Model-based Design Space Exploration for RTES with SysML and MARTE

Marcello Mura, Luis Gabriel Murillo, Mauro Prevostini
ALaRI, Faculty of Informatics
University of Lugano (USI)
via G. Buffi 13, CH-6904 Lugano, Switzerland

Abstract

The features of the emerging modeling languages for system design allow designers to build models of almost any kind of heterogeneous hardware-software systems, including Real Time Embedded Systems (RTES). An important goal to achieve is the implementation and use of these models in all the steps of a common design flow. One of these steps is the Design Space Exploration (DSE), which helps designers in discovering the optimal solutions among all possible combinations after mapping functional to architectural specifications; for RTES this step is particularly hard as it should include scheduling analysis in order to proof the time validity after the mapping. This paper presents some guidelines on how to use SysML and MARTE profiles to identify design points fulfilling the timing constraints of an RTES, and thus allowing to automate DSE analysis within the system design phase.

1 Introduction

As complexity of electronic applications constantly increases and features like time-to-market, time-in-market and quality assume an ever growing importance, the use of advanced design methodologies and description languages is necessary in order to speed up the design time, to minimize the development risks and in general to deal with systems development. Lately Model Driven Development techniques have evolved to support the modeling of Embedded Systems, providing the mechanisms necessary to successfully describe all the structural and behavioral specifications of the system in a model. Furthermore, when the design of an RTES is involved, it is also necessary to annotate the models with qualitative and quantitative requirements, known as Non-Functional Properties (NFP), aiming at verifying and validating e.g. the temporal behavior, the fulfillment of the time constraints, the power estimation, etc. Systems Modeling Language (SysML) [2] and Modeling and Analysis of Real-Time and Embedded systems (MARTE) [1], are two profiles of UML2.0 [4] meant to be used in this field. Basically SysML is used when modeling from a system engineering point of view representing system architectures and linking them with their behavioral components. Taking advantage of concepts like Requirements, Blocks, Flow Ports, Parametric Diagrams and Allocations it is easy to achieve a profitable way to model embedded systems. MARTE on the other hand was created to replace the profile for Schedulability, Performance, and Time (UML profile for SPT), and aims at providing support for specification, design, validation and verification stages in RTES development. Not only modeling domain but also analysis domain is comprised within the specification. Guidelines are given on how to annotate models with information required to perform time based analysis, namely performance and schedulability analysis. These analysis estimate the capability of a real-time system to perform tasks accomplishing their time constraints, to handle the required frequency of requests, and to work under other specialized conditions. The goal of this paper is to show an approach which combines embedded systems modeling by means of SysML, MARTE and Design Space Exploration (DSE) to help designers in evaluating the hardware/software partitioning solutions. In particular we show how to use SysML in order to include information useful for the construction of a Design Space that is then restricted by means of MARTE non-functional properties annotation and schedulability analysis, shrinking it to the design points fulfilling timing requirements. The paper is organized as follows: Section 2 describes the state-of-the-art and related works in the area of UML combined with DSE activities. Section 3 describes how to add information, useful for DSE, into SysML diagrams, in particular for allocation and binding. It also shows how to model systems for DSE by means of a mathematical structure. Section 4 shows how to model the schedulabilty analysis by means of MARTE profile and how to combine it with SysML. Section 5 illustrates our approach with a case study. Section 6 draws our conclusions and briefly outlines future work.

2 State-of-the-art and related works

In this section we describe the state-of-the-art and related works that combine UML RTES with DSE as in [11] where modelling patterns to compose systems for DSE are presented or in [9]. In this last methodology, the analysis and synthesis tools available in the Metropolis framework [20] were used to support a DSE. This work relies on simulation to obtain performance estimations. Some related works, dealing with real time analysis using UML-SPT profile [15], namely the previous standard UML profile (based on UML1.x) to model time, or some proprietary real-time modeling using UML are [10], [13], [21], [22], [23], [25], and [26]. Recent works using MARTE profile are: [6] that introduces the Time Model subprofile of MARTE; [27] introduces concisely the Hardware Resource Model profile and then presents an appropriate methodology to apply it during the hardware design process; [8] highlights the expressivity of the MARTE notation for the modeling of regular distributions and clarify its usage through examples and comparisons to other distribution notations such as in High Performance Fortran. Moreover [14] and [5] deal with UML-based hardware/software partitioning. [19] discusses the main phases of a co-design methodology and their use of MARTE, while [17] is about the scheduling analysis view extending the usual design view, focusing in particular on the specification and analysis of real-time aspects. We started from our previous [12] in which an approach to the use of SysML for Design Space Exploration analysis, within a system design phase through a mathematical structure, is described. The paper deals only with *allocation* and *binding* while *scheduling* is completely missing. In this work we will improve this mathematical structure and we will make use of MARTE profile to describe the schedulability analysis. Moreover we will show the interesting complementarity between SysML and MARTE.

3 Modeling Approach

This section describes our modeling approach which fits within a system engineering process. The work-flow used is described in Figure 1 and it starts with a requirements specification phase which includes a requirements analysis producing the refined requirements. The refined requirements are used as input for the system-level design phase where the designer has to describe the system by means of SysML diagrams like Internal Block, Requirements, Activity and Allocation diagrams. Once the system design phase has been completed and all diagrams are available, the work-flow continues with the generation of an XMI (XML Metadata Interchange) file from where the designer will extract relevant data for the DSE phase. At this moment we need to build the design space by computing all design points consisting of an *allocation* (selection of components), a *binding* (assignment of tasks to selected components) and a *scheduling* (execution order for the tasks). Then the designer has to compute the optimal Pareto-Points [7] that isolate the optimal solutions that are not comparable to each other. The designer can select the most appropriate one according to his experience. The idea of our approach is to propose how to represent DSE data within SysML and MARTE diagrams, so that data can be extracted and reused by a DSE tool. Essentially we have to figure out how to represent *allocation* and *binding* by means of SysML diagrams, in particular using Requirement and Internal Block diagrams. The *scheduling* will be represented by means of MARTE Schedulability Analysis Modeling diagrams. When the designed system has been required to complete some tasks accomplishing hard temporal constraints, a common DSE is not enough since all possible combinations won't give solutions with temporal correctness. It is necessary to select only the Design Space points corresponding to allocation options satisfying certain temporal constraints (e.g., deadlines, miss ratios) defined for the entire system or for a group of individual units. The selection of such points could be made based on the results of a time quantitative analysis performed over some of the execution characteristics of each task and over some of the behavioral characteristics of resources, or better, using Schedulability Analysis means.

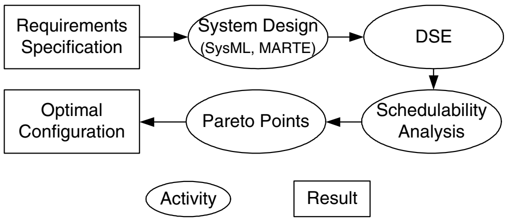


Figure 1. Work-flow used in our approach

3.1 Modeling Systems for DSE

This section, based on our previous [12], deals with the formalization of the DSE with a five tuple structure *M* and mapping of the structure to SysML diagrams. The *DSE structure M* is defined as follows:

$M = \{H, T, S, C, B\}$ where:

- *H* is the set of available hardware components.
- *T* is the set of tasks that the system has to perform.
- $S \subseteq (H \times T)$ is a specification relation where for each hardware component $h \in H$ there is *at least* a task $t \in T$ such that $S(h,t)$. This relation describes the possible combinations of *allocation* and *binding*.
- *C* is the set of virtual components where $|C| = |T|$. The virtual components are necessary in order to describe the allocation relation by means of SysML diagrams.

- $B \subseteq$ (*C* × *T*) is a binding relation that must be total, that is, for every virtual component $c \in C$ there is a task $t \in T$ such that *B(c,t)*.

The *DSE structure M* is then described by means of SysML diagrams. The steps to be performed are as follows:

1. Describe *B* by means of an Internal Block diagram. The Internal Block Diagram is used in order to describe the NFP of each task.
2. Describe *H* by means of a Requirement diagram. The Requirement diagram is used in order to describe the technical specifications of *H* by means of parameters like e.g. *Execution Time* and *Cost*.
3. Describe *S* by means of an Internal Block diagram where each $h \in H$ is linked to a virtual component *c* $\in C$ according to the binding relation *B*.

The three steps described above are represented below in order to explicitly show how *M = {H, T, S, C, B}* should be modeled. Assume that:
$H = \{h_1, ..., h_k\}$ $\quad T = \{t_1, ..., t_n\}$ $\quad S = \{(h_i, t_j)\}$
are respectively: the set of hardware components available to the designer, the set of tasks to be performed by the system and the specification relation *H* × *T*. Then
$C = \{c_1, ..., c_n\}$ and $B = \{(c_i, t_j)\}$ are the set of virtual components and the binding relation where $(c_i,t_j) \in$ (*C* × *T*) such that *i=j* with *i,j* ∈ {1, ... , n}.
The related SysML diagrams according to the above three steps will be expressed as explained in Section 5. In this section we described a possible solution to represent *allocation* and *binding* by means of SysML diagrams. In order to represent *scheduling* we propose to use MARTE profile as described in the following section.

4 Time Analysis

The time analysis techniques are targeted to estimate both the capability of a real time system to give satisfactory responses according to the time constraints, and the capability to manage the required amount of operations. In particular, during the early design phases, Schedulability Analysis is used to discover if a concrete resource is able to complete its assigned tasks before the expiration of deadline times. This helps developers to detect potentially unfeasible real-time architectures, to improve the behavior of the system or to prevent costly design mistakes, particularly related to timing behavior. On the other hand, a successive analysis of an implemented system allows designers to discover (with more precise quantitative information of the system) temporal-related faults [16]. The growing complexity of real time systems has led to the development of a wide set of mathematical formalisms to perform Schedulability Analysis, like the Rate Monotonic Analysis (RMA) [18], holistic techniques, or extended timed automata, which are implemented by tools available in the market. The challenging point of all these processes during the last years has been to find the proper methodology to create Model Driven Architecture specifications with the intention to link them to the analysis tools and the selection of highly-meaningful information to annotate the models [24]. To enable Schedulability Analysis, a UML model must specify the system-level operations with time constraints, the frequency of requests, and the conditions of execution (which depend on the environment); and using the new OMG specification for Modeling and Analysis of RTES is a way to attach to such models all information required to perform timing analysis, as shown in the next section.

4.1 MARTE for Schedulability Analysis

MARTE offers the possibility to precisely describe RTES by means of its different and useful components (Time Modeling, Allocation Modeling, Resource Modeling) in terms of design domain, but this specification was also tailored to bridge the gap between the previously commented design domain and the analysis domain, in order to create a link providing feedback to improve the design quality. Composed by three main components, MARTE Foundations, MARTE Design Model and MARTE Analysis Model, this UML profile reserves great portion of its features to provide the capability of quantitative properties specification, and the way to use them for time analysis. In concrete, and due to the purpose of this study, the Non-Functional Properties Modeling (NFPs) contained in MARTE Foundations component and the Schedulabilty Analysis Modeling (SAM) from the MARTE Analysis Model are the points of interest when modeling RTES for DSE with validation and verification of time constraints, as we will expose in the following sections.

4.2 NFPs for DSE Oriented Modeling

During the construction of a model it is a fundamental practice to attach the externally visible properties to the model elements. These so called properties comprise both functional properties and non-functional properties. The non-functional properties, are essential for the quantitative analysis of the system. NFPs provide information about different characteristics like throughput, delays, overheads, execution times and scheduling policies [10]. Both qualitative and quantitative NFPs are used for schedulability analysis, since it is essential to have specifications of numerical measurable characteristics as well as specifications of other high level abstraction characterizations. System models designed for DSE have the same requirements and the same needs in terms of property annotations as other RTES, and since the NFP Modeling component of MARTE provides a framework targeted to flexibly and effectively support and attach NFPs to model elements, we conclude that this profile is the best way to annotate our model with properties that will help us to get the time-correct Design Space starting from a SysML description.

4.3 SAM in DSE Oriented Modeling

The Schedulability Analysis Model (SAM) allows quantitative annotations to be attached at the level of detail desired by the designer. Furthermore it enables a set of modeling techniques and steps to achieve the highest communication possible between the analysis tool and the model itself [16]. The SAM domain model was developed over the concept of Analysis Context, which collects all the relevant data and is the starting point for an analysis tool to extract information needed to analyze the model. Analysis Context is associated with two modeling concerns: Workload Behavior (WB) and Resources Platform (RP). WB (see e.g. Figure 5) is addressed by the SAM_Workload package, which allows to represent and to describe a defined set of end-to-end processing flows corresponding to specific modes of system operations or specific situations leading to defined intensities of environmental events. The RP concern consists in inserting scheduling related annotations to a set of processing resource instances in order to be used during the schedulability analysis. As explained in Figure 7 the schedulable (or virtually schedulable) resources (tasks) are written as components allocated on processing resources, with a given priority parameter or value (depending on the scheduling policy). In DSE oriented models the description of a workload behavior is necessary to prepare the model for schedulability analysis, and it is achieved using diagrams representing the characteristics of the arrival events and the time characteristics with annotations of NFPs (arrival period of external events, response time, end-to-end deadline, execution time). The use of variable NFPs in the annotations is also useful to complete the specification since the analysis tool is able to return some results to the model after the schedulability analysis (e.g. utilization and execution times). These results will be used later to shrink the Design Space. Internally, each of the behavioral scenarios that compose the end-to-end flows should be described by state machines, activity and sequence diagrams (using the stereotype «gaScenario») with NFPs annotations related to execution time (best and worst) and message sizes of each step (message) in the scenario. In our approach, the end-to-end flows should be detected and their NFPs should be annotated using a «gaWorkload Behavior» diagram. On the other hand, the annotations regarding the behavior scenario should be made in the activity diagram representing the task chain, if each task is executed after another task independently. If there is active interaction and communication between the tasks, the utilization of a sequence diagram is mandatory. Finally with the inclusion of a final diagram, stereotyped as «saAnalysisContext», we define the variables to get results for certain parameters out of the schedulability analysis (variable for WCET report, flag to indicate schedulable resources). This is the last step to turn the model into a schedulability analyzable model. Taking again our approach, we reference some variables previously defined in the WB context. A schedulability analysis for all possible combinations in the design space should be performed, and these points should be withdrawn or not according to the value of the Analysis Context returned by the analysis tool. In that way it is possible to build a design space made exclusively of points satisfying all the timing constraints of the real time system. Then, the Pareto Points could be calculated and finally the designer can select the implementation of the system which better fits the requirements in terms of cost, area, power or any other metric used for the DSE.

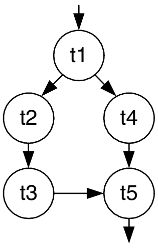
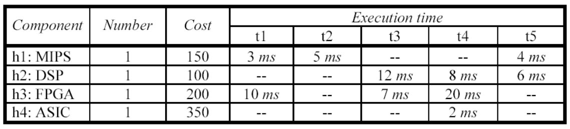


Figure 2. Task-chain of a system specification



Component	*Number*	*Cost*	*Execution time*				
			t1	t2	t3	t4	t5
h1: MIPS	1	150	3 *ms*	5 *ms*	--	--	4 *ms*
h2: DSP	1	100	--	--	12 *ms*	8 *ms*	6 *ms*
h3: FPGA	1	200	10 *ms*	--	7 *ms*	20 *ms*	--
h4: ASIC	1	350	--	--	--	2 *ms*	--

Figure 3. Available hardware components with costs and execution times for tasks

5 Case Study

This section discusses a case study with the aim to describe a system by means of SysML and MARTE profiles following the formalism explained in Sections 3 and 4. Consider the task chain shown in Figure 2. It depicts a system specification with a set of tasks $t_i \in T$ that can be executed by different hardware components $h_i \in H$. The various components to execute the tasks together with the costs and the execution times are given in Figure 3. The only constraint that we impose on the system is that every hardware component executes the tasks sequentially. Namely, at any given time a component can execute at most one task.

5.1 Implementation of our approach

Taking into account the constraints and the requirements of the system shown in Figures 2 and 3, we will now

978-1-4244-2265-4/08/$25.00 © 2008 IEEE

apply our approach by following the three steps described in Section 3.1. To accomplish DSE, we start with the construction of the *DSE structure M*, which is given by

$$M = \{H, T, S, C, B\}$$

In the following subsections, the possible values that each element in the *DSE structure M* can take, is given.

Description of ***B*** The binding relation $B \subseteq (C \times T)$ is defined as follows:

$B = \{(c_1,t_1), (c_2,t_2), (c_3,t_3), (c_4,t_4), (c_5,t_5)\}$

and the related Internal Block diagram, showing the relation between virtual components and NFP of the tasks, is shown in Figure 4.

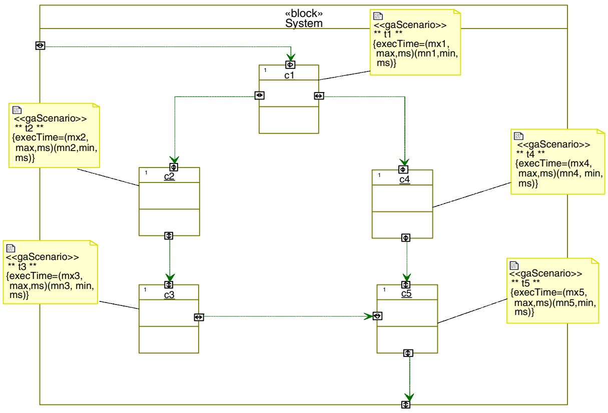

«block»
System
<<gaScenario>>
** t1 **
{execTime=(mx1, max,ms)(mn1,min, ms)}
1
c1
<<gaScenario>>
** t2 **
{execTime=(mx2, max,ms)(mn2,min, ms)}
1
c2
<<gaScenario>>
** t4 **
{execTime=(mx4, max,ms)(mn4, min, ms)}
1
c4
<<gaScenario>>
** t3 **
{execTime=(mx3, max,ms)(mn3, min, ms)}
1
c3
1
c5
<<gaScenario>>
** t5 **
{execTime=(mx5, max,ms)(mn5,min, ms)}

Figure 4. Internal Block diagram describing the activities allocations

Description of ***H*** The set of hardware components is defined as follows:

$$H = \{h_1, h_2, h_3, h_4\}$$

and the related Requirements are shown within Figure 7.

Description of ***S*** The specification relation $S \subseteq (H \times T)$, according to Figure 3, is expressed as follows:

$S = \{(h_1,t_1), (h_1,t_2), (h_1,t_5), (h_2,t_3), (h_2,t_4), (h_2,t_5), (h_3,t_1), (h_3,t_3), (h_3,t_4), (h_4,t_4)\}$

and the related Internal Block diagram is shown in Figure 7. Figures 5 and 7 show how to integrate the SAM package in the modeling of RTES aiming to reach DSE. In both diagrams the NFPs beginning with $ are variable types returned by the analysis tool, and the remaining are constants which should be replaced by well known time characteristics of the system.

Due to the fact that at this point our approach already contains the allocation diagram, we only need to add the correct SAM stereotypes, and to annotate the priorities and the time characteristics of the resources, as shown in Figure 7. In the Analysis Context diagram shown in Figure 6 isSched_System defines the global scheduling correctness regarding all required deadlines annotated in the context under analysis, while the variable wcet_Report is a parameter of interest. Look that wcet_Report references a variable defined in the WB context.

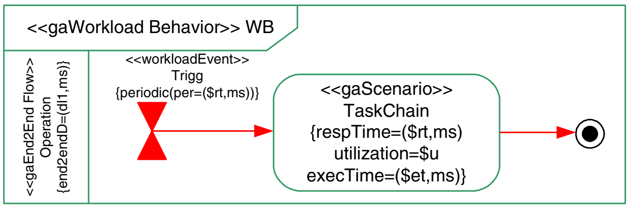


Figure 5. Workload Behavior representation

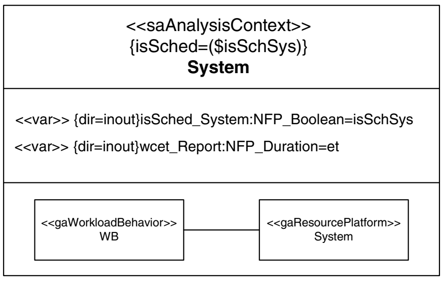

<<saAnalysisContext>>
{isSched=($isSchSys)}
System
<<var>> {dir=inout}isSched_System:NFP_Boolean=isSchSys
<<var>> {dir=inout}wcet_Report:NFP_Duration=et
<<gaWorkloadBehavior>>
WB
<<gaResourcePlatform>>
System

Figure 6. Analysis context diagram

6 Conclusions and Future Works

In this paper we presented a modeling approach to describe systems by means of SysML and MARTE profiles to address the DSE phase. Our contribution is towards the system-level design using SysML and MARTE by showing the complementarity of the two UML profiles and to encourage designers of embedded systems to develop systems in an efficient way by performing DSE activities in early design stages. Our approach aims at proposing a way in order to have in one set of diagrams all possible combinations of system configurations. Since formal specifications provide a way to express problems without ambiguity, we specified information relevant to DSE using a mathematical structure. This structure was then mapped on to subsets of SysML and MARTE diagrams. The main advantage of our approach is to have both DSE information and system model in one tool repository. As part of the future works we would like to automate the extraction of DSE information. Having the design space modeled by means of a SysML Internal Block diagram as in Figures 4 and 7, the optimal configuration can be identified by extracting information from an XMI file, which can be generated for example by the UML tool. At

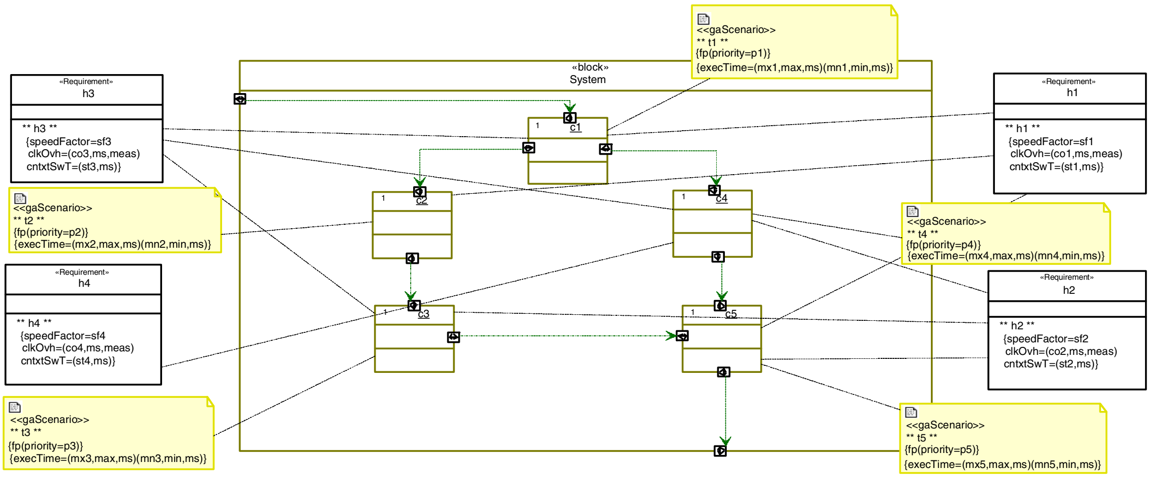


Figure 7. Resources Platform with specification relation and requirements

this point designers would be able to compute the optimal Pareto-Points and produce the final Internal Block diagram. Experimental measurements will be part of our future work, as well as the evaluation of this approach using different formalisms. In our work we used the UML/SysML tool named Rhapsody V7.1 from Telelogic [3].

References

[1] *http://www.omgmarte.org*.

[2] *http://www.sysml.org*.

[3] *http://www.telelogic.com*.

[4] *http://www.uml.org*.

[5] W. Ahmed and D. Myers. *Using UML 2.0 for the Creation of Concept-Based SoC Design*. UML-SoC'07.

[6] C. Andre, F. Mallet, and R. de Simone. *Time modeling in MARTE*. FDL'07.

[7] R. Bayton and R. Spence. *Sensitivity and Optimization*. Elsevier, 1980.

[8] P. Boulet, P. Marquet, E. Piel, and J. Taillard. *Repetitive Allocation Modeling with MARTE*. FDL'07.

[9] R. Chen, M. Sgroi, G. Martin, L. Lavagno, A. Sangiovanni-Vicentelli, and J. Rabaey. *Embedded System Design Using UML and Platforms*. In Proc. of Forum on specification and Design Languages, FDL'02.

[10] H. Espinoza, H. Dubois, S. Gérard, J. Medina, D. Petriu, and M. Woodside. *Annotating UML Models with Non-Functional Properties for Quantitative Analysis*. In Satellite Events at MoDELS 2005.

[11] O. Florescu, J. Voeten, M. Verhoef, and H. Corporaal. *Reusing Real-Time Systems Design Experience Through Modelling Pattern*. FDL'06.

[12] S. Ganesan and M. Prevostini. *Bridging the Gap between SysML and Design Space Exploration*. FDL'06.

[13] P. Green and M. Edwards. *The modeling of Embedded Systems Using HASoC*. In Proc. of DATE 02.

[14] P. Green and Y. Lu. *Hardware/Software Partitioning of UML Models*. UML-SoC'07.

[15] O. M. Group. *UML Profile for Schedulability, Performance, and Time*. In OMG document n. ptc/02-03-02, 2002.

[16] O. M. Group. *UML Profile for MARTE, Beta 1*. OMG Adopted Beta Specification ptc/07-08-04, 2007.

[17] M. Hagner and M. Huhn. *Tool Support for a Scheduling Analysis View*. UML Workshop at Date'08.

[18] M. Klein, T. Ralya, B. Pollak, R. Obenza, and M. Harbour. *A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems*. Kluwer Academic Publishers, 1993.

[19] A. Koudri, D. Aulagnier, D. Vojtisek, P. Soulard, C. Moy, J. Champeau, J. Vidal, and J. L. Lann. *Using MARTE in a Co-Design Methodology*. UML Workshop at Date'08.

[20] E. Lee. *Advanced Tool Architectures*. Chess Conferences Reviews, UC Berkeley, 2004.

[21] M. Natale and M. Saksena. *Schedulability analysis with UML*. In UML for Real: Design of Embedded Real- Time Systems, Kluwer Academic Publishers, 2003.

[22] D. Petriu and C. Woodside. *Performance analysis with UML: layered queueing models from the performance profile*. In UML for Real: Design of Embedded Real-Time Systems, Kluwer Academic Publishers, 2003.

[23] T. Phan, S. Gerard, and D. Lugato. *Scheduling Validation for UML-modeled Real-Time Systems*. ERCT 2003.

[24] B. Saksena and P. Karvelas. *Designing for schedulability: integrating schedulability analysis with object-oriented design*. 12th Euromicro Conference on Real-Time Systems, 2000.

[25] B. Selic. *A Generic Framework for Modeling Resources with UML*. In IEEE Computer, 2000.

[26] B. Selic and J. Rumbaugh. *Using UML for Modeling Complex Real- Time Systems*. In White paper, Rational (Object Time), 1998.

[27] S. Taha, A. Radermacher, S. Gerard, and J. Dekeyser. *MARTE: UML-based Hardware Design from Modeling to Simulation*. FDL'07.