**Proceedings of the 17th World Congress**
**The International Federation of Automatic Control**
**Seoul, Korea, July 6-11, 2008**

# The UML – MARTE Standardized Profile

**Sébastien Gérard*. Bran Selic.****

**CEA LIST, Saclay, F-91190*
*France (Tel: 303-555-5555; e-mail: sebastien.gerard@cea.fr).*
***Malina Software Corp., Nepean, Ontario K2V 1C8*
*Canada (e-mail: selic@acm.org)*

**Abstract:** The UML Profile for Modeling and Analysis of Real-Time and Embedded Systems (also called the UML profile for MARTE, or simply, MARTE) adds capabilities to UML for model-driven development of real-time and embedded systems (RTES). It provides support for specification, design, and verification/validation stages. This new profile replaces the existing UML Profile for Schedulability, Performance and Time (formal/03-09-01), which was based on an earlier version of UML (UML 1.x). MARTE defines the foundations for model-based description and analysis of real time and embedded systems. At its core is a set of fundamental concepts which are further refined for modeling and a variety of quantitative analyses. The modeling capabilities range from support for system specification through to detailed design of real-time and embedded systems. For the analysis portion, the intent is not to define new techniques for analyzing real-time and embedded systems, but to support the ones that currently exist as well as potential new ones. Hence, it provides facilities to annotate models with information required to perform specific types of analyses. In particular, MARTE focuses on performance and schedulability analysis. But, it also defines a general analysis framework which can be specialized for other kinds of analysis.

## 1. INTRODUCTION

The Object Management Group (OMG, www.omg.org) is one of the key international organizations promoting standards supporting the usage of model-based system development. The Unified Modelling language (UML [1]) standard is probably the most representative of these and has had significant success in the software industry as well as in other kinds of domain such as IT and financial systems. UML is now the most widespread language used for modelling in both industry and academia. But, because UML has been designed to be a general-purpose modelling language, it needs to be specialized to suit specific domains or applications. The real-time embedded (RTE) domain is one of these specific domains for which extensions to UML are required to provide more precise domain-specific concepts. Previously, the OMG had issued the UML profile for Schedulability, Performance and Time (SPT, [2]). This UML extension provided concepts for dealing with model-based schedulability analysis, focussed primarily on rate monotonic analysis, and also concepts for model-based performance analysis based on queuing theory. In addition, SPT also provided a rich framework for representing time and time-related mechanisms. However, experience with applying SPT revealed shortcomings within the profile in terms of its expressive power for modelling RTE phenomena. Furthermore, it was necessary to upgrade the profile to the new release of UML, UML2, and also to extend the scope of the profile. For example, it was necessary to support the design of both hardware and software aspects of embedded systems and a more extensive support for schedulability and performance analysis, encompassing additional techniques such as hierarchical scheduling. This has resulted in a Request For Proposals (RFP) for a new UML profile named MARTE (Modeling and Analysis of Real-Time and Embedded systems, [4]). The intent was to address these issues as well as compliance with the UML profile for Quality of Service and Fault Tolerance (QoS & FT, [3]), which allows specification of not only real-time constraints but also other embedded systems characteristics, such as memory capacity and power consumption. MARTE was also required to support modelling and analysis of component-based architectures, and also a variety of different computational paradigms (asynchronous, synchronous, and timed).

To cope with this new RFP, a group of OMG members decided to develop a joint submission. This consortium, called the ProMARTE consortium, consisted of the following enterprises: Alcatel, ARTISAN Software Tools, Carleton University, CEA LIST, ESEO, ENSIETA, France Telecom, International Business Machines, INRIA, INSA from Lyon, Lockheed Martin, MathWorks, Mentor Graphics Corporation, NoMagic, the Software Engineering Institute (Carnegie-Mellon University), Softeam, Telelogic AB, Thales, Tri-Pacific Software, and Universidad de Cantabria. This submission resulted in a proposal that was accepted and voted by the OMG in June 2007 [5].

The purpose of this paper is to provide an overview of the MARTE standard; readers who want to know more about this

new specification should refer to the OMG web site dedicated to MARTE at http://www.omgmarte.org.

## 2. MARTE IN A NUTSHELL

The design approach used to design MARTE was an iterative engineering process involving the definition of two key related artifacts. The first of these is a domain-specific meta-model[1] of the core MARTE concepts, their characteristics, and their mutual relationships. We will refer to this as the *domain model.* The second artifact was the actual *profile* definition, which is derived by mapping the domain model onto the corresponding elements of the UML2 meta-model. The purpose of the domain model is to provide a precise and formalized definition of the domain concepts, unfettered by concerns related to the pragmatic mechanisms of profile definition. The domain-specific constructs defined in this way are then expressed in the profile as UML stereotypes with domain-specific properties and, possibly, OCL constraints.

The MARTE profile itself is structured around two main areas of concern, one to model the features of real-time and embedded systems and the other to annotate application models so as to support analysis of system properties. These represent the two different ways of using UML profiles. One way is to provide a domain-specific language, that is, a language in which domain-specific concepts are provided as first-order language constructs. This allows users to construct models that express their ideas directly and succinctly (and, therefore, more accurately).

The other way of using UML profiles is to use them for domain-specific *interpretations* of existing UML models. This allows a given model to be viewed from the perspective of a specific set of concerns that may not have been taken into account when the model was originally developed. For instance, to determine the performance characteristics a design, it is useful to re-cast its UML model in the form of a queueing network model (e.g., a model which explicitly identifies the clients and servers in the system and their respective performance characteristics). This can be achieved with a suitable profile.

The structure of the profile is shown in Figure 1. The *MARTE design model*, provides a domain-specific language for modelling phenomena specific to RTE systems, while the *MARTE analysis model* is a viewpoint-based sub-profile suitable for model analysis. *MARTE foundations* provides a shared conceptual base for the two sub-profiles, such as the representation of time, the use of concurrent resources, quality of service modelling, and so on. In addition, MARTE defines numerous complementary cross-cutting modelling constructs which are collected in the *MARTE annexes* package.

[1] A *meta-model* is a model that specifies the set of modelling concepts (and relationships) of a modelling language.

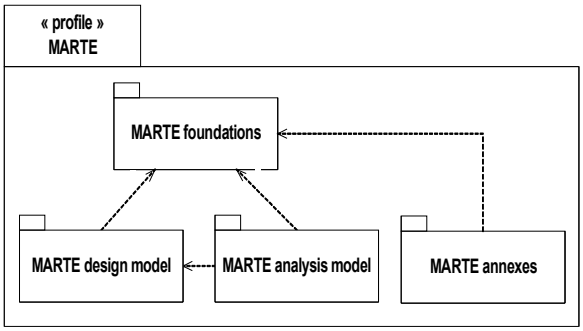


Fig. 1. Overall architecture of MARTE

The rest of this paper is then dedicated to tackle the three key concerns related to modelling real-time and embedded systems. The next section will introduce how MARTE supports modelling of *qualities of service* or *non-functional* properties, time, and concepts dedicated to supporting the modelling of architecture specifications. Section 4 covers the modelling of hardware and software platforms that support RTE systems, while section 5 deals with the sub-profile for model-based analyses. In conclusion, we describe some of the ongoing experiments with application of MARTE to industrial problems and outlines the main remaining challenges for MARTE.

*N.B.: We use previously the term "non-functional" to identify those characteristics of a system that primarily pertain to the quality of the functionality (e.g., speed, reliability, or security), because it has become commonplace. However, we recognize that this term is somewhat problematic since it is often interpreted that requirements in this category have a secondary status – something which is clearly not the case in the vast majority RTE systems.*

## 3. NFPs, TIME AND ARCHITECTURE

One of the main features that make the development of real-time systems different from most other kinds of software development is the significance of the so-called non-functional properties – and time in particular – which are most often as important for system correctness as are the functional aspects. To deal with this concern, MARTE has defined two frameworks: the *non-functional properties (NFP)* specification framework and the *time* modeling framework (Time). The former package consists of a set of modeling constructs to declare, qualify and apply semantically well-formed non-functional properties into UML models. It is complemented by the *Value Specification Language (VSL),* which defines a textual language for specifying algebraic and time expressions conforming to an extended system of data types. NFP supports the declaration of the non-functional properties as UML types, whereas VSL is used to describe the *values* of those types. The Time package contains both basic concepts to define time and its various forms as supported within MARTE, and also time-related phenomena such as time events and clocks.

Considering the time aspect, real-time systems are specifically concerned with two important features: the *cardinality* of time (e.g., delay, duration or clock time) and the qualitative *ordering* of events in time (e.g., event1 "happens before" event2). Consequently, MARTE defines

concepts for three different time models: *chronometric* time model, *logical* time, and *synchronous logical* time. The latter is a refinement of the logical time model with the additional assumption of the concept of simultaneity (i.e., the possibility of several events occurring at the same time). All three paradigms rely on partial ordering of instants.

In addition to time representation and NFP specification, the MARTE foundations provides two additional packages for architectural modelling. One of these defines a *general component model (GCM)*. GCM relies on a refinement of the UML structured classes and provides a common denominator among a number of widely used component models, which in general, do not target exclusively the RTE domain. The purpose is to provide a model that is as general as possible, and which is not tied to specific execution semantics, and to which real-time characteristics can be applied. The MARTE GCM relies mainly on UML structured classes, on top of which a support for SysML blocks has been added. Aligning GCM with Lightweight-CCM, AADL and EAST-ADL2 [5] have also influenced its definition.

*N.B.: SysML or Systems Modelling Language, is another OMG profile of UML adopted by the OMG, used for systems engineering.*

## 4. PLATFORM MODELING

Platform modelling is, of course, a key aspect in MARTE, since platforms can have a fundamental impact on the NFP characteristics of an RTE system. The MARTE approach is fully compatible with the general MDA approach [6]. Platform concerns are handled in the several places within MARTE, depending, first, on the purpose of the platform model and, second, on the granularity of the platform description. As depicted in Figure 2, MARTE deals with platform modelling concern in all three of its main parts, foundations, design, and analysis. The foundations part provides a basic framework for platform-based modelling, *General Resource Modelling (GRM)*. This is a distillation and generalization of the key concepts involved platform modelling at a very high-level abstraction. It is based on a straightforward design pattern, which represents platforms as a set of resources, possibly comprising finer-grained resources in hierarchical manner, each resource offering at least one service. The key concept here is the notion of a *resource* as *a service provider with finite capacity* – its finite nature stemming, ultimately, from the physical limitations of the underlying hardware (e.g., memory capacity, bandwidth, processing power). Resources may be instantiated and the resulting instances may provide resource services in response to service requests.

As denoted in Figure 2, GRM is refined differently for more detailed modelling or analysis purposes. The *Software Resource Model (SRM, [8])* and the *Hardware Resource Model (HRM, [9 and 10])* are dedicated to describing software and hardware computing platforms respectively. SRM consists of a set of concepts focussed on describing a model-based version of the API of real-time operating systems such as semaphores and concurrent tasks (processes). The MARTE annex contains some examples of such an API including OSEK and Arinc. The HRM provides concepts needed for describing hardware computing platforms at three levels of abstraction, serving primarily the following three use cases:

- *Software design and allocation* using a high level hardware description model of the targeted platform architecture. This abstraction level is intended to be a more formal alternative to the block diagrams often used for describing hardware platforms of embedded systems.
- *Analysis of real-time and embedded properties* of software-intensive systems executing on top of specialized hardware.
- *Simulation* of detailed hardware models; the required level of detail depends on the simulation accuracy desired.

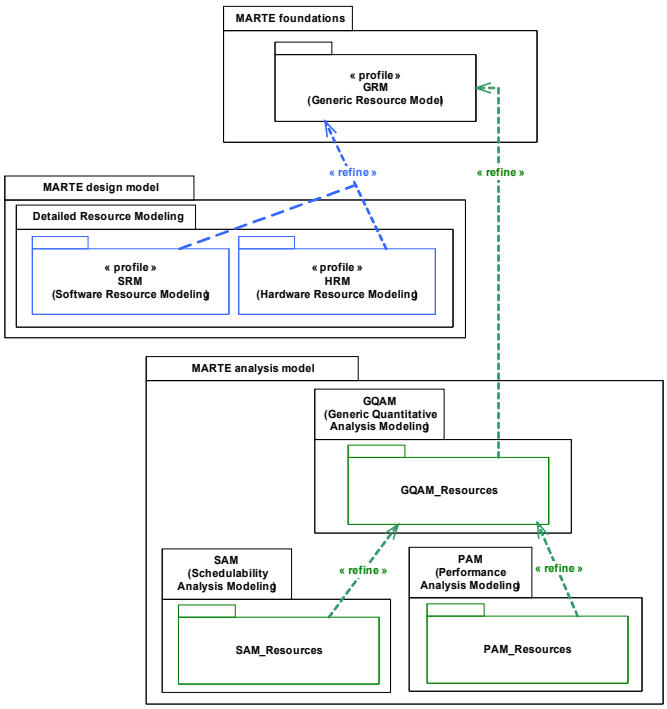


Fig. 2. MARTE support for platform modeling

Describing platforms is crucial for embedded systems design and analysis. Both analysis and design need the ability to denote the relationships between the platforms model and the functional application model. For that purpose, MARTE propose the *allocation* package, which defines the requisite allocation and refinement relationships. The allocation concepts of MARTE are aligned with those defined in SysML. However, whereas SysML allocation enables allocation of any kind of model elements to any other kind of model elements, MARTE restricts allocation to describe only the allocation of application elements to platform model elements. In addition, the allocation specification may further be refined by defining non-functional properties related to the allocation.

## 5. MODEL-BASED ANALYSIS

Model-based analysis is a key feature of MARTE. As noted earlier, the intent of MARTE was not to define new analysis techniques but to support the most popular existing real-time and embedded analysis techniques. These are mostly in the domain of schedulability and performance analysis. However, both the schedulability and the performance analysis packages rely on the generic one: the Generic Quantitative Analysis Model (GQAM). This model defines the basic UML extensions needed to annotate UML model in order to perform any kind of analysis. The expectation is that GQAM will be further specialized or refined in order to cope with analysis techniques that are not currently supported by MARTE.

The annotation mechanism used in MARTE to support model-based analysis uses UML stereotypes. These map model elements of the application description into the semantics of an analysis domain such as schedulability, and also give values for properties which are needed to carry out the analysis. It is possible to distinguish "input" properties, which are needed to carry out the analyses, and "output" properties which are the results provided by the analyses. However, the modeler may also input required values of output properties, which can then be used to determine how well a system meets its requirements. Analysis is not always simply "pass/fail", and the particular goals of analysis are specific to their domain. Output properties may include details of how and where time and resources are consumed in order to diagnose problems, and may include sensitivity studies to explore the importance of parameters whose values are uncertain.

The analysis sub-profile is intended to provide a foundation for applying transformations from UML models into a wide variety of analysis models. The environment for exploiting the profile should consist of a set of tools, including model transformers to map the technological space of the annotated UML model (the UML + MARTE design space in this case) to different possible analysis technological spaces (e.g., Rate Monotonic Analysis tools). The forward path of this process denotes the way a model is expected to be transformed (e.g., via the XMI output of the UML model) to the input format required by an analysis tool. The reverse path provides the feedback path, with the intent to re-import the analysis results into the original UML models, where they can be viewed in an application-specific context (versus an analysis-specific context).

## 6. ABOUT THE MARTE USAGE

As for the UML, one important feature of MARTE is to be methodology neutral. Different domains and applications may use MARTE in different ways and it is anticipated that domain-specific methodologies based on MARTE will eventually evolve. It is also important to note that MARTE was intentionally designed to not constrain the usage of the UML itself. For example, if you use the SRM profile to describe the API of a real-time operating system, you should have the option to use whatever UML concepts you deem appropriate, without restriction. In a sense, MARTE by itself is incomplete and should be accompanies by appropriate methodological guidance.

Among possible MARTE use cases, as an example, we mention multi-tasking system design, hardware platform simulation, model-based schedulability/performance analysis, as well as high-level architectural descriptions of RTES. For the latter use case, it is assumed that people will propose methodologies defining how to combine the usage of both UML profiles, MARTE and SysML, as for example in the EDONA project (www.edona.org.

Following the example of SPT (the UML profile for Schedulability, Performance and Time [OMGb]), MARTE fosters model processing in the way that enables adding semantics to a given UML model (e.g., for code production or quantitative analysis purposes). In this context, one generic usage of MARTE relies on the model-processing schema described in Figure 3.

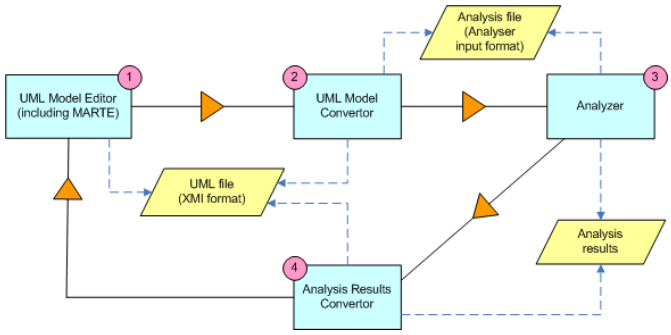


Fig. 3. Schema of the MARTE model-processing framework

The MARTE framework for model-processing relies on four components:

- The *UML Model Editor* is, of course, the initial component used to describe the application using the UML extended with MARTE. Modelers annotate their UML models with MARTE defined extensions. If, for example, the purpose is to make a schedulability analysis of a UML model, sthe model will be annotated with stereotypes of the SAM sub-profile of MARTE. To this end, the engineer needs to choose an appropriate analysis technique (e.g., Rate Monotonic Analysis) and a tool that supports that technique.
- The *UML Model Convertor* will then transform the annotated UML model (usually serialized using the XMI format) into a format understood by the analysis tool.
- The *Analyzer* component, fed by the previously generated files, performs the analysis and generates a result file. At this stage, it is possible to either analyse the results file directly or to convert it into a UML form
- The *Analysis Results Convertor* transforms the analysis results, either into corresponding UML forms (e.g., sequence diagrams if the Analyser was a test case generator), and may even insert them directly into the initial UML model.

RTES differ from other kinds of systems because platforms that support their them may have a critical role. Therefore, the effect of such platforms on the application have to be taken into account early in the development process to ensure that performance requirements and other key system characteristics are met. For that reason, platform modelling concerns play an important role in the MARTE specification. As noted previously, MARTE does support platform-based design through two dedicated sub-profiles, HRM and SRM. These sub-profiles are dedicated to modeling of hardware and software platforms respectively. The intent was not to describe new standard APIs for modelling computing platforms, but to support accurate modelling of platforms and their characteristics. The information included in such models can then be used by external tools for various purposes, such as generating optimized code.

## 7. MARTE NEXT STEPS

MARTE has been adopted as an official OMG specification as of the end of June 2007. However, this version is, in effect, a "beta" version that still needs to be validated by various tool vendors. This finalization step will yield the official validated 1.0 version. This is being undertaken by an OMG *Finalization Task Force (FTF),* which was launched in July 2007. The work of this team is divided into two phases. First, implementers of MARTE (e.g., tool vendors) have been invited to provide their feedback on the standard by logging issues where appropriate using the OMG Issue Procedure (defined in the specification itself). (The deadline for sending issues to be resolved by this team was December 22nd, 2007.) After this date, the FTF is not obliged to respond to any additional issues raised, although it has the discretion to do so. Second, the FTF must respond to each of the issues raise, including recommending changes to the published Beta specification. Once these changes have been accepted by the OMG membership, the new version becomes the official 1.0 version.

## REFERENCES

OMGa, UML Superstructure, v2.1.1, formal/07-02-05.

OMGb, UML Profile for Schedulability, Performance, and Time, v1.1, formal/05-01-02.

OMGc, UML Profile for Modeling QoS and FT Characteristics and Mechanisms, v1.0, formal/06-05-02.

OMGd, UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) RFP, realtime/05-02-06.

OMGe, UML Profile for MARTE, Beta 1, ptc/07-08-04.

P. Cuenot, D. Chen, S. Gérard, H. Lönn, M.-O. Reiser, D. Servat, R. T. Kolagari, M. Törngren, M. Weber, Towards Improving Dependability of Automotive Systems by Using the EAST-ADL Architecture Description Language, In the book Architecturing Dependable Systems IV, Edited by R. d. Lemos, C. Gacek and A. Romanovsky, Berlin Heidelberg, Springer-Verlag, LNCS 4615, p. 39-65, 2007.

F. Terrier and S. Gérard, "MDE benefits for distributed, real time and embedded systems", In book "From Model-Driven Design to Ressource Management for Distributed Embedded Systems" edited by L. Kleinjobann, R. J. Machado, C. Pereira, and P.S. Tbiagarajan, Springer, ISBN 978-0-387-39361-2, 2006.

F. Thomas, S. Gérard, J. Delatour and Francois Terrier, Software Real-Time Resource Modeling, In Proceedings of the International Conference Forum on Specification and Design Languages (FDL) 2007, Barcelona, Spain, September 2007.

S. Taha, A. Radermacher, S. Gerard, and J.-L. Dekeyzer, An Open Framework for Hardware Detailed Modeling, In IEEE Proceedings of SIES'2007, ISBN 1-4244-0840-7, Lisbon, Portugal, July 2007.

S. Taha, A. Radermacher, S.Gerard, and J.-L. Dekeyzer, MARTE: UML-based Hardware Design from Modeling to Simulation, In Proceedings of the International Conference Forum on Specification and Design Languages (FDL) 2007, Barcelona, Spain, September 2007.