL-PORT. In: Computer Safety, Reliability, and Security, *LNCS*, vol. 6894, pp. 243–256. Springer Berlin / Heidelberg (2011)
139. Ke, X., Sierszecki, K., Angelov, C.: COMDES-II: A Component-Based Framework for Generative Development of Distributed Real-Time Control Systems. In: 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), 2007, pp. 199 –208 (2007). DOI 10.1109/RTCSA.2007.29
140. Kee, H., Shen, C.C., Bhattacharyya, S., Wong, I., Rao, Y., Kornerup, J.: Mapping Parameterized Cyclo-static Dataflow Graphs onto Configurable Hardware. Journal of Signal Processing Systems pp. 1–17 (2011)
141. Kenneth, J.: Schedulability analysis of embedded applications modelled using marte (2009)
142. Kern, A., Schmutzler, C., Streichert, T., Hübner, M., Teich, J.: Network bandwidth optimization of Ethernet-based streaming applications in automotive embedded systems. In: Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN), pp. 1–6 (2010)
143. Keutzer, K., Newton, A., Rabaey, J., Sangiovanni-Vincentelli, A.: System-level design: orthogonalization of concerns and platform-based design. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **19**(12), 1523–1543 (2000)

144. Kienhuis, B., Deprettere, E.F., Wolf, P.v.d., Vissers, K.A.: A methodology to design programmable embedded systems - the y-chart approach. In: Embedded Processor Design Challenges: Systems, Architectures, Modeling, and Simulation - SAMOS, pp. 18–37. Springer-Verlag, London, UK, UK (2002)
145. Klobedanz, K., Kuznik, C., Thuy, A., Mueller, W.: Timing Modeling and Analysis for AUTOSAR-Based Software Development - A Case Study. In: Design Automation and Test in Europe, pp. 642 – 645 (2010)
146. Knorr-Bremse AG: Home page. http://www.knorr-bremse.com
147. Ko, M., Shen, C., Bhattacharyya, S.S.: Memory-constrained block processing for DSP software optimization. Journal of Signal Processing Systems **50**(2), 163–177 (2008)
148. Ko, M., Zissulescu, C., Puthenpurayil, S., Bhattacharyya, S.S., Kienhuis, B., Deprettere, E.: Parameterized looped schedules for compact representation of execution sequences in DSP hardware and software implementation. IEEE Transactions on Signal Processing **55**(6), 3126–3138 (2007)
149. Künzli, S., Hamann, A., Ernst, R., Thiele, L.: Combined Approach to System Level Performance Analysis of Embedded Systems. In: Proceedings of the 5th IEEE/ACM International Conference on Hardware/software Codesign and System Synthesis (CODES+ISSS 2007), pp. 63–68 (2007)
150. Lalgudi, K.N., Papaefthymiou, M.C., Potkonjak, M.: Optimizing computations for effective block-processing. ACM Transactions on Design Automation of Electronic Systems **5**(3), 604–630 (2000)
151. Lampka, K., Perathoner, S., Thiele, L.: Analytic Real-time Analysis and Timed Automata: A Hybrid Method for Analyzing Embedded Real-time Systems. In: Proceedings of the 9th ACM International Conference on Embedded software (EMSOFT 2009), pp. 107–116 (2009)
152. Lapsley, P., Bier, J., Shoham, A., Lee, E.A.: DSP Processor Fundamentals. Berkeley Design Technology, Inc. (1994)
153. Larsen, K.G., Nyman, U., Wasowski, A.: Interface input/output automata. In: Proc. of FM'06, *LNCS*, vol. 4085, pp. 82–97 (2006)
154. Larsen, K.G., Nyman, U., Wasowski, A.: Modal I/O automata for interface and product line theories. In: Proc. of ESOP'07, *LNCS*, vol. 4421, pp. 64–79 (2007)
155. Lauwereins, R., Engels, M., Adé, M., Peperstraete, J.A.: Grape-II: A System-Level Prototyping Environment for DSP Applications. Computer **28**(2), 35–43 (1995)
156. Ledeczi, A., Maroti, M., Bakay, A., Karsai, G., Garrett, J., Thomason, C., Nordstrom, G., Sprinkle, J., Volgyesi, P.: The generic modeling environment. In: Workshop on Intelligent Signal Processing (2001)
157. Lee, E., Sangiovanni-Vincentelli, A.: A framework for comparing models of computation. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **17**(12), 1217–1229 (1998)
158. Lee, E.A., Messerschmitt, D.G.: Synchronous Data Flow. Proc. of the IEEE **75**(9), 1235–1245 (1987)
159. Lee, E.A., Parks, T.M.: Dataflow process networks. Proceedings of the IEEE pp. 773–799 (1995)
160. Lehoczky, J., Ramos-Thuel, S.: An optimal algorithm for scheduling soft-aperiodic tasks in fixed-priority preemptive systems. In: Real-Time Systems Symp. (RTSS), pp. 110 –123 (1992)
161. Lehoczky, J.P., Sha, L., Ding, Y.: The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In: RTSS, pp. 166–171 (1989)
162. Leung, J.Y.T., Whitehead, J.: On the complexity of fixed-priority scheduling of periodic, real-time tasks. Performance Evaluation **2**, 237–250 (1982)
163. Lim, H., Volker, L., Herrscher, D.: Challenges in a future IP/Ethernet-based in-car network for real-time applications. In: Design Automation Conference (DAC), pp. 7–12 (2011)
164. Lim, H.T., Herrscher, D., Waltl, M.J., Chaari, F.: Performance analysis of the IEEE 802.1 Ethernet Audio/Video Bridging Standard. In: Proceedings of the International Conference on Simulation Tools and Techniques (ICST) (2012)

165. Lim, H.T., Krebs, B., Völker, L., Zahrer, P.: Performance evaluation of the inter-domain communication in a switched Ethernet based in-car network. In: Proceedings of the Conference on Local Computer Networks (LCN) (2011)
166. Lim, H.T., Weckemann, K., Herrscher, D.: Performance study of an in-car switched Ethernet network without prioritization. In: Communication Technologies for Vehicles, pp. 165–175. Springer (2011)
167. Liu, C.L., Layland, J.W.: Scheduling algorithms for multiprogramming in a hard-real-time environment. J. ACM **20**, 46–61 (1973)
168. Lönn, H., Freund, U.: Automotive Embedded Systems Handbook, chap. Automotive Architecture Description Languages. CRC Press (2009)
169. Lukasiewycz, M., Glaß, M., Haubelt, C., Teich, J.: SAT-Decoding in Evolutionary Algorithms for Discrete Constrained Optimization Problems. In: Proc. of CEC '07, pp. 935–942 (2007)
170. Lukasiewycz, M., Glaß, M., Haubelt, C., Teich, J., Regler, R., Lang, B.: Concurrent topology and routing optimization in automotive network integration. In: Proceedings of Design Automation Conference (DAC), pp. 626–629 (2008)
171. Lukasiewycz, M., Glaß, M., Reimann, F., Teich, J.: Opt4J – a modular framework for metaheuristic optimization. In: Proceedings of the Genetic and Evolutionary Computing Conference (GECCO). Dublin, Ireland (2011)
172. Lukasiewycz, M., Streubühr, M., Glaß, M., Haubelt, C., Teich, J.: Combined System Synthesis and Communication Architecture Exploration for MPSoCs. In: Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2009), pp. 472–477 (2009)
173. Maier, P.: A lattice-theoretic framework for circular assume-guarantee reasoning. Ph.D. thesis, Universität des Saarlandes (2003)
174. Mäki-Turja, J., Nolin, M.: Efficient implementation of tight response-times for tasks with offsets. Real-Time Syst. **40**(1), 77–116 (2008). DOI http://dx.doi.org/10.1007/s11241-008-9050-9
175. Manderscheid, M., Langer, F.: Network calculus for the validation of automotive Ethernet in-vehicle network configurations. In: 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 206–211 (2011)
176. Mecel AB: Home page. http://www.mecel.se
177. Modelica Association: Modelica and the modelica association. https://www.modelica.org/
178. Mok, A.K.: Fundamental design problems of distributed systems for the hard-real-time environment. Ph.d. thesis, Cambridge, MA, USA (1983)
179. Montag, P., Nowotka, D., Levi, P.: Verification in the Design Process of Large Real-Time Systems: A Case Study. In: Automotive Safety and Security 2006, Stuttgart (Germany), October 12-13, 2006, pp. 1–13 (2006)
180. Moreira, O.M., Bekooij, M.J.G.: Self-Timed Scheduling Analysis for Real-Time Applications. EURASIP Journal on Advances in Signal Processing **2007**(83710), 1–15 (2007)
181. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Support for end-to-end response-time and delay analysis in the industrial tool suite: Issues, experiences and a case study. In: Computer Science and Information Systems, vol. 10, no. 1, pp 453-482, January 2013. ISSN: 1361-1384
182. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Exploring Options for Modeling of Real-Time Network Communication in an Industrial Component Model for Distributed Embedded Systems. In: The 6th International Conference on Embedded and Multimedia Computing (EMC-2011), *Lecture Notes in Electrical Engineering*, vol. 102, pp. 441–458. Springer Berlin / Heidelberg (2011)
183. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending response-time analysis of controller area network (CAN) with FIFO queues for mixed messages. In: 16th IEEE Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1–4 (2011). DOI 10.1109/ETFA.2011.6059188
184. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending schedulability analysis of controller area network (CAN) for mixed (periodic/sporadic) messages. In: 16th IEEE Conference on

Emerging Technologies and Factory Automation (ETFA) (2011). DOI 10.1109/ETFA.2011.6059010
185. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Tracing event chains for holistic response-time analysis of component-based distributed real-time systems. In: 23rd Euromicro Conference on Real-Time Systems (ECRTS 2011), WIP Session. ACM SIGBED Review (2011)
186. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Response-Time Analysis of Mixed Messages in Controller Area Network with Priority- and FIFO-Queued Nodes. In: 9th IEEE International Workshop on Factory Communication Systems (WFCS) (2012)
187. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Worst-case response-time analysis for mixed messages with offsets in controller area network. In: 17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA) (2012)
188. Mubeen, S., Mäki-Turja, J., Sjödin, M., Carlson, J.: Analyzable modeling of legacy communication in component-based distributed embedded systems. In: 37th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 229–238 (2011). DOI 10.1109/SEAA.2011.43
189. Muskens, J., Chaudron, M.R.V., Lukkien, J.J.: A component framework for consumer electronics middleware. In: Component-Based Software Development for Embedded Systems. pp 164-184, 2005 (2005)
190. Natale, M.D., Stankovic, J.A.: Dynamic end-to-end guarantees in distributed real time systems. In: Real-Time Systems Symp. (RTSS) (1994)
191. Natale, M.D., Zheng, W., Pinello, C., Giusto, P., Sangiovanni-Vincentelli, A.L.: Optimizing End-to-end Latencies by Adaptation of the Activation Events in Distributed Automotive Systems. In: IEEE Real-Time and Embedded Technology and Applications Symposium (2007)
192. National Instruments Corp.: LabVIEW FPGA. URL http://www.ni.com/fpga
193. Neukirchner, M.: System models for free. http://smff.sourceforge.net (2011). URL http://smff.sourceforge.net
194. Neukirchner, M., Stein, S., Ernst, R.: A lazy algorithm for distributed priority assignment in real-time systems. In: Workshop on Self-Organizing Real-Time Systems (SORT) (2011)
195. Neukirchner, M., Stein, S., Ernst, R.: SMFF: System Models for Free. In: 2nd Int'l. Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS) (2011)
196. Neukirchner, M., Stein, S., Schrom, H., Ernst, R.: A software Update Service with Self-Protection Capabilities. In: Conf. on Design, Automation and Test in Europe (DATE) (2010)
197. NuSMV: A New Symbolic Model Checker. http://nusmv.fbk.eu//
198. Nussbaum, D., Kalfallah, K., Knopp, R., Moy, C., Nafkha, A., Leray, P., Delorme, M., Palicot, J., Martin, J., Clermidy, F., Mercier, B., Pacalet, R.: ropen platform for prototyping of advanced software defined radio and cognitive radio techniques. In: Digital System Design, Architectures, Methods and Tools, 2009. DSD '09. 12th Euromicro Conference on, pp. 435–440 (2009). DOI 10.1109/DSD.2009.123
199. NVIDIA: NVIDIA CUDA Compute Unified Device Architecture: Programming Guide, Version 1.0 (2007)
200. Object Management Group: Home page. http://www.omg.org/
201. Object Management Group: Omg model driven architecture. http://www.omg.org/mda/
202. Object Management Group: UML profile for schedulability, performance, and time (spt), formal/2005-01-02 (2005). http://www.omg.org/spec/SPTP/
203. Object Management Group: UML profile for modeling and analysis of real-time and embedded systems (MARTE), version 1.1, formal/2011-06-02 (June 2011). http://www.omg.org/spec/MARTE/1.1/
204. Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E., Phillips, J.C.: GPU computing. Proceedings of the IEEE **96**(5), 879–899 (2008)
205. Palencia, J., Harbour, M.G.: Schedulability Analysis for Tasks with Static and Dynamic Offsets. Real-Time Systems Symposium, IEEE International p. 26 (1998). DOI http://doi.ieeecomputersociety.org/10.1109/REAL.1998.739728

206. Palencia, J.C., González Harbour, M.: Schedulability analysis for tasks with static and dynamic offsets. In: Proceedings of the IEEE Real-Time Systems Symposium, RTSS '98, pp. 26–. IEEE Computer Society, Washington, DC, USA (1998). URL http://dl.acm.org/citation.cfm?id=827270.829048
207. Palopoli, L., Abeni, L., Cucinotta, T., Lipari, G., Baruah, S.: Weighted feedback reclaiming for multimedia applications. In: Workshop on Embedded Systems for Real-Time Multimedia (ESTImedia) (2008). DOI 10.1109/ESTMED.2008.4697009
208. Papadopoulos, G.: Automatic code generation: A practical approach. In: Information Technology Interfaces, 2008. ITI 2008. 30th International Conference on, pp. 861–866 (2008). DOI 10.1109/ITI.2008.4588524
209. Parnas, D., Clements, P., Weiss, D.: The Modular Structure of Complex Systems. In: IEEE Conference on Software Engineering, pp. 551–556 (1984)
210. Partners, S.: SPEEDS metamodel. SPEEDS project deliverable D2.1.5 (2009)
211. Pasaje, J.L.M., et al.: UML-MAST. http://mast.unican.es/umlmast/
212. Phan, T.H., Gerard, S., Terrier, F.: Languages for system specification. chap. Real-time system modeling with ACCORD/UML methodology: illustration through an automotive case study, pp. 51–70. Kluwer Academic Publishers, Norwell, MA, USA (2004). URL http://dl.acm.org/citation.cfm?id=1016425.1016431
213. Pinto, A., Bonivento, A., Sangiovanni-Vincentelli, A.L., Passerone, R., Sgroi, M.: System level design paradigms: Platform-based design and communication synthesis. ACM Transactions on Design Automation of Electronic Systems **11**(3), 537–563 (2006). DOI http://doi.acm.org/10.1145/1142980.1142982
214. Plishker, W., Sane, N., Bhattacharyya, S.S.: A generalized scheduling approach for dynamic dataflow applications. In: Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, pp. 111–116. Nice, France (2009)
215. Plishker, W., Sane, N., Bhattacharyya, S.S.: Mode grouping for more effective generalized scheduling of dynamic dataflow applications. In: Proceedings of the Design Automation Conference, pp. 923–926. San Francisco (2009)
216. Plishker, W., Sane, N., Kiemb, M., Anand, K., Bhattacharyya, S.S.: Functional DIF for rapid prototyping. In: Proceedings of the International Symposium on Rapid System Prototyping, pp. 17–23. Monterey, California (2008)
217. Plishker, W., Sane, N., Kiemb, M., Bhattacharyya, S.S.: Heterogeneous design in functional DIF. In: P. Stenström (ed.) Transactions on High-Performance Embedded Architectures and Compilers IV, *Lecture Notes in Computer Science*, vol. 6760, pp. 391–408. Springer Berlin / Heidelberg (2011)
218. Plunkett, D.: Safety Critical Software Development for a Brake By-wire system. In: SAE Technical Paper Series, 2006-01-1672 (2006)
219. Pnueli, A.: The Temporal Logic of Programs. In: 18th IEEE. Foundations of Computer Science, pp. 46–57 (1977)
220. Ponsard, C., Massonet, P., Molderez, J.F., Rifaut, A., Lamsweerde, A.V., Van, H.T.: Early Verification and Validation of Mission Critical Systems. Form. Methods Syst. Des. **30**(3), 233–247 (2007)
221. Pont, M.J.: Applying time-triggered architectures in reliable embedded systems: challenges and solutions. e & i Elektrotechnik und Informationstechnik **125**, 401–405 (2008). URL http://dx.doi.org/10.1007/s00502-008-0587-z. 10.1007/s00502-008-0587-z
222. Pop, T., Pop, P., Eles, P., Peng, Z., Andrei, A.: Timing Analysis of the FlexRay Communication Protocol. Real-Time Systems **39**(1), 205–235 (2008)
223. Proakis, J.: Digital Communications, 4 edn. McGraw-Hill Science/Engineering/Math (2000)
224. Quinton, S., Graf, S.: Contract-based verification of hierarchical systems of components. In: Proc. of SEFM'08, pp. 377–381. IEEE Computer Society (2008)
225. Qureshi, T.N.: Enhancing Model-Based Development of Embedded Systems: Modeling, Simulation and Model-Transformation in an Automotive Context. Trita-mmk, issn 1400-1179; 2012:16, isbn 978-91-7501-465-4, Department of Machine Design, KTH - The Royal Institute of Technology, Sweden (2012)

226. Qureshi, T.N., Chen, D., Lönn, H., Törngren, M.: From EAST-ADL to AUTOSAR. Tech. Rep. TRITA-MMK 2011:12, ISSN 1400-1179, ISRN/KTH/MMK/R-11/12-SE, Mechatronics Lab, Department of Machine Design, KTH, Stockholm, Sweden (2011)
227. Qureshi, T.N., Chen, D.J., Persson, M., Törngren, M.: Towards the Integration of UPPAAL for Formal Verification of EAST-ADL Timing Constraint Specification. In: Workshop on Time Analysis and Model-Based Design, from Functional Models to Distributed Deployments (2011)
228. Qureshi, T.N., Chen, D.J., Törngren, M.: A Timed Automata-based Method to Analyze EAST-ADL Timing Constraint Specifications . In: Modelling Foundations and Applications, *Lecture Notes in Computer Science*, vol. 7349, pp. 303–318. Springer Berlin / Heidelberg (2012)
229. Raclet, J.B., Badouel, E., Benveniste, A., Caillaud, B., Legay, A., Passerone, R.: Modal interfaces: Unifying interface automata and modal specifications. In: Proceedings of the Ninth International Conference on Embedded Software (EMSOFT09), pp. 87–96. Grenoble, France (2009)
230. Raclet, J.B., Badouel, E., Benveniste, A., Caillaud, B., Legay, A., Passerone, R.: A modal interface theory for component-based design. Fundamenta Informaticae **108**(1–2), 119–149 (2011). DOI 10.3233/FI-2011-416
231. Raclet, J.B., Badouel, E., Benveniste, A., Caillaud, B., Passerone, R.: Why are modalities good for Interface Theories? In: Proceedings of the Ninth International Conference on Application of Concurrency to System Design (ACSD09), pp. 119–127. Augsburg, Germany (2009)
232. Racu, R., Li, L., Henia, R., Hamann, A., Ernst, R.: Improved response time analysis of tasks scheduled under preemptive round-robin. In: Conf. on Hardware-Software Codesign and System Synthesis (2007)
233. Ravindran, K., Ghosal, A., Limaye, R., Wang, G., Yang, G., Andrade, H.: Analysis Techniques for Static Dataflow Models with Access Patterns. In: Proc. of the 2012 Conference on Design & Architectures for Signal & Image Processing, DASIP '12 (2012)
234. Reimann, F., Kern, A., Haubelt, C., Streichert, T., Teich, J.: Echtzeitanalyse Ethernet-basierter E/E-Architekturen im Automobil. In: GMM-Fachbericht – Automotive meets Electronics (AmE), vol. 64, pp. 9–14 (2010)
235. Richter, K.: Compositional scheduling analysis using standard event models. Ph.D. thesis, Technical University of Braunschweig, Department of Electrical Engineering and Information Technology (2004)
236. Richter, K., Ziegenbein, D., Jersak, M., Ernst, R.: Model Composition for Scheduling Analysis in Platform Design. In: Proceedings of the 39th Conference on Design Automation (DAC 2002), pp. 287–292 (2002)
237. Ritz, S., Pankert, M., Meyr, H.: Optimum vectorization of scalable synchronous dataflow graphs. In: Proceedings of the International Conference on Application Specific Array Processors (1993)
238. ROBOCOP Team: ROBOCOP project. http://www.hitech-projects.com/euprojects/robocop/deliverables.htm
239. Rox, J., Ernst, R., Giusto, P.: Using timing analysis for the design of future switched based Ethernet automotive networks. In: Proceedings of Design, Automation and Test in Europe (DATE 12), pp. 57–62 (2012)
240. Saksena, M., Karvelas, P.: Designing for schedulability: integrating schedulability analysis with object-oriented design. In: Proceedings of the 12th Euromicro conference on Real-time systems, Euromicro-RTS'00, pp. 101–108. IEEE Computer Society, Washington, DC, USA (2000). URL http://dl.acm.org/citation.cfm?id=1947412.1947431
241. Sandell, M., van de Beek, J.J., Brjesson, P.O.: Timing and Frequency Synchronization in OFDM Systems Using the Cyclic Prefix. In: Proc. Int. Symp. Synchronization, pp. 16–19 (1995)
242. Sangiovanni-Vincentelli, A., Damm, W., Passerone, R.: Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. European Journal of Control **18**(3), 217–238 (2012). DOI 10.3166/EJC.18.217-238

243. Sangiovanni-Vincentelli, A., Di Natale, M.: Embedded system design for automotive applications. IEEE Computer **40**(10), 42–51 (2007)
244. Scheickl, O., Rudorfer, M.: Automotive Real Time Development Using a Timing-augmented AUTOSAR Specification. In: ERTS 2008
245. Schioler, H., Jessen, J., Nielsen, J.D., Larsen, K.G.: Network Calculus for Real Time Analysis of Embedded Systems with Cyclic Task Dependencies. In: Proceedings of the 20th International Conference on Computers and Their Applications (CATA 2005), pp. 326–332 (2005)
246. Schliecker, S., Ernst, R.: A recursive approach to end-to-end path latency computation in heterogeneous multiprocessor systems. In: Conf. on Hardware Software Codesign and System Synthesis (CODES-ISSS) (2009)
247. Schmidt, D.: Guest editor's introduction: Model-driven engineering. Computer **39**(2), 25–31 (2006). DOI 10.1109/MC.2006.58
248. Sedgewick, R.: Algorithms in C, Part 5: Graph Algorithms. Addison-Wesley (2002)
249. Selic, B.: The pragmatics of model-driven development. Software, IEEE **20**(5), 19–25 (2003). DOI 10.1109/MS.2003.1231146
250. Sentilles, S., Vulgarakis, A., Bures, T., Carlson, J., Crnkovic, I.: A Component Model for Control-Intensive Distributed Embedded Systems. In: 11th International Symposium on Component Based Software Engineering (CBSE2008), pp. 310–317. Springer (2008)
251. Shen, C., Plishker, W., Bhattacharyya, S.S.: Dataflow-based design and implementation of image processing applications. In: L. Guan, Y. He, S. Kung (eds.) Multimedia Image and Video Processing, second edn., pp. 609–629. CRC Press (2012). Chapter 24
252. Shen, C., Plishker, W., Wu, H., Bhattacharyya, S.S.: A lightweight dataflow approach for design and implementation of SDR systems. In: Proceedings of the Wireless Innovation Conference and Product Exposition, pp. 640–645. Washington DC, USA (2010)
253. Shen, C., Wu, S., Sane, N., Wu, H., Plishker, W., Bhattacharyya, S.S.: Design and synthesis for multimedia systems using the targeted dataflow interchange format. IEEE Transactions on Multimedia **14**(3), 630–640 (2012)
254. Sifakis, J.: A framework for component-based construction. In: Proc. of SEFM'05, pp. 293–300. IEEE Computer Society (2005)
255. SPEEDS Consortium: Home page. http://www.speeds.eu.com
256. Stappert, F., Jonsson, J., Mottok, J., Johansson, R.: A Design Framework for End-To-End Timing Constrained Automotive Applications. Embedded Real-Time Software and Systems (ERTS'10) (2010)
257. Stein, S., Hamann, A., Ernst, R.: Real-time property verification in organic computing systems. In: Second Int'l. Symp. on Leveraging Applications of Formal Methods, Verification and Validation (2006)
258. Stein, S., Neukirchner, M., Schrom, H., Ernst, R.: Consistency challenges in self-organizing distributed hard real-time systems. In: Workshop on Self-Organizing Real-Time Systems (SORT) (2010)
259. Steinbach, T., Kenfack, H.D., Korf, F., Schmidt, T.C.: An extension of the OMNeT++ INET framework for simulating real-time ethernet with high accuracy. In: Proceedings of the International Conference on Simulation Tools and Techniques (ICST), pp. 375–382 (2011)
260. Störrle, H.: Semantics of UML 2.0 activities with data-flow (2004)
261. Streichert, T., Buntz, S., Leier, H., Schmerler, S.: Short and long term perspective for Ethernet for vehicle-internal communication. 1st Ethernet & IP Automotive Techday (2011)
262. Streubühr, M., Gladigau, J., Haubelt, C., Teich, J.: Efficient Approximately-Timed Performance Modeling for Architectural Exploration of MPSoCs. In: Advances in Design Methods from Modeling Languages for Embedded Systems and SoC's, vol. 63, pp. 59–72. Springer (2010)
263. Stuijk, S., Geilen, M., Basten, T.: Exploring Trade-offs in Buffer Requirements and Throughput Constraints for Synchronous Dataflow Graphs. In: Proc. of DAC '06, pp. 899–904 (2006)
264. System Modeling Language. http://www.omg.org/sysml
265. Tarjan, R.: Depth-first Search and Linear Graph Algorithms. SIAM Journal on Computing **1**(2), 146–160 (1972)

266. Texas Instruments, Inc.: TMS320C6678 Multicore Fixed and Floating-Point Digital Signal Processor Data Manual (2012)
267. The ATESST2 Consortium: EAST ADL 2.0 Specification. Project Deliverable D4.1.1. http://http://www.east-adl.info/repository/EAST-ADL2.1/EAST-ADL-Specification_2010-06-30.pdf (2010)
268. The MAENAD Consortium: Language Concepts Supporting Engineering Scenarios. Project Deliverable D3.1.1. http://www.maenad.eu/publications.html (2012)
269. The Mathworks, Inc.: Matlab simulink. www.mathworks.com
270. The MathWorks Inc.: Simulink User's Guide (2005). URL http://www.mathworks.com
271. Theelen, B.D., Geilen, M.C.W., Basten, T., Voeten, J.P.M., Gheorghita, S.V., Stuijk, S.: A Scenario-aware Data Flow Model for Combined Long-run Average and Worst-case Performance Analysis. In: Proc. of MEMOCODE'06, pp. 185–194 (2006)
272. Thiele, L., Chakraborty, S., Naedele, M.: Real-time calculus for scheduling hard real-time systems. In: Symp. on Circuits and Systems (ISCAS) (2000). DOI 10.1109/ISCAS.2000.858698
273. TIMMO Consortium: TIMMO-2-USE. Http://www.timmo-2-use.org/
274. TIMMO Consortium: TADL: Timing Augmented Description Language, Version 2. TIMMO (TIMing MOdel), Deliverable 6 (2009)
275. TIMMO Consortium: TIMMO Methodology , Version 2. TIMMO (TIMing MOdel), Deliverable 7 (2009). The TIMMO Consortium
276. TIMMO Consortium: Mastering Timing Information for Advanced Automotive Systems Engineering – In the TIMMO-2-USE Brochure (2012). http://www.timmo-2-use.org/pdf/T2UBrochure.pdf
277. Tindell, K.: Adding Time-Offsets to Schedulability Analysis. Tech. rep., Department of Computer Science, University of York, England (1994)
278. Tindell, K., Burns, A., Wellings, A.: Calculating Controller Area Network (CAN) Message Response Times. Control Engineering Practice **3**, 1163–1169 (1995)
279. Tindell, K.W.: An extendible approach for analysing fixed priority hard real-time systems. Journal of Real-Time Systems **6**, 133–152 (1994)
280. Törngren, M., Chen, D., Malvius, D., Axelsson, J.: Automotive Embedded Systems Handbook, chap. Model-Based Development of Automotive Embedded Systems. CRC Press (2009)
281. Tripakis, S., Andrade, H., Ghosal, A., Limaye, R., Ravindran, K., Wang, G., Yang, G., Kornerup, J., Wong, I.: Correct and Non-Defensive Glue Design using Abstract Models. In: Proc. of the Seventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS '11, pp. 59–68. ACM, New York, NY, USA (2011)
282. Tripakis, S., Lickly, B., Henzinger, T.A., Lee, E.A.: On relational interfaces. In: Proc. of EMSOFT'09, pp. 67–76 (2009)
283. Ulversoy, T.: Software defined radio: Challenges and opportunities. Communications Surveys Tutorials, IEEE **PP**(99), 1–20 (2010). DOI 10.1109/SURV.2010.032910.00019
284. Unified Modeling Language, 2.0. http://www.omg.org/uml
285. Velasco, M., Martí, P., Bini, E.: Control-driven tasks: Modeling and analysis. In: Proceedings of the 2008 Real-Time Systems Symposium, RTSS '08, pp. 280–290. IEEE Computer Society, Washington, DC, USA (2008)
286. Vidal, J., de Lamotte, F., Gogniat, G., Soulard, P., Diguet, J.P.: A co-design approach for embedded system modeling and code generation with uml and marte. In: Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09., pp. 226–231 (2009)
287. Volvo AB: Volvo Construction Equipment. http://www.volvoce.com
288. Wang, S., Merrick, J.R., Shin, K.G.: Component Allocation with Multiple Resource Constraints for Large Embedded Real-Time Software Design. In: IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 219–226 (2004)
289. Wang, S., Shin, K.G.: Task Construction for Model-Based Design of Embedded Control Software. IEEE Transactions on Software Engineering **32**(4), 254–264 (2006)
290. Ward, P., Mellor, S.: Structured Development for Real-time Systems. Prentice Hall (1985)

291. Wu, S., Shen, C., Sane, N., Davis, K., Bhattacharyya, S.: Parameterized scheduling for signal processing systems using topological patterns. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, pp. 1561–1564. Kyoto, Japan (2012)
292. Xilinx Inc.: System Generator for DSP: Getting Started Guide. URL www.xilinx.com
293. Xilinx Inc.: Xilinx Core Generator, ISE Design Suite 12.1. Xilinx Inc. (2010)
294. Yang, X.: A feasibility study of uml in the software defined radio. In: Electronic Design, Test and Applications, 2004. DELTA 2004. Second IEEE International Workshop on, pp. 157–162 (2004). DOI 10.1109/DELTA.2004.10050
295. Yices: An SMT Solver. http://yices.csl.sri.com/
296. Yovine, S.: Kronos: A verification tool for real-time systems. (kronos user's manual release 2.2). International Journal on Software Tools for Technology Transfer **1**, 123–133 (1997)
297. Zaki, G., Plishker, W., Bhattacharyya, S., Clancy, C., Kuykendall, J.: Vectorization and mapping of software defined radio applications on heterogeneous multi-processor platforms. In: Proceedings of the IEEE Workshop on Signal Processing Systems, pp. 31–36. Beirut, Lebanon (2011)
298. Zhao, Y.: On the Design of Concurrent, Distributed Real-Time Systems. Ph.D. thesis, EECS Department, University of California, Berkeley (2009)
299. Zhu, Y., Sun, Z., Wong, W.F., Maxiaguine, A.: Using uml 2.0 for system level design of real time soc platforms for stream processing. In: Embedded and Real-Time Computing Systems and Applications, 2005. Proceedings. 11th IEEE International Conference on, pp. 154–159 (2005). DOI 10.1109/RTCSA.2005.101
300. Zivojnovic, V., Ritz, S., Meyr, H.: Retiming of DSP programs for optimum vectorization. In: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, pp. 492–496 (1994)
301. 3GPP LTE: The Mobile Broadband Standard (2008). URL http://www.3gpp.org/