A General Structure for the Analysis Framework of the UML MARTE Profile

Huáscar Espinoza[a1], Hubert Dubois[a], Julio Medina[b], and Sébastien Gérard[a]

[a] CEA Saclay, DRT/LIST/DTSI/SOL/L-LSP,
F-91191, Gif sur Yvette Cedex, France
{huascar.espinoza, hubert.dubois, sebastien.gerard}@cea.fr
[b] Universidad de Cantabria, Departamento de Electrónica y Computadores,
Av. Los Castros s/n, 39005 Santander, Spain
{medinajl}@unican.es

Abstract. The ongoing underlying work aims to provide a robust and straightforward basis to the UML profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) issued by the OMG. Particularly, in this paper, we analyze the existing annotating mechanisms of extra-functional properties and some specific requirements of the concerned OMG's Request For Proposal (RFP) to consistently derive a preliminary framework for the Analysis sub-profile. Our proposal provides a flexible mechanism to easily increase and suppress QoS attributes without changing the associated Domain Model and Profile, which covers inclusion of modeling capability for new analysis techniques. Furthermore, we allow the unification of the existing Schedulability and Performance modeling sub-profiles in the pertinent aspects, letting them separated in the specialized ones. At the same time, we attempt to provide a generic framework able to be applied to all the UML Profile for MARTE.

1 Introduction

The Real-Time Embedded Systems (RTES) domain has claimed for years a specialized profile of the Unified Modeling Language (UML™) to integrate temporal Validation and Verification (V&V) within the global RTES lifecycle in a fast and standardized way. In spite of the fact that the Object Management Group (OMG) has adopted the UML Profile for Schedulability, Performance, and Time (SPT) [11] to model real-time concerns, a lot of lacks regarding its flexibility have limited a broad adoption by the RTES community. The need for major modifications [12], [3] and the unavoidable evolution to be in compliance with other relevant OMG standards (e.g., UML 2) arose in a Request For Proposals (RFP) for a new UML Profile named MARTE (Modeling and Analysis of Real-Time and Embedded systems) [13].

The UML MARTE profile extends the former scope by comprising:

- modeling of both software and hardware aspects,

[1] Supported by the Programme AlBan, the European Union Programme of High Level Scholarships for Latin America, scholarship No.(E04D028544BO)

- modeling of platform, platform-independent, and their allocation viewpoints in a Model Driven Architecture (MDA™) style approach [16],
- compliance with the UML Profile for Modeling Quality of Service and Fault Tolerance (QoS & FT) [14],
- specification of not only real-time constraints but also other embedded QoS characteristics as power consumption and memory size,
- modeling of embedded, reactive, control/command, and intensive data flow computational systems,
- component-based architectures modeling and analysis,
- capability to Asynchronous/Causal, Synchronous/Clocked and Real/Continuous time modeling.

In particular, the underlying work is part of a major effort performed by the CEA-List in the context of a project called PROTES, conceived within the CARROLL[2] French Research program. The core goal of the involved contributors is to promote the standardization of the UML MARTE profile at the OMG.

Recently, at the CEA-List, some work has been done around the Accord|$_{UML}$ project [2], [6], [4] to connect UML modeling of real-time embedded systems with schedulability analysis tools. A first approach was defined in [19], [20] where the authors have presented a schedulability analysis model which is semi-automatically derived from a conception model, and is then analyzed by a symbolic execution tool [8]. At this point, after a first study about the integration of performance analysis within Accord|$_{UML}$ [24], the intention is to ensure full compliance with the upcoming UML MARTE profile.

Besides, in the MAST[3] project some other efforts have been done to define and build UML conceptual models for the timing properties of object-oriented distributed systems [9]. These models align quite well with the SPT profile concepts, and allow practitioners to feed the state-of-the-art schedulability analysis, design and simulation tools included in MAST [5]. MAST stands for Modeling and Analysis Suit for Real-Time Applications, and its main goal is to bring an open environment for modelers, analysts and practitioners, integrating modeling, analysis and design tools to validate real-time systems, using modern schedulability analysis techniques. During this effort a number of desirable improvements to the SPT profile have been found [10], most of which are addressed in this work.

In this way, the motivation for this paper is to report our current progress in the construction of a more robust structure for the Schedulability and Performance Analysis sub-profiles defined in the current SPT version, by merging them in a unique framework useful for whatever analysis technique, by reorganizing it into generic and consistent modeling concerns, and by providing a flexible mechanism for annotating QoS properties. Since a trustworthy profile is targeted, we put special effort on bridging the gap among different available works, add some missing links, and fuse them into a coherent whole.

The paper is organized as follows. In section 2, we describe the UML profiles for SPT and QoS & FT, and their annotation styles of extra-functional properties. Section

[2] See CARROLL Web Site: *http://www.carroll-research.org/*
[3] See the MAST Web Site: *http://mast.unican.es/*

3 organizes the concerned RFP requirements and other desired key features of the UML profile for MARTE. Next, a preliminary proposal of the analysis sub-profile structure, as well as possible solutions to the concerned requirements, are depicted in section 4. Finally, conclusions are presented in section 5.

2 The UML Profiles for Modeling Extra-functional Properties

In the context of model driven software development approaches, modeling of extra-functional properties (i.e. QoS) plays an essential role to design and analyze real-time systems. Since 1997, Unified Modeling Language (UML) [17] has become the most used specification language for software systems. However, UML itself weakly supports modeling of QoS properties. The main rationale for this lack is that UML intends to become a generic specification language and particular concerns could limit its widespread use. Therefore, a lightweight extension mechanism, the *Profile*, is used to extend UML semantics for particular domains. Thus, two specific UML profiles currently support QoS modeling. The first one, the UML Profile for SPT [11], is specifically customized for the real-time systems domain. The second one, the UML Profile for QoS & FT [14], has a broader scope that includes all kinds of QoS properties.

In this section, we analyze the general structure and goal of both profiles. This will give us a foundation for the next sections in this paper.

2.1 The UML Profile for SPT

One of the key concerns the UML profile for SPT supports is the modeling of QoS characteristics in order to allow quantitative analysis of real-time systems models. Quantitative analysis techniques are used to early verify some extra-functional properties (e.g., response times, utilization, queue sizes) with basis in other available extra-functional properties (e.g., worst case execution times -WCET-, deadlines). The analysis techniques supported by this profile belong to the following two categories: *Schedulability Analysis* [22] and *Performance Analysis*. The first one uses mathematical means (e.g., RMA-based techniques) to predict whether a set of software tasks meets all its timing constraints. Then, they are oriented to verify temporal correctness. The second one generally uses statistical techniques (e.g., queuing theory, Petri Nets, etc.) to calculate response times, delays, and resources requirements in order to determine the rate at which a system can perform a function.

The SPT profile organizes its framework in various sub-profiles (Fig. 1.a). The core package, named *General Resource Modeling* (GRM) *Framework*, is dedicated to specify modeling extensions for generic concepts and it provides a common base for the analysis sub-profiles. This core package consists of three sub-packages: *Resource Modeling*, *Time Modeling* and *Concurrency Modeling.* They respectively introduce a general framework for modeling resources and their QoS, describe the UML extensions to represent time-related concerns and introduce concurrency facilities. Despite the fact that there currently exist only two analysis sub-profiles, for Schedulability and Performance, it was initially planned that future ones dealing with other types of QoS (memory size, power consumption, etc.) could be adopted.

The UML Profile for SPT considers QoS information as the physical properties of hardware and software resources represented in models. This framework does not define concrete types of resources, but focuses on the notion of an *abstract resource* concept (see Fig. 1.b). Then, resources are modeled as servers with *offered QoS*, such as capacity, availability, performance and timing. Users of resources are called *clients* and have *required QoS*, such as deadlines. The relationship between clients and resources is called a *QoS contract*. Thus, it may be possible to determine analytically whether the offered services can satisfy the QoS required by the resource clients [21].

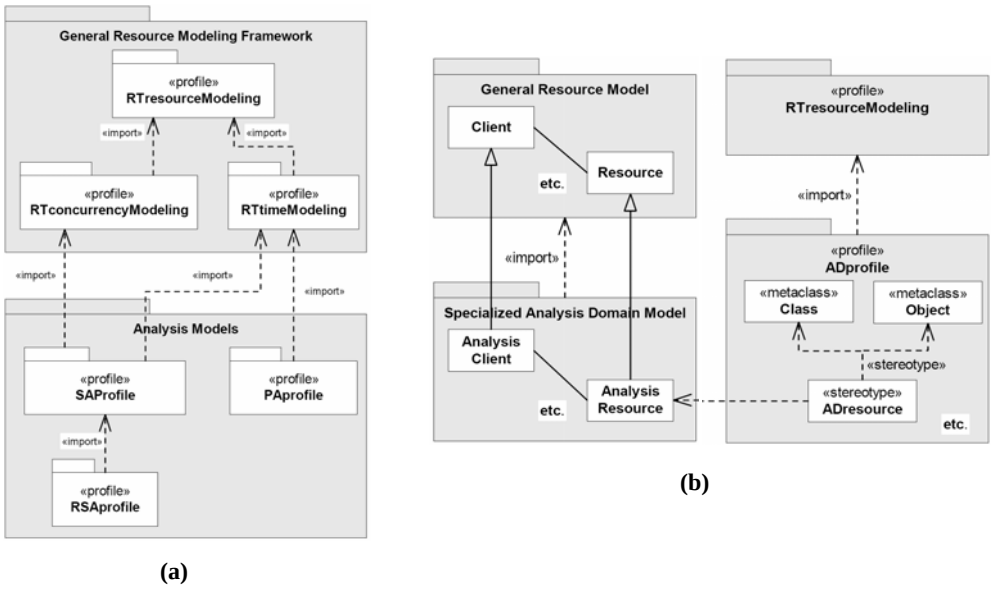

Fig. 1. General Structure of the SPT profile: (a) A view of the SPT packages, (b) A view of the SPT domain model and profile (Source: [11]).

Although this profile represents the notion of a QoS characteristic as an explicit concept, in most cases specific QoS characteristics are represented as attributes of other concepts. For example, an *execution engine* stereotype has QoS properties as *processing rate* and *utilization*. These characteristics are defined as attributes of execution engine, rather than as a subclass of the QoS characteristic concept that is associated with the execution engine concept. This approach was adopted in order to have a simpler domain model. Thus, the SPT annotation method consists in appropriately stereotyping the model elements that have to be characterized by quantitative properties and assigning values to the related attributes using the comment-based notation [11].

To illustrate the use of this profile, we depict a simplified version of a *Speed Regulator* application example. In Fig. 2, we show the class model of our system where two concurrent active classes were defined: *RegulatorManager* and *SpeedGatherer*. In the Accord$|_{UML}$ methodology, they are stereotyped as *RealTimeObject*. The purpose of this system is to maintain the speed of a car to a value (*targetSpeed*) selected by the driver. Both active entities have accordingly two main operations (*regulateSpeed* and *updateSpeed*) which need to access to the same shared resource: *SpeedData*.

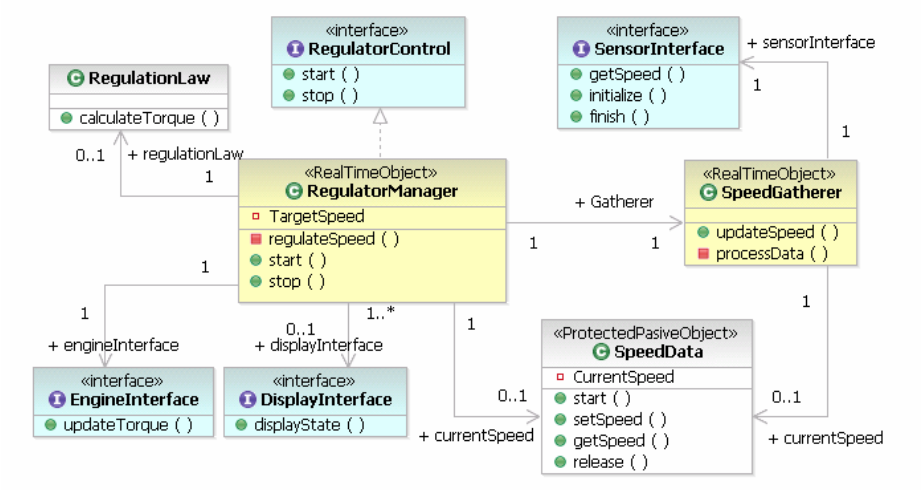

Fig. 2. Structural view for the Speed Regulator example

In Fig. 3, we annotate, as example, the UML sequence diagram of the *updateSpeed* operation with Schedulability stereotypes and QoS values. In UML schedulability modeling, this scenario represents a *scheduling job*, which is associated with a starting *trigger* and a *response* composed by a sequence of schedulable *actions*.

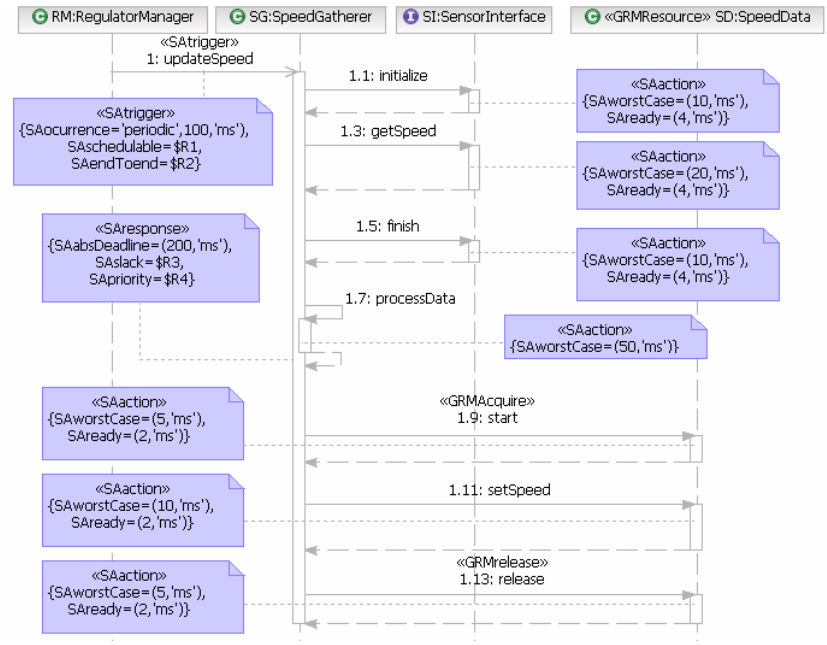

Fig. 3. Sequence Diagram for *updateSpeed* annotated with the UML profile for SPT

The event that triggers this scheduling job is stereotyped *SAtrigger* and a set of QoS properties are associated to characterize it. For instance, its occurrence pattern is de-

fined and quantified, as well as some non-quantified parameters are specified by using variables ($R1…$R4). This means in SPT that their values will be returned to the model by the schedulability analysis tools. The response (*SAresponse*) is characterized by a set of actions stereotyped by *SAaction*, and quantified by a *SAdeadline* and other schedulability parameters (*SAslack* and *SApriority*). Each atomic action (stereotyped *SAaction*) is annotated with latency metrics such as WCET and ready time.

2.2 The UML Profile for QoS & FT

The UML profile for QoS & FT has a broader scope than the UML profile for SPT one. It aims to allow the user to define a wider variety of QoS requirements and properties (performance, fault tolerance). The framework of this UML profile supports a general categorization of different kinds of QoS; including QoS that are fixed at design time as well as ones that are managed dynamically. Furthermore, it supports the integration of different categories of QoS in order to model QoS of system aspects. This QoS framework provides a metamodel to define the domain concepts and to construct the actual QoS profile and a repository of QoS specifications (named *QoS Catalog*).

A set of fundamental concepts are defined in this QoS profile's metamodel (see Fig. 4). Thus, a *QoS Characteristic* is a quantifiable service feature of computational systems (e.g., latency, throughput, availability). There exist usually general characteristics, but particular domains may offer additional specific QoS characteristics. A QoS Characteristic may have parameters (*QoS Parameters*). For example, the QoS characteristics provided by the profile have a *unit* parameter that needs to setup when used.

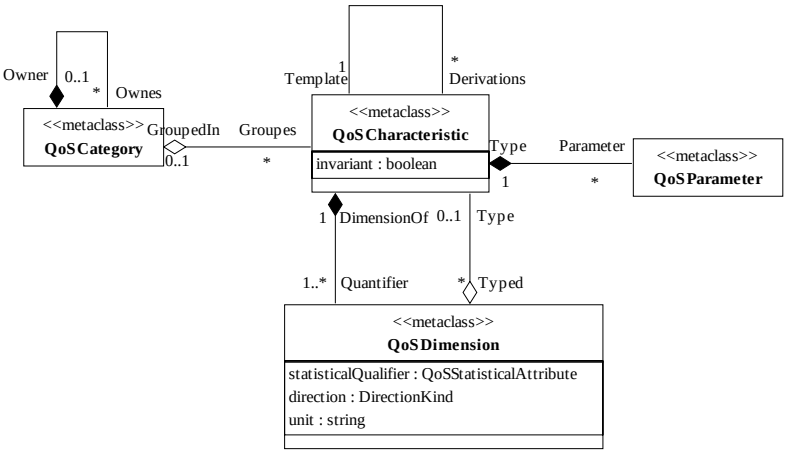

Fig. 4. The QoS Characteristics core model of the UML profile for QoS & FT (Source: [14])

In order to specify the different metrics of each characteristic, the *QoS Dimension* concept was adopted. QoS Characteristics could require more than one type of metric

(e.g., worst case, best case, minimum). Hence, a QoS Dimension is characterized by its *statistical qualifier* (mean, average, variance, etc.), its *direction* (increasing or decreasing values, in order to define what quality values are better), and its *unit* (seconds, Kbps, etc.). In addition, by means of *QoS Categories* it is possible to group QoS Characteristics.

Other important concepts are: *QoS Contexts* which are applied to describe quality expressions when it includes multiple QoS Characteristics and model elements; *QoS Values* and *QoS Dimension Slots*, i.e., the instances of QoS Characteristics and QoS Dimensions respectively; *QoS Constraints*, to limit the allowed values of QoS Characteristics by means of *QoS Required*, *QoS Offered*, and *QoS Contract* constraints.

The use of the QoS & FT-annotation mechanism consists in three phases: Initially, a *QoS Catalog* with the most common *QoS Characteristics* and associated *QoS Dimensions* must be defined by each domain (e.g., the RTES domain). The QoS Catalog can be basically considered as a UML library of QoS properties. Second, a *Quality Model* has to be derived from the QoS Catalog for each modeled application by resolving all the parameters of the *QoS Characteristics* template classes. At the end, the UML user-models are annotated with *QoS Constraints* (*QoS Offered*, *QoS Required* or *QoS Contract*) and *QoS Values*, according to the *Quality Model* defined in Step 2.

These three phases are illustrated in Fig. 5, as well as the SPT's annotation mechanism in order to compare both approaches. We can clearly see that the SPT modeling approach is more direct comparing to the QoS & FT one.

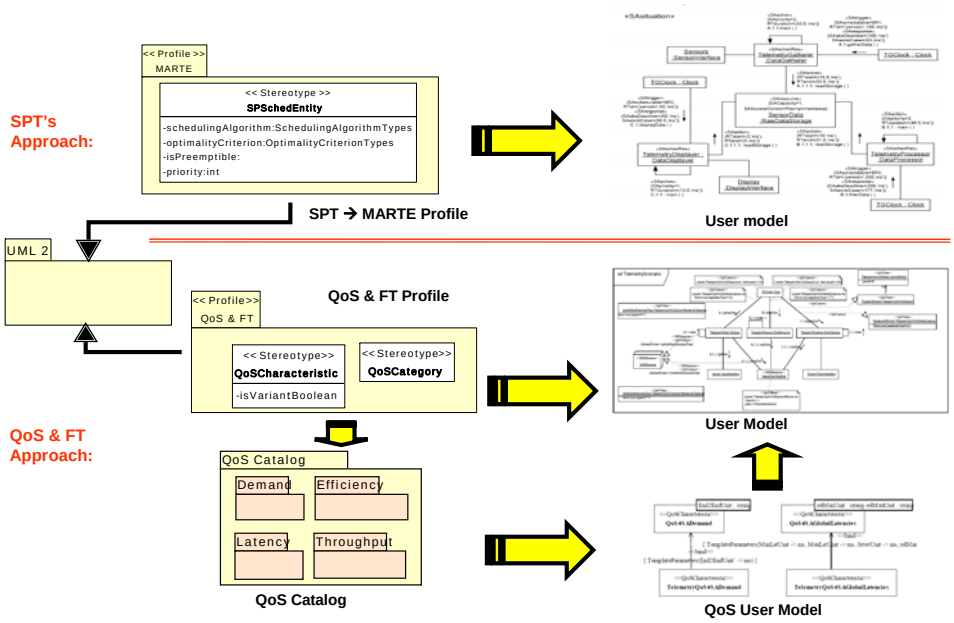

Fig. 5. Comparative scheme between the SPT and QoS & FT modeling methods

As an example, we illustrate the use of the QoS & FT profile for the Speed Regulator system. For that, we base on the QoS Catalog for Schedulability Analysis defined in

the UML profile for QoS & FT [14]. In order to define the *unit* parameters of this QoS Catalog, a QoS model for the Speed Regulator system is constructed (see Fig. 6).

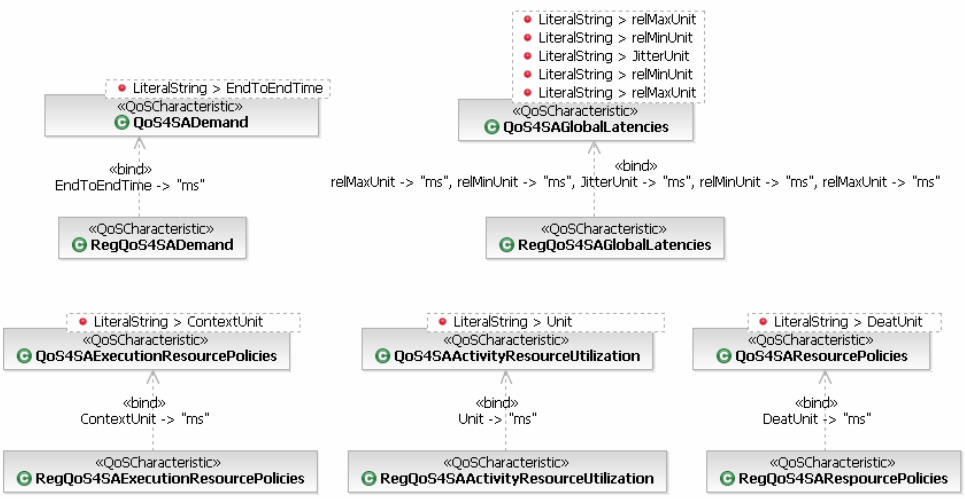

Fig. 6. The QoS Model for the Speed Regulator example

Once this QoS Model is defined, extra-functional properties are added into the specific user models for the Speed Regulator system. In Fig. 7, we depict the scenario for the *updateSpeed* operation annotated with the QoS & FT profile.

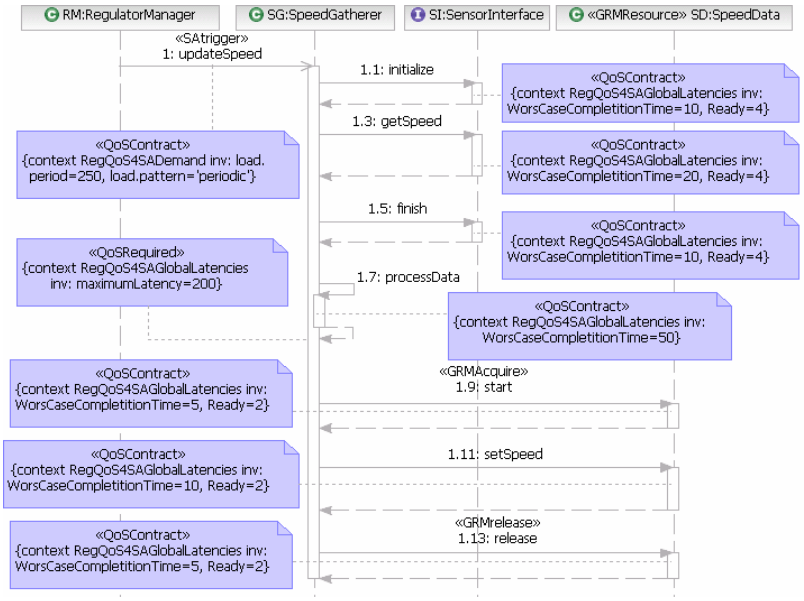

Fig. 7. Sequence Diagram for *updateSpeed* annotated with the UML profile for QoS & FT

Only one of the three possible annotating methods defined by the QoS & FT profile is used. Indeed, we use *constraint symbols* attached to the constrained model element and stereotyped by the corresponding QoS constraint type (*QoS Required* or *QoS Contract* here). These constraints include OCL expression whose context is a *QoS Characteristic* class from the QoS Model. The expression in OCL defines the allowed values of the *QoS Dimensions* associated to a *QoS Characteristic*.

Note that this annotation just covers QoS properties, but it does not define specific entities for analysis as for example triggers or responses like in the SPT profile. The unique generic entity that adopts this profile is *GRM Resource*. Therefore, the QoS & FT profile does not fully support analysis modeling since it does not enable tools to extract basic analysis concepts as tasks, triggers, etc. That could be possible with special considerations and restrictions in the user models, but the QoS & FT profile is clearly insufficient to make an easy model extraction for the analysis tools.

In summary, we have seen that the SPT profile's modeling method and annotation style are really simple for users. But its structure is not enough flexible to increase new QoS properties or different analysis techniques. Conversely, the QoS profile's annotation style is more complicated for users. But its structure is more flexible than this one proposed by the SPT profile because of the library style for defining QoS properties, OCL constraints to describe complex QoS functions, and the useful qualifiers to QoS properties [1]. In our work, we intended to provide a flexible and straightforward framework for MARTE while adopting best practices of both profiles.

3 UML MARTE Requirements Concerning the Proposal

The concepts of the UML profiles for SPT and QoS & FT described in section 2 are essential to well understand the proposal presented in section 4. However, a synopsis of the concerned requirements for the upcoming UML profile for MARTE will complete a comprehensible overview of the main rationales that support our proposal.

The MARTE RFP implicitly defines a global framework for the profile. This structuring model results from the organization of its mandatory requirements into three sub-profiles (Fig. 8).

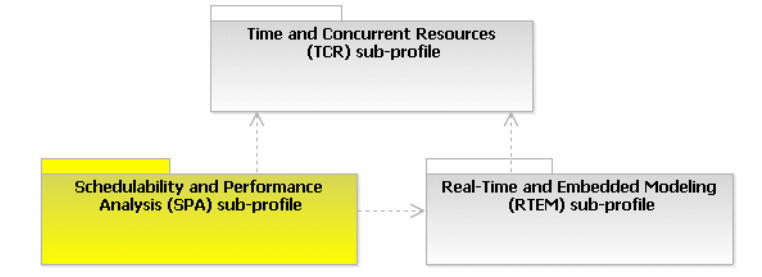

Fig. 8. Global Framework for the UML MARTE Profile

The *Time and Concurrent Resources* (TCR) sub-profile holds the basic modeling constructs related to Time, Concurrency and General Resources from chapters 3 to 5 of

the SPT profile. The Schedulability and Performance Analysis (SPA) sub-profile consists on the concepts that evolve from those corresponding to SPT's chapters 6 to 9. Finally, the Real-Time Embedded Modeling (RTEM) sub-profile holds a number of new requirements for modeling several categories of embedded systems. This includes those that have hardware/software coexistence at the system level, and in general those that may be characterized by any of the embedded systems QoS properties like memory size and power consumption.

The work presented in this paper addresses in particular the general structure and main modeling mechanism for the SPA sub-profile. Among the requirements that appear in the MARTE RFP [13], and which are addressed in our proposal, we mention:

- Compliance with the UML profile for QoS & FT.
- Harmonization between previous schedulability and performance sub-profiles.
- Automation for tools following already available standards and using OMG modeling languages (i.e., UML 2.0, MDA, XMI).
- Factor out as much as possible concepts that could be used in different contexts and specify them separately.
- A rich set of measures for SPA must be provided. Measures for these quantities must include statistical measures (such as average, variance, percentile values, histograms, etc.). Any measure must be expressible in multiple versions such as a required value, an offered value, an estimated value, a test result value, etc.
- Measures should be applicable to any interval bounded by defined start and end events, and be applicable to any resource.
- It is expected to support parameterized expressions for QoS annotations. The SPT profile defined the *Tag Value Language* (TVL), a specialized language that supports symbolic variables and expressions. This could be useful to describe multiple cases with alternative values and results, and to support a more structured analysis.

The structure for the analysis sub-profile proposed in this paper help in one hand to respond to the requirements of the MARTE RFP mentioned above. But, in the other hand, it brings some other enhancing features for the profile:

- To provide a flexible framework for extra-functional annotations. That means to adopt a mechanism to easily increase or suppress QoS attributes without changing the domain model and the underlying profile. That must cover inclusion of modeling capability for future analysis techniques that would use MARTE.
- To provide a simple annotation mechanism to make easy the construction and comprehension of user models. The current UML QoS & FT Profile's annotation style seems too much complex and then not well accepted by final users.
- Appropriate unification of the schedulability and performance sub-profiles in the pertinent aspects, letting them separated in the specialized aspects.
- To help to reduce potential incoherencies between the GRM framework and the specialized sub-profiles. For instance, repetition of modeling entities: *processor* for GRM versus *executionEngine* for SPA; or QoS properties as *entities* in GRM versus QoS properties as *attributes* of stereotypes in SPA and RTEM.
- To provide a generic approach for quantitative annotations able to be applied to all the MARTE profile. That means an approach not just for analysis concerns, but also for QoS annotations of power consumption and memory size.

4 A Proposal for Structuring the Analysis Sub-profile

In this section, we describe a proposal for structuring the *Analysis Sub-profile* intended to meet the major requirements stated in section 3. We organize this description by initially presenting a global overview of the concerning sub-profile and its core features. Next, a close-up view of each constituent package is expounded in order to better understand the key rationales behind the proposal. Finally, a very simple but illustrative example of *modeling for analysis* is depicted.

Since a general framework is proposed, we abstract away exhaustive definitions of *modeling entities* (i.e., its nomenclature and relationships) and their detailed *QoS metrics* (i.e., a full QoS Catalog for MARTE). Also, a proof of concept based on complete modeling and analysis examples is left out, since our main intention is to discuss a preliminary skeleton to support a further more in-depth work.

4.1 Overview of the Analysis Framework for UML MARTE

Figure 9 shows the relationships among the different models of the proposed structure for the UML MARTE profile. Note that despite of the particular focus on the Analysis Sub-profile, the proposed architecture has a large impact on the global MARTE structure. In fact, an essential requisite to keep a homogeneous approach throughout the overall MARTE organization is to adopt the proposed framework for the concerned RTEM QoS models also.

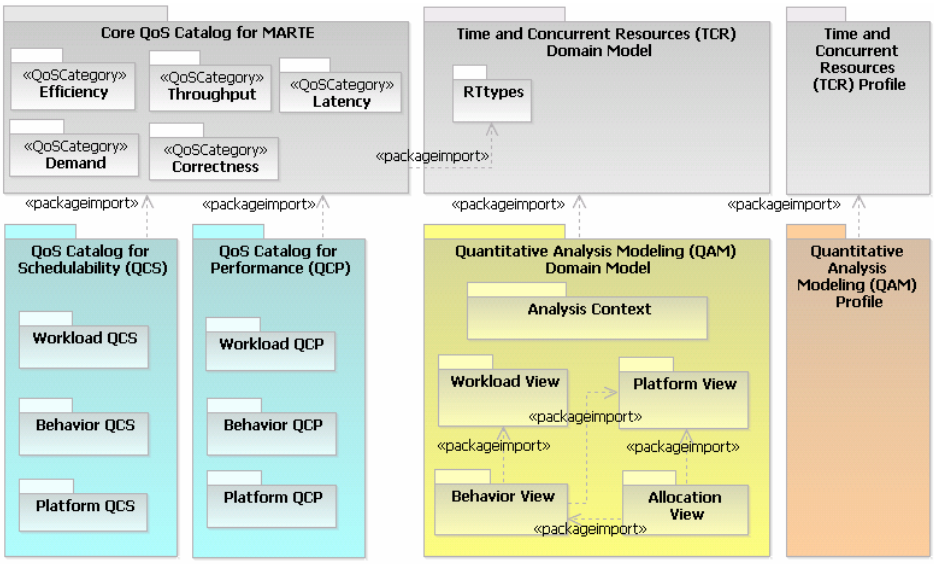

Fig. 9. Global Overview of the Analysis Framework for UML MARTE

The assumed approach involves the adoption of both the extra-functional annotation mechanism and the library style of QoS Catalogs defined by the UML profile for QoS

& FT. However, some restrictions to reduce the QoS profile's inherent complexity and to facilitate the modeling process are defined. Additionally, some useful mechanisms provided by the UML profile for SPT are adopted. In this manner, we intended to provide a flexible and straightforward framework for supporting a wide variety of analysis techniques while adopting best practices of both UML profiles.

Hence, the structure depicted in Fig. 9 highlights six core packages. The three packages located on the top of the figure correspond to the TCR framework and the ones bottom deal with analysis concerns. Horizontally ordered, the packages at the left side collect the different class models of the *QoS Catalog for MARTE*. At the middle and right sides the *Domain Models* and the actual normative *UML profile* are shown.

This global overview allows us to identify the following key features:

- At the core, a generic framework for whatever analysis technique is defined. We call it *Quantitative Analysis Modeling (QAM)*. Its intent is to define basic modeling concepts for different analysis techniques. It will also contain *generics* QoS properties for these basic concepts. We will see that these generic QoS properties map to particular *QoS Characteristics* of the QoS Catalog for MARTE.
- The QAM domain model specifies a set of common concerns useful for defining model-based analysis. These modeling concerns are articulated by the concept of *Analysis Context*. Particularly, we have identified four generic modeling concerns: *workload, behavior, allocation,* and *platform*, each one defined in separated packages.
- The specific metrics for each analysis technique are defined in *specialized QoS Catalogs*. These metrics correspond to the various *QoS Dimensions* assigned to the QoS characteristics. We have defined two basic specialized QoS catalogs: for schedulability analysis (QCS) and for performance analysis (QCP). Other types of analyses are expected to be defined, as for example for WCET analysis.
- Since various QoS Characteristics are common for the different analyses, we have factorized them in a *Core QoS Catalog*. This core library allows us to organize the base QoS Characteristics in *QoS Categories*.
- Finally, the specialized QoS Catalogs (e.g., QCS, QCP) organize the more specific QoS Characteristics in modeling concern packages (e.g., *Workload QCS, Behavior QCS, Platform QCS*). Note that these packages match (when the QoS annotations apply) to the modeling concerns defined in the QAM domain model.

Next, the proposed framework and its rationales are explained in detail.

4.2 The Quantitative Analysis Modeling (QAM) Domain Model

We have organized the QAM domain model around the concept of *Analysis Context*. As defined in [11], an analysis context represents a specific scenario of system operation with a particular configuration of computational resources providing particular quantitative information (e.g., schedulability) for analysis tools. Indeed, it provides a starting point for extracting the information that analysis tools need.

In our approach, an analysis context is described by separated models associated to a set of generic modeling concerns. This separation of modeling concerns permits us

to organize the QAM domain models into comprehensible parts, and also to highlight the commonality between the modeling aspects of different analysis techniques. The choice of this particular set of modeling concerns is based on the conceptual composition for analysis context defined in the GRM framework defined in [11][4]. Nevertheless, we have redefined some key notions. In the SPT's GRM framework, an analysis context is composed of a set of modeling instances [7]. Thus, a *usage demand* represents the load on the system; a *resources usage* describes how a set of clients uses resources in *static* and *dynamic* fashions; and the *resources instances* define the specific used resources and their services.

Figure 10 shows the *Analysis Context* model of the QAM package. In our case, the components of an analysis context are not instances but *modeling views* (i.e., a configuration of modeling elements with a meaning for each particular analysis technique). We argue that the concept of analysis context has meaning only whether we represent it by a specific *configuration of modeling elements* but not by a single *set of modeling elements* as defined in the GRM framework.

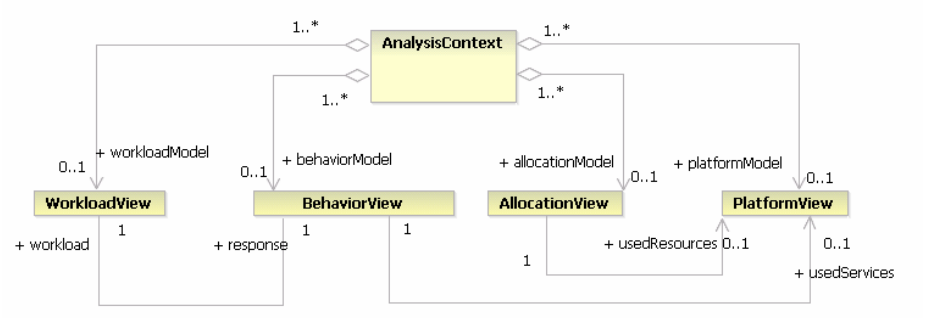

Fig. 10. The Analysis Context model of the QAM Framework

Regardless of the different nomenclature that has been adopted, there exists a sort of analogy between the proposed concerns and the GRM concepts of: usage demand, dynamic usage, static usage, and instance resource. Therefore, in the QAM domain model, a single analysis context is characterized by:

- *Workload View*: a stable workload of jobs executed in response to external (e.g. the environment) or internal (e.g. a timer) stimulus,
- *Behavior View*: a description of the executed actions chain as response to the workload, including the access to shared resources and their services,
- *Allocation View*: a configuration of the deployment among different logical elements (e.g. actions) and engineering ones (e.g. threads, processors) [21] and,
- *Platform View*: a specific architecture of the computational resources.

[4] The root domain model for our Analysis Context framework is *the resource usage framework* of the General Resource Model (GRM) described at page 3-8 in the SPT document.

Note that each identified concern has a *domain sub-model* which collects specific sets of *modeling entities* and, at the same time, is a *modeling entity* itself which is susceptible to be stereotyped. Subsequently, we could use these "views" stereotypes to structure user models, allowing organization of models not only at the level of the QAM framework but also at the user model level. Thus, we could arrange user models into separated *analysis views* stereotyped with the four generic modeling concerns. We will return on this issue in Section 4.3.

In order to depict our approach, the *Workload View* domain model is shown in Figure 11. This model describes the concepts required to specify the load on the system and the associated end-to-end responses for a set of particular *transactions*. By transaction (a concept introduced in [23] and [18]), we mean a sequence of actions triggered by an event occurrence with an invariant arrival pattern: not only periodic patterns but also non-periodic ones with a stable set of QoS values. It replaces and merges the SPT concepts of *Scheduling Job* (from the Schedulability sub-profile) and *Scenario* (from the Schedulability sub-profile).

A workload view could correspond to a mode of system operation (e.g. starting, fault recovering, normal operation, etc.) or a level of intensity (i.e. arrival patterns) of environment events.

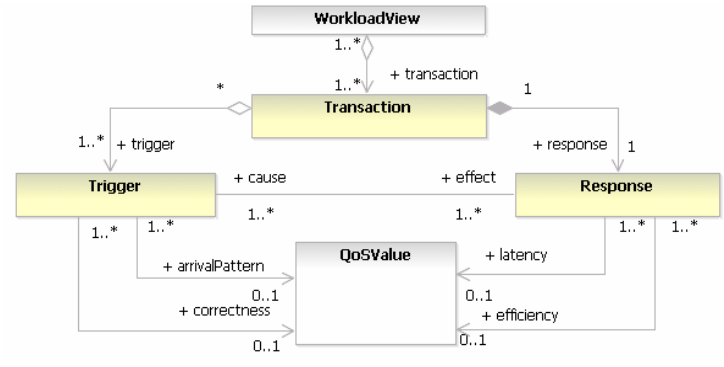

Fig. 11. The Workload View model of the QAM Framework

The modeling entities (e.g., trigger, response) could be imported from the TCR domain model if they are common for all the UML MARTE profile, or alternatively they could be exclusively defined in the QAM domain model whether they are specific.

Independently of the conceptual differences with the related SPT models (which are not the scope of this paper), we can notice in this model a fundamental difference regarding to the description of QoS properties. Indeed, we do not use the approach to represent QoS properties as *entity attributes* like in the SPT's schedulability and performance sub-profiles. Actually, we keep the GRM style of representing QoS properties as *separated entities* by means of the QoS Characteristics instances, named *QoS Values*. Thus, full consistency among the different domain models is provided, including those ones defined in the UML profile for QoS & FT.

In addition, each modeling entity has only generic QoS properties (e.g., *correctness*[5], *efficiency*, *latency*). These QoS properties will be concretized just for their specialized analysis techniques by the corresponding QoS Catalog (i.e. QCS, QCP, etc.). For example, we use the *isSchedulable* QoS Dimension of the *correctness* QoS Characteristic for schedulability analysis, but for performance analysis this concrete QoS Dimension is not useful.

In the following sections we give more details on this issue.

4.3 The Quantitative Analysis Modeling (QAM) Profile

After defining a domain model, UML extensions (i.e., stereotypes, tagged values, and constraints) has to be defined. This section depicts the set of proposed UML extensions for the *Workload View* model (Fig. 12). Notice the following assumptions in this partial profile: a) the OCL constraints, to ensure well-formedness models are not considered, and b) just a set of possible extensions to UML metaclasses, biased toward our example, is depicted.

We will use *Interaction Overview Diagrams* (IOD), a specialized form of *Activity Diagrams*, to represent the Workload of an application. Nevertheless, *Workload Views* are extended to UML *activities*, since this one is the actual UML element used to enclose activity diagrams. *Triggers* will be specializations of UML *Control Flows* in order to express the stimulation of *Responses*, which are in turn extensions of *Interaction Use*. Finally, the *Activity Partitions* will be stereotyped as *Transactions*. In section 4.5 we will see an example illustrating the application of this UML sub-profile for MARTE dedicated to workload descriptions.

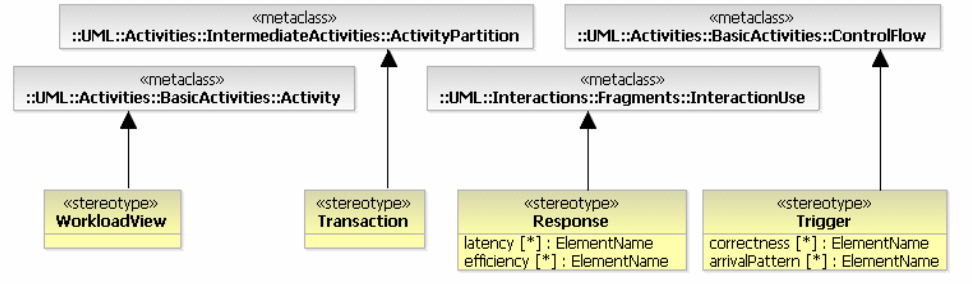

Fig. 12. The Workload Stereotypes of the QAM Framework

The main feature here is the use of attributes in each stereotype to annotate the reference to QoS properties. We can notice that we apply the same mechanism of making reference to UML elements than the one used in the UML profile for QoS & FT ([14] page 27) and in the SysML draft standard ([15] page 126). Also, note that QoS attributes have to be *QoS Characteristic class names*. For the tools, it is important to use

[5] *Correctness* is a new QoS Characteristic we propose to represent: *isSchedulable*, *deadlocks*, and other temporal and logical QoS correctness properties.

here compound element names. An example of a compound element name is the form *'PackageName::ElementName'*. This reference must be defined by means of tagged values associated to the stereotyped elements within the user models.

Additionally, the multiplicity *[*]* attached to attributes allows making reference to different UML elements. For instance, a *Response*-stereotyped element could be noted by efficiency QoS Characteristics of performance and schedulability at the same time, when we are interested in applying both analysis techniques in the same user model.

In both following sections, we will see how we map these QoS attributes to the QoS Characteristic defined in the QoS Catalog for MARTE, and the mechanism to apply it in the user models.

4.4 The QoS Catalogs for Analysis Models

As explained in Section 4.1, QoS Characteristics are organized in QoS Categories within the Core QoS Catalog for MARTE, and in QAM modeling concerns within the specialized QoS Catalogs for Schedulability, Performance, etc. (see Figure 9). The Workload QoS Catalog for Schedulability (Workload QCS) is shown in Figure 13.

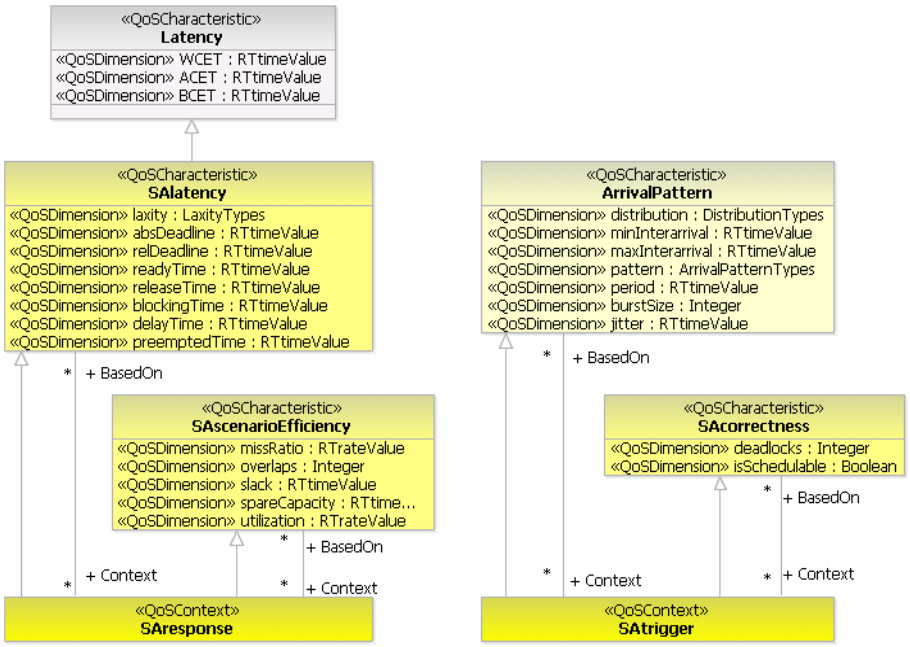

Fig. 13. The Workload QoS Catalog for Schedulability Analysis

Note that the QoS Dimensions of specific QoS Characteristics are collected in *QoS Contexts* by means of the *generalization* association. Also, the QoS profile meta-

model's association between QoS Characteristic and QoS Context is kept with the roles *BasedOn* and *Context*. Each *QoS Context* represents the application context of a set of *QoS Characteristics* mapped to a specific QAM stereotype. This enables to assign, for example, *efficiency* and *latency* QoS Characteristics to a *response* stereotype by means of a unique *response context*, and thus, to reduce the complexity of the QoS annotations, as will be shown in section 4.5.

The following additional features are highlighted in the QoS Catalog's example:

- The QoS Dimensions stereotypes enable to define statistical qualifiers (e.g. min, max, mean, distribution). This is not shown in this paper for simplicity.
- Generic MARTE *Types* will be defined and imported by the QoS Catalogs from the TCR model (Figure 9). As in the SPT standard, a special nomenclature (e.g., *RTtimeValue*) will define normative means to annotate *values*, *units*, and *measures types*.
- Some QoS Characteristics will be shared among the different QoS Catalogs for MARTE (e.g. *ArrivalPattern*), and others will be specialized for each QoS Catalog (e.g. *SAlatency* for Schedulability Analysis inherits from *Latency*).

4.5 A Modeling Example with the Proposed Approach

Figure 14 shapes an Interaction Overview Diagram (IOD) for the Speed Regulator example introduced in section 2.1. This example is developed in the context of the Accord|$_{UML}$ methodology and we show in this paper some of the model elements that are defined in this methodology. This activity diagram represents a *Workload View*, consisting of two transactions characterized by their triggers and responses are shown. Thus, these two *transactions* explicitly introduce the semantic of concurrency for the *Activity Partitions*, and the *triggers* the semantic of periodicity for the execution of each *Interaction* invocation.

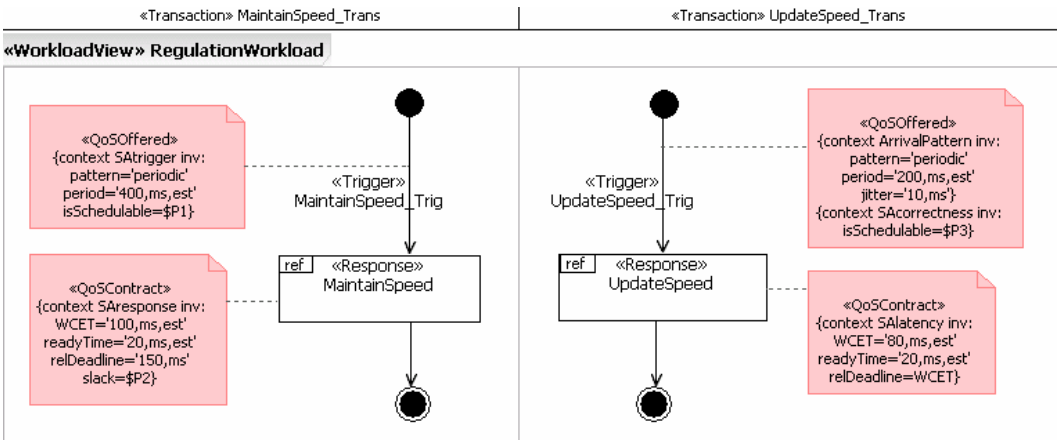

Fig. 14. A modeling example of Workload View for Schedulability Analysis

This modeling method is explained in the UML profile for QoS & FT ([14] page 27, method 1). Accordingly, we annotate UML constraints with the stereotypes *QoSRequired*, *QoSOffered*, or *QoSContract*. These constraints include OCL expressions whose context is a *QoS Context* or *QoS Characteristics* classifier. The OCL expressions define the allowed values of *QoS Characteristics* associated to the *QoS Context*. Moreover, we could alternatively use *QoS Characteristic* (like in the OCL constraints defined for the *Maintain Speed* transaction in Fig. 14) or *QoS Context* (like in the OCL constraints for the *Update Speed* transaction in Fig. 14) for the context of OCL constraints.

The *Trigger*'s tagged values (*arrivalPattern* and *correctness*) are not shown in Figure 14 because of it is redundant for the users. However, they must be specified with the associated QoS Characteristic class names in the tagged values table for the stereotype *Trigger*. That will be important for tools which will extract the analysis information.

A notation (like in SPT) is adopted for special types. In this notation we define possible *units* of measures and qualifiers of values: 'estimated', 'tested', etc. The use of these units (e.g. 'ms', 'sec', etc.) avoids the construction of the *QoS User Model*, and so it reduces the complexity of the modeling process.

5 Conclusions

In this paper, we described an approach for organizing the upcoming UML MARTE profile in a more comprehensive framework than the former SPT one. We focus in the *Analysis* sub-profile, but the QoS modeling framework and annotation style can also be reused in the Real-Time and Embedded Modeling (RTEM) sub-profile.

The presented approach is based on the following criteria:

- To adopt the major QoS & FT profile's features; since its flexibility, and broad possibilities to characterize QoS metrics supply foremost of the desired and RFP MARTE's requirements. We mean *flexibility* the fact we could define QoS Catalogs as modifiable, easy to improve and increase libraries of QoS properties. Users could easily define their own QoS properties. On the other hand, the QoS & FT profile provides a set of metric qualifiers defined at the metamodel level (an advantage for tools based on metamodels), as for example: *QoS Offered*, *QoS Required*, *QoS Dimension*'s statistical qualifiers, etc. In SPT that is partially defined at the level of special notations as *RTtimeValue*.
- But the QoS & FT profile is insufficient for analysis annotations and has a too-complex modeling method. It is insufficient because it is aimed only for QoS properties annotations and does not define a set of modeling entities (schedulers, processing engines, real-time responses, etc.) useful to organize analysis models. Then, we propose a set of entities for analysis (*QAM model*) organized in *modeling concerns*, with generic QoS properties.
- The QoS & FT profile's annotation mechanism is particularly complex due to it requires constructing a QoS model for each modeled application, and gives users too much ways to annotate QoS properties. This profile defines three different methods to annotate QoS attributes. That could be confusing for users and hard to

implement for different tools which in some cases must share models. We restrict the annotation style to the first method by means of OCL constraints, which at the same time enables to define parameters, functions, and reference to other QoS properties. Also, we remove the step of defining a *QoS User Model* by defining the *units* of QoS metrics as was defined by the SPT profile, i.e., with *RTtimeValue* notation conventions.

- We propose to organize the domain model in modeling concerns which are relevant for the different analysis techniques. This separation provides users a more comprehensible framework than the SPT domain models for schedulability and performance, since they could easy identify the different relevant aspects for the analysis tools.
- We propose an *Analysis Framework* for whatever *Quantitative Analysis Technique*, while providing flexibility to introduce new techniques and new QoS properties without changing the base framework.

Nevertheless, more research must still be done to validate this approach. This work can be seen as a first reflection to enhance the flexibility of the upcoming UML MARTE profile in order to support a more agile design process of RTES.

References

1. S. Bernardi, D. Petriu: Comparing two UML Profiles for Non-functional Requirement Annotations: the SPT and QoS Profiles, UML'2004, Lisbon, Portugal, October 2004.
2. CEA, I-Logix, Uppsala, OFFIS, PSA, MECEL, ICOM, "UML based methodology for real time embedded systems," version 1.0, april 2003, Project IST 10069 AIT-WOODDES.
3. S. Gérard (edited by): Report on SIVOES'2004-SPT Workshop on the usage of the UML profile for Scheduling, Performance and Time Mai 25th, 2004, Toronto, Canada.
4. S. Gérard: "Modélisation UML exécutable pour les systèmes embarqués de l'automobile", PhD Thesis. 2000, Evry, Paris.
5. M. González, J. L. Medina, J. Gutiérrez, J. Palencia, J. M. Drake: MAST: An Open Environment for Modeling, Analysis, and Design of Real-Time Systems. CARTS Workshop, Aranjuez, Spain, October 2002.
6. A. Lanusse, S. Gérard, F. Terrier, Real-time Modelling with UML: The ACCORD Approach, In Proceedings of the UML'98, Springer Verlag LNCS 1618.
7. L. Lavagno, G. Martin, and B. Selic, "UML for Real. Design of Embedded Real-Time Systems," Kluwer Academic Publishers, 2003.
8. D. Lugato, C. Bigot, Y. Valot. Validation and automatic test generation on UML models: the AGATHA approach. In Proceedings of the Workshop FMICS, ENTCS 66 n°2, 2002.
9. J.L. Medina, M. González Harbour, and J.M. Drake: "MAST Real-Time View: A Graphic UML Tool for Modeling Object-Oriented Real-Time Systems" Proceedings of the 22nd IEEE Real-Time Systems Symposium (RTSS 2001), London, UK, IEEE Computer Society Press, pp. 245-256, December 2001.
10. J. L. Medina, M. G. Harbour and J. M. Drake: The "UML Profile for Schedulability, Performance and Time" in the Schedulability Analysis and Modeling of Real-Time Distributed Systems. SIVOES-SPT Workshop. Toronto (Canada). May, 2004.
11. Object Management Group: UML Profile for Schedulability, Performance, and Time, Version 1.1. 2005. OMG document: formal/05-01-02.

12. Object Management Group: Pending Issues sent to the OMG Finalization Task Force corresponding to the UML Schedulability, Performance and Time profile http://www.omg.org/issues/uml-scheduling-ftf.open.html.
13. Object Management Group: UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE), RFP. 2005. OMG document: realtime/05-02-06.
14. Object Management Group: UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms. 2004. OMG document ptc/04-09-01.
15. Object Management Group: Systems Modeling Language (SysML) Specification, Version 0.9. Draft. 2005.
16. Object Management Group. MDA Guide Version 1.0.1. 2003.
17. Object Management Group. Unified Modeling Language: Superstructure Version 2.0. 2004. OMG document ptc/04-10-02.
18. J. C. Palencia and M. G. Harbour: Exploiting Precedence Relations in the Schedulability Analysis of Distributed Real-Time Systems: Proceedings of the 20th Real-Time Systems Symposium, IEEE Computer Society Press, pp 328-339, December 1999.
19. T. H. Phan, S. Gérard and D. Lugato. Schedulability Validation for UML-modeled real-time systems with symbolic execution and jitter compensation. ERCT Workshop, 2003.
20. T. H. Phan: "Analyse d'ordonnançabilité d'applications temps réel modélisées en UML", PhD Thesis. 2004, Evry, Paris.
21. B. Selic, A Generic Framework for Modeling Resources with UML. IEEE Computer, Vol.33, N. 6, pp. 64-69. June, 2000.
22. Sha, L., Abdelzaher, T., Arzen, K., E., Cervin, A., Baker, T., Burns, A., Buttazzo, G., Caccamo, M., Lehoczky, J., Mok, A., K.: Real Time Scheduling Theory: A Historical Perspective: Real-Time Systems Journal, Vol. 28, No, 2-3, pp. 101-155, ISSN:0922-6443, November-December 2004.
23. K. Tindell: Adding Time-Offsets to Schedulability Analysis: Technical Report YCS 221, Department of Computer Science, University of York, January 1994.
24. N. Torrecillas, H. Dubois, S. Gérard: Performance Evaluation of Real Time Embedded Systems with the Accord/UML methodology, UKPEW'04, UK, July 2004.
25. C.M.Woodside, D.C. Petriu: Capabilities of the UML Profile for Schedulability Performance and Time (SPT) Workshop SIVOES-SPT RTAS'2004, Toronto, Canada, May 2004.