# Applying MARTE Profile for Optimal Automotive System Specifications and Design

Fabíola Gonçalves C. Ribeiro*, Achim Rettberg†, Carlos E. Pereira‡, Michel S. Soares§
*Federal Institute Goiano, Catalão, Brazil
Email: fabiola.ribeiro@ifgoiano.edu.br
†Carl von Ossietzky Universitt Oldenburg, Oldenburg, Germany
Email: achim.rettberg@iess.org
‡Federal University of Rio Grande do Sul, Porto Alegre, Brazil
Email: cpereira@ece.ufrgs.br
§Federal University of Sergipe, São Cristóvão, Brazil
Email: mics.soares@gmail.com

***Abstract*—The UML profile for Modeling and Analysis of Real-Time and Embedded Systems (MARTE) describes semantics and syntax for designing embedded and real-time systems, providing capabilities for representing the intrinsic characteristics of these systems, such as resource allocation, time criteria, non-functional characteristics, among others. MARTE provides different constructors and appropriate annotations for design activities allowing representation of quantitative characteristics that are relevant to the domain of a real-time system such as, for example, deadlines, periods, processing capacity, timing, and also qualitative characteristics that relate to system performance, including methods of communication and concurrence. This paper presents an in-depth study about the CoreElements, Time and GRM packages of the MARTE profile. In addition, it presents an initial analysis of conformity of MARTE constructors in the context of the specification processes and design of automotive systems. It is important to emphasize that the presented models are strengthened with MARTE constructors,by allowing the representation of functional and non-functional requirements, performance and temporal restrictions in the field of automotive systems already in initial stages of system design.**

## 1. Introduction

Real-time embedded systems (RTES), such as automotive systems, are complex computer systems composed by hardware and software components. Moreover, these systems present functional and non-functional critical requirements related to security, performance, temporal restrictions, trust, among others. Therefore, the design and development of RTES consist of non-trivial processes in which strategies must be elaborated in order to contribute favorably to their development. Given the complex and heterogeneous nature of RTES, stakeholders with distinct concerns and interests, it is necessary to provide multiple modeling languages to cover the several aspects of a system [1]. However, problems could arise due to the combination of multiple models from different languages, and also possible occurrence of errors/difficulties in the communication between different models [2].

Software specification is a fundamental activity in the development process of a system, since it allows one to express its structure, to capture early decisions of the project (contributing to overall system reliability) and also describing their architectural features between components and connectors [3], [4]. RTES design is vulnerable to erroneous choices in the early phases of software design process and, often, will negatively influence development stages and system implementation [5]. For RTES, analysis, description and documentation of artifacts arising from the design process is of great relevance, since this activity contributes to understanding these systems, minimizing the complexity of their description and analysis [6].

Due to particular demands related to RTES specification and the inherent characteristics of this domain, industry has often applied multiple languages in order to model requirements. Thus, it is necessary to minimize inconsistencies in the generated models. A variety of *Domain-Specific Modeling Languages*(DSMLs) has been proposed as extensions to the UML metamodel, also designated as *profiles*, such as SysML [7] and MARTE [8]. UML definition includes several semantic variables and supplies special constructors in the language for refinements. These constructors, stereotypes, labeled values and *tags* are used to define the DSMLs.

The research strategy proposed here has focus on the combination of SysML and MARTE profiles for designing RTES. In addition, this research has as main objectives to analyze the consonance between methods for modeling requirements of RTES and the different markings/stereotypes offered by MARTE profile and, still, to understand how MARTE packages, in special GRM, NFP and Time, could strengthen initial architectural settings of RTES. As a working example, this paper presents SysML models for the description of requirements and structural characteristics of the Body Comfort Control System, through the MARTE profile, including temporal properties, event description, hardware resources, among others. The developed research proposes to use a specific diagram for the description of dif-

URI: http://hdl.handle.net/10125/41906
ISBN: 978-0-9981331-0-2

ferent types of requirements (related to automotive control systems) and the representation of the initial architecture components which characterizes the system.

## 2. Background on Real-Time Modeling and Specifications

Literature has often presented works that describe strategies to combine SysML and MARTE *profiles* for modeling, specification and detailed design in the field of RTES, as briefly presented in this section.

The work published in [1] discusses strategies to combine MARTE and SysML in a single modeling framework, with the aim to cover the needs of multiple perspectives in modeling RTES, and also to alert about possible inconsistencies in the joint use of multiple profiles. In [9], the authors describe the characteristics of a research conducted within the *MeMVaTEx* project in which a method to express and track requirements, in the field of automotive applications, is created. With this purpose, the authors integrate EAST-ADL2, AUTOSAR, MARTE and SysML to define three modeling perspectives (requirements model, solution model and V&V). The MADES methodology [10] focuses on a subset of SysML and MARTE to express different aspects related to real-time systems, but only the temporal notations of the Time package for each proposed level of abstraction is used.

Authors of paper [11] present a new proposal for a combination of MARTE and SysML profiles to allow specification and requirements management. The main objective of article [12] is to present the combined application of SysML with MARTE stereotypes, which enables the specification of different features of individual software requirements. Article [13] explores the use of UML profiles SysML and MARTE for the modeling of Real Time software requirements, with its main area of application being the control of urban traffic. In [14], the authors propose a multi-view approach based on MARTE and SysML for modeling of energy consumption. In this approach, each domain may be treated separately through different views, maintaining connections of model elements with other views. Nonetheless, despite showing some of the MARTE packages that contribute to real-time system modeling, each view is exposed as a black box. The authors describe the definition of each vision without presenting, accurately, which MARTE elements were employed. In [15], the authors present an approach suitable for describing the processes of engineering requirements by using in conjunction UML Use Case diagrams, UML Class diagrams, UML Sequence diagrams, SysML Requirements diagram and the MARTE profile. The authors proposed an approach entitled MDEReq (*Model Driven Engineering for Requirement Management*) for requirements specification and change management.

The research presented in this paper is different from previous works by proposing the development of a comprehensive approach to cover two fundamental activities of RTES development, requirements specification and detailed design of these systems. In addition, this work, differently from the others, focuses on constructors and stereotypes of MARTE packages NFP, Time and GRM, by presenting precisely the semantics and syntax of MARTE stereotypes employed to facilitate the comprehension of the proposed modeling strategy.

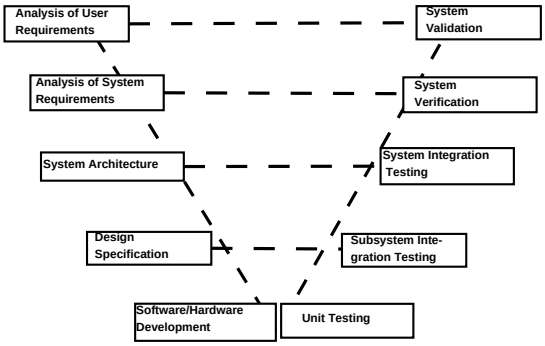


Figure 1. V-Model

## 3. V-Model and Covered Processes in this Research

The V-Model, depicted in Fig. 1, is an evolution of the Waterfall model. It still brings some of its disadvantages including the sequential nature in development stages. However, there has been employed great efforts in each one of the initial specification stages in order to facilitate the final stages of system validation and integration.

The letter $V$ reflects the two main flows indicated for the development process. The left side presents characteristics which are related to the user of the system, viability studies, architecture definition and detailed project. In addition, other activities in this side reflects the defined activities of the project, including implementation and verification. The right side of the V-Model presents all the integration activities and tests that are performed for the other activities, at each level, exposed on the left side. System requirements are defined, structured and detailed on the left side and are symmetrically analyzed and verified on the right side. Thus, it can be claimed that the plans, developed on one side, direct the activities to be performed on the other. For instance, the specified activity in the model (on the left side), named Analysis of System Requirements, is directly related to the activity System Verification (in the right side).

This proposed research establishes different contributions for activities of system requirements analysis and architecture when proposing modeling strategies for requirements specification and design of RTES. SysML Requirements diagram is considered for System Requirements analysis activities, together with MARTE constructors and stereotypes for the purpose of enriching the models and allowing description of non-functional properties already in initial models. SysML Block diagrams are applied for the Architecture Systems phase with MARTE Time and GRM

packages in order to strengthen description of the physical and logical components of the system.

## 4. MARTE Constructors and Stereotypes for RTES

The architecture of the MARTE *profile*, depicted in Fig. 2, consists of three main packages named MARTE Foundations, MARTE Design Model and MARTE Analysis Model. Package **MARTE Foundations** aims at defining fundamental concepts for RTES, and brings concepts that serve as a general basis for the description of most of the elements linked to the remainder of the specification. Package **MARTE Design Model** provides the necessary support to conduct a detailed specification of a project for RTES. Package **MARTE Analysis Model** provides concepts for verification and validation of models [8]. In addition to these packages, the MARTE profile proposes numerous other sub-packages. Package **MARTE Foundations** relates to the scope of this work, more specifically the CoreElements, NFR, Time and GRM sub-package, which are briefly introduced bellow.

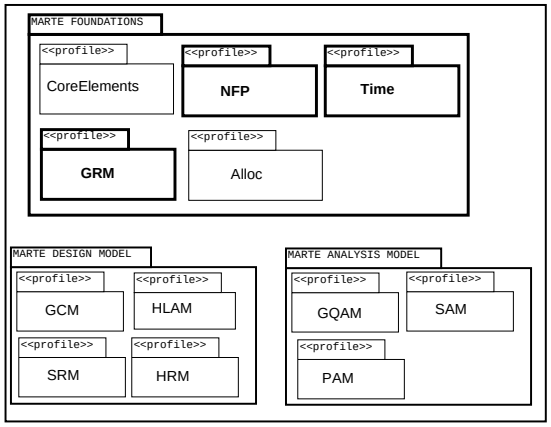


Figure 2. MARTE Profile [8]

Sub-package **CoreElements** presents a number of important concepts to the elements of MARTE specification, being useful to define other more elaborated elements. This package is subdivided into two packages. Package Foundations holds the basic elements used to represent the dual descriptor/instance nature of any modeling entity. This package is important to provide several theoretical concepts that are relevant to the entire MARTE specification. Package Causality describes the basic elements necessary for behavioral modeling, and their run-time semantics.

Sub-package **Time** allows modeling of time and related structures over time. Concepts related to physical time, logical time, and representation of instants representing time bases, and occurrences of events over time, are defined in this sub-package. Time sub-package describes a general framework for representing time and appropriate concepts/mechanisms for modeling aspects of RTES. Thus, it is useful to describe different characteristics of systems such as, for example, delays, durations, chronometric time and logical time.

Sub-package **NFP** describes the domain model for specification and description of non-functional properties (NFPs) of RTES and shows how these properties can be connected to UML model elements. NFP provides the capability to describe various types of values for physical quantities such as time, mass and energy. These values are used to describe the architecture, the behavior and properties of a model for a computer system, via components of modeling. The framework annotation NFP has many aspects which are grouped into three main sub-packages, named as *NFP_Nature*, *NFP_Anotation* and *NFP_Declaration*, which together allow to create a wide range of non-functional annotations in model elements.

Sub-package **GRM** aims to offer different concepts necessary to model a generic platform for execution of RTES. The GRM package is appropriate for modeling execution platforms at different levels of detail, hardware modeling, such as storage units or physical communication channels, and software, for example, Real-Time Operating Systems, and to provide fundamental modeling constructors.

These sub-packages deal with the semantic representation of concepts through stereotypes and marked elements that can be added in elements of the model in order to express their embedded and real-time properties. Contextualization for each MARTE constructor, described in the following subsections, derives from interpretation of the authors and through analysis of annexes $F.2$, $F.3$ e $F.4$ of MARTE profile [8], the semantic and syntactic information presented in NFPs, Time and GRM packages of MARTE, the research presented in [16] and also in works which applied MARTE stereotypes in their models, as presented in the literature review.

Due to space constraints, in the subsequent subsections, we only introduce the semantic definitions for stereotypes that were used in the models developed in this research. A prior analysis of each MARTE constructor facilitates the understanding and adoption of modeling presented elements.

### 4.1. NFP Stereotypes

NFPs provide information about different characteristics, as for example throughput, delays, overheads, scheduling policies, deadlines, or memory usage.

Stereotype **ConstraintKind** is an enumeration type that defines literals used to specify the nature of constraint assertions by either required, offered, or contract nature. A constraint of type required will define the minimum/maximum qualitative or quantitative demand by constrained elements.

Stereotype **Nfp** is intended to declare, qualify, and assign extended data types to NFP values. Stereotype **NfpType** is a type whose instances are identified only by NFP value specifications. A NfpType contains specific attributes to support modeling of NFP tuple types. Stereotype **NfpConstraint** aims to apply a condition or restriction to modeled elements. Restrictions for NFP support textual expressions to specify

assertions about performance, scheduling and other characteristics of embedded systems, and their relationship to other features by means of variables, mathematical, logical, and time expressions.

### 4.2. Time Stereotypes

Within MARTE, each sub-package of the Time package has a set of stereotypes which allows representation of created models based on different characteristics of time, including physical and logical time, concepts and semantics related to time. Stereotypes and the possible enumerations to a temporal element, related to this research, are detailed as follows.

**TimedElement** is an abstract stereotype that must be used to associate one clock(s) to a UML model element. A TimedElement is an abstract class, generalization of all other timed concepts. It associates a non empty set of clocks with a model element. Stereotype **Clock** maps Clock and gives access to time. A Clock is a model element that represents an instance of ClockType. Stereotype **ClockType** is a classifier for Clock. Attributes define the nature of the represented time (discrete or dense), the type of units, and whether its instances are logical clocks or chronometric clocks.

Time modeling introduces two stereotypes for Constraint stereotypes that specializes the NfpConstraint stereotype (of NFP package), that are **TimedConstraint** and **ClockConstraint** (as TimedElement, both stereotypes refer to clocks). Stereotype **TimedConstraint** deals with constraints imposed on either instant value or on duration value, according to the value given to the interpretation attribute. It also relates indirectly to TimedInstantConstraint and TimedDurationConstraint. A **ClockConstraint** stereotype imposes dependency between clocks or between clock types. A ClockConstraint refers to a set of clocks or clock types, and possibly to other model elements.

Stereotype **TimedProcessing** represents activities that have known start and finish times or a known duration, and whose instants and durations are explicitly bound to clocks. Stereotype **TimedInstantObservation** denotes an instant in time, associated with an event occurrence, and observed on a clock. The obsKind attribute may specify the kind of the observed event. Stereotype **TimeInterpretationKind** is an enumeration type that defines literals used to specify the way to interpret a time expression: either as a duration or as an instant. Stereotype **TimeNatureKind** is an enumeration type that defines literals used to specify the nature discrete or dense of a time value.

### 4.3. GRM Stereotypes

GRM package allows to represent characteristics related to the resources that are physically or logically related to the design of RTES. Stereotypes provided in this package allow to annotate, in elaborated models, concepts related to physical resources, resource synchronization, temporal resources, communication resources, processing resources and the demand for a resource scheduling.

Stereotype **Resource** is useful to provide further refinement and for representation of generic resources from a holistic system-wide perspective. The nature of the concrete element defines the domain concept that it represents. This stereotype represents a generalization to all stereotypes presented in this section and is important for the root concepts defined for modeling of resources.

Stereotype **ConcurrencyResource** is a protected active resource that is capable of performing its associated flow of execution concurrently with others, all of which take their processing capacity from a potentially different protected active resource (eventually a ComputingResource). Concurrency may be physical or logical, when it is logical the supplying processing resource needs to be arbitrated with a certain policy.

Stereotype **ProcessingResource** is an active, protected, executing-type resource that is allocated to the execution of schedulable resources, and hence any action that use those schedulable resources to execute. Specializations of *ResourceProcessing* are the *CommunicationMedia*, *ComputingResource* and *DeviceResource* stereotypes. Stereotype **CommunicationMedia** represents the means to transport information from one location to another. Stereotype **ComputingResource** represents either virtual or physical processing devices capable of storing and executing source code. Hence, its fundamental service is to compute. Stereotype **DeviceResource** typically represents a physically or logically persistent entity that offers one or more services. Resources and their services are the available means to perform the expected duties and/or satisfy the requirements for which the system under consideration is aimed.

A *Scheduler*, *SchedulableResource* and *SecondaryScheduler* provide UML with extensions for scheduling in the GRM package. A **Scheduler** is a stereotype that is defined as a kind of ResourceBroker that brings access to its brokered ProcessingResource or resources following a certain scheduling policy. Stereotype **SchedulableResource** is an active resource able to perform actions using the processing capacity from a processing resource that is managed by the scheduler. A SchedulableResource is defined as a kind of ConcurrencyResource with logical concurrency. This implies that it takes the processing capacity from another active protected resource, usually a ProcessingResource, and competes for it with others linked to the same scheduler. Stereotype **SecondaryScheduler**, a specialization of the Scheduler, provides the concept introduced to support hierarchical scheduling schemes. It is conceived as a kind of Scheduler, for which the processing capacity that will be shared among its scheduling resources is not obtained directly from processing units, but from other scheduling resources.

## 5. Body Comfort Control System

Automotive control systems are complex systems composed of sensors, actuators, components, control systems, supervisory control, data acquisition and other methods

using electrical, electronic, mechanical and computer resources [17]. According to [18], there are five different main domains, which are described below, to be explored in the development of automotive systems.

The described modules are named as power train domain, chassis domain, body domain, HMI domain and telematics domain, as depicted in Fig. 3.

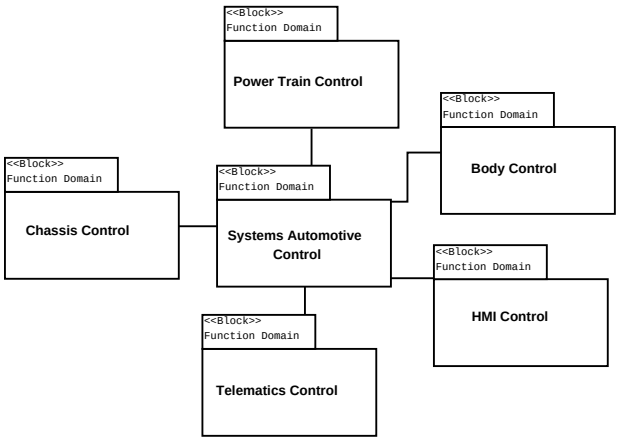


Figure 3. Main Software Domains for Automotive Systems

The **Powertrain Domain** deals with systems of the vehicle motorization, including the engine, data transmission between brakes, accelerator and gear, speeding up, among others. The **Chassis Domain** is composed by systems whose aim is to control the interaction of the vehicle with the road (wheel, suspension). In this domain, the embedded systems are mainly correlated to driving and braking. They have to ensure the comfort to the driver and passengers, as well as their safety.

The **Body Domain** is composed by entities that offers support to activities, comfort and safety related to users. In this domain, the embedded systems are mainly related to airbags, illumination, window raiser, air conditioning, windshield, and so on. The **HMI Domain** consists of equipments which allows information exchange between electronic systems and the driver. In this domain, the embedded systems represent the information about the status of the car (e.g., vehicle speed, oil level, status of a door, status of lights), status of multimedia devices (e.g., current frequency for a radio device), or the result of a request (e.g., visualization of a map provided by a navigation system).

The **Telematics Domain** is related to components which allows information exchange between the vehicle and the external environment including, for example, radio, navigation system, and Internet access.

This research will explore and analyze, for the development of the models mentioned above, functional and non-functional requirements related to the Body Domain or, more specifically, to the Body Control Systems.

# 6. An Example in Automotive Domain

Automotive control systems are complex systems composed of sensors, actuators, components, control systems, supervisory control, data acquisition and other methods using electrical, electronic, mechanical and computer resources [17]. The Body Domain [18] contains the embedded functions of a vehicle which are not directly related to its dynamics, but strongly relates to essential components which allows greater comfort and safety to drivers. Nowadays, different functions which are coupled in vehicles as windshields, lights, windows, seat controls and mirrors are controlled by software-based systems. Figure 4 is a graphic diagram which describes the main components of a Body Comfort Control System.

The **Door Control** corresponds to locking and unlocking control systems, according to internal users requests, to the control signs (sensors and actuators) or, yet, automatic response, for instance, after vehicle inactivity. The **Windshield Wiper** represents management systems for back and front wipers. The **Control Seat** systems describes the embedded systems responsible to configure height, backboard and other automatic functions related to seat controls.

The **Air Conditioning Control** system refers to functionalities able to manage several temperature conditions of the vehicle and to provide different states according to the environment conditions or the users' preferences. The **Window Control** controls the window lifter of front and back doors. The **Lighting Control** presents the functional modules to operate frontal lights, back lights, brake lights and emergency brake lights.

**Airbag Control** is composed by subsystems that diagnose collisions through internal sensors (drivers and passengers compartments), external sensors, through physical devices for manual activation and deactivation and by the diagnosis module (able to evaluate the operation, when the vehicle is turned on, of the airbag system).

Initially, the structural model is designed by SysML Block diagram and MARTE profile, which represents the functional Body domain. Proposed model relates the early architecture to four important modules that comprise this domain. In these models, functional block window control, the Windshield Wiper Control, Light Control and Door Control are considered. Subsequently, requirements specification will be performed.

## 6.1. Structural Modeling Operational Module Block Diagram

SysML Block diagram represents the system components and their relationships. A block can represent the characteristics of a physical or logical entity, such as a hardware component or a physical object. Blocks can have attributes and operations and these represent, respectively, the internal properties of the block (shown in the second compartment) and operations that describe the behavior presented by the block (shown in the third compartment).

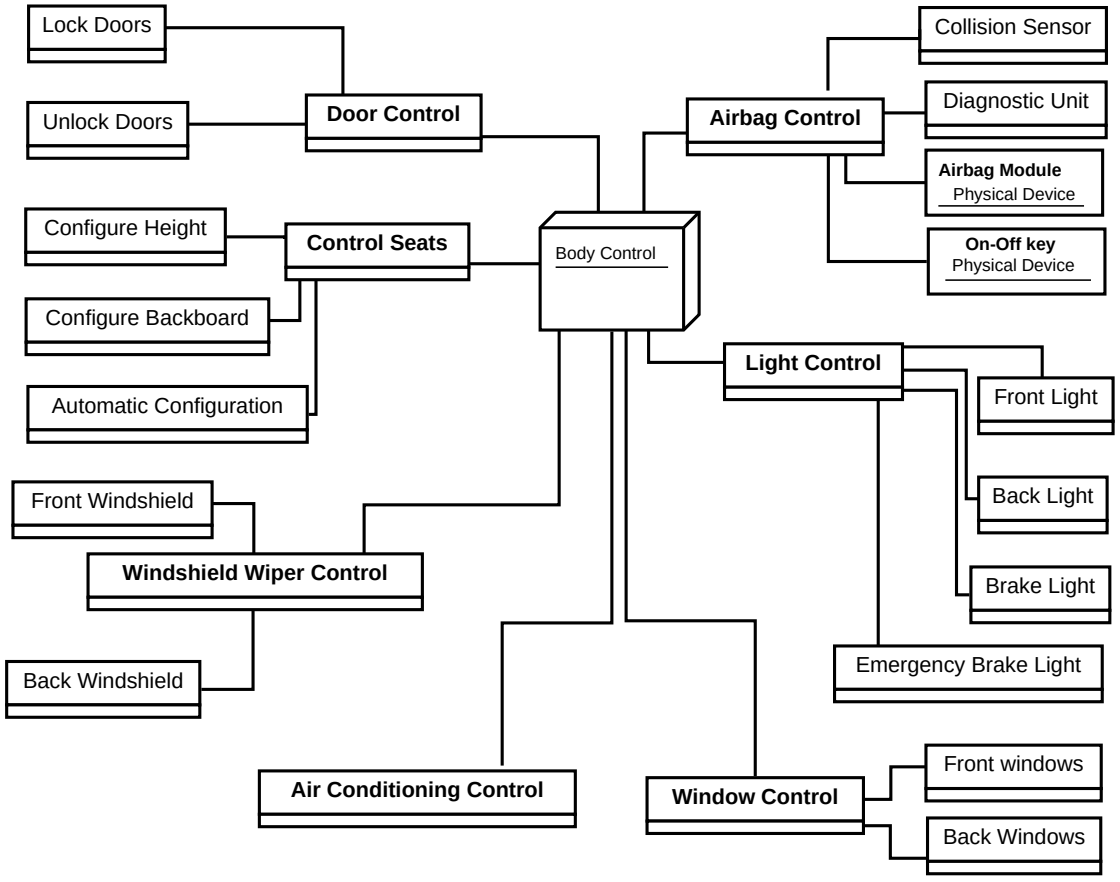


Figure 4. Structure of the Body Comfort Control System

Figure 5 depicts the structural model for the Body domain, representing controls and communications for the functional modules of the Comfort Body of an automobile system. Central ECU is a physical component that has the logic needed to controlling sensors, actuators and the different actions that represent embedded and control activities of automatic devices of an automobile. It is responsibility of the central ECU the coordination of actuators that provide the expected behavior for each component and, also, communication and control of the remaining ECUs embedded in the automobile.

The structural model consists of four main ECUs named Door Control, Window Control, Windshield Wiper Control and Light Control. As observed in the elaborated model, each one of the main ECUs were doubly stereotyped, once these components represent resources with process capability and are able to schedule other resources. The SchedulableResource (from MARTE::GRM::Scheduling) allows to explicit that each component represents a scheduler in a logical level, because it is up to it to receive and process the requests for the other peripheral ECUs in a synchronous way. This component represents a physical resource of the system with logical processing capability (also stereotyped as ComputingResource) that simultaneously processes the received requests from each one of the peripheral ECUs. In addition, it is able to communicate with the network controller and, also, is able to schedule different and simultaneous states to set an actuator so that it controls an open right front window, a semi-open right front window, among other tasks. Furthermore, the ComputingResource (from MARTE:GRM::ResourceTypes) stereotype may represent both physical and logical processing devices able to execute programs. This stereotype was added to each main ECU for representing a physical or logical component/device that offers a fundamental service of the system (main elements of Body Comfort domain) and abstracts the fundamental capability of performing any assigned behavior to an element of the model.

Each of the embedded modules of the automotive system is connected to a LIN (Local Interconnect Network) communication controller that is able to connect different modules of integrated mechatronic and autonomous systems. The block that represents the Communication Controller, in Fig. 5, is the system component that, in addition to perform the interconnections between the ECUs, abstracts data switches between each of the peripheral ECU and the main ECU. The deviceResource, from MARTE:GRM::ResourceTypes, represents a physically or logically persistent entity that offers one or more services of the system. This notation was used, in the LIN component, to indicate modules which

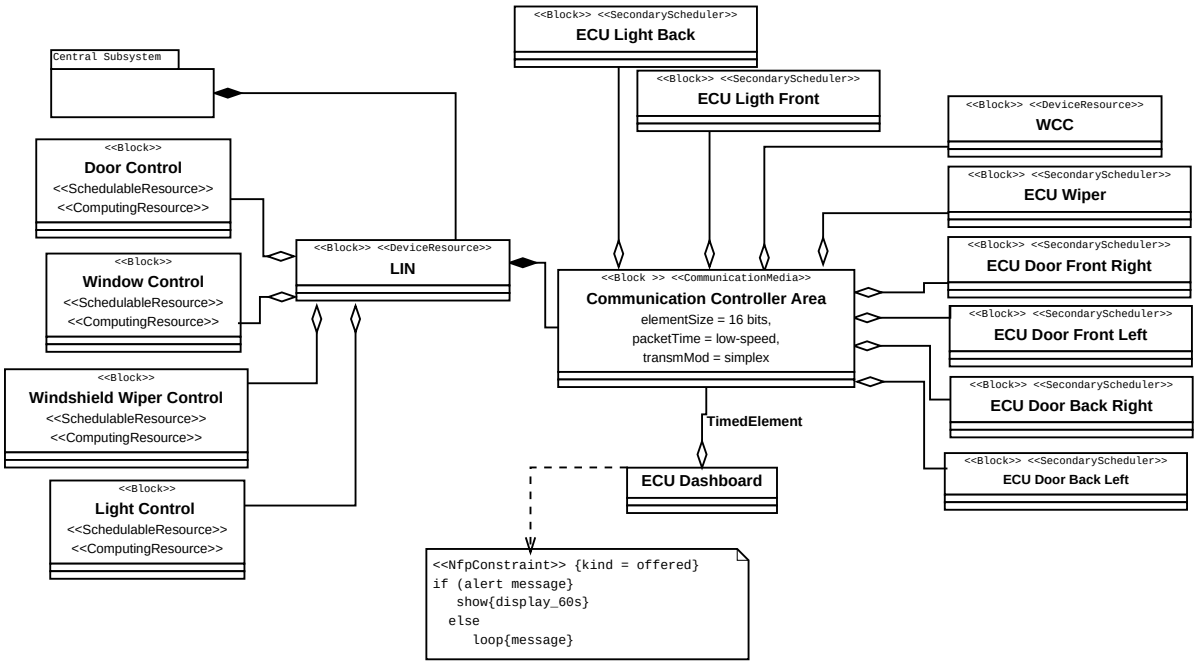


Figure 5. SysML Block Diagram with MARTE Profile for Body Domain

represents a physical resource and/or interacts with external devices.

Each one of the ports (left and right front doors, left and back doors) has a local control module, i.e., a specific ECU which sends a signal, through Communication Controller, to the central ECU. In this way, the local ECUs are connected to a low-speed transmission channel (not shown in this modeling) which is controlled by a module named Communication Controller Area (similar to components used in real applications). The block that represents the Communication Controller, in Fig. 5, is the system component to manage the exchange of information/actions between each of the peripheral ECU and the main ECU.

Communication Controller component is labeled with stereotype CommunicationMedia, from MARTE::GRM::ResourceTypes, to describe/represent the different properties related to transportation of data of each peripheral ECU for their respective main ECUs. This stereotype possesses the elementSize attribute which represents the maximum size in bits that can be transmitted and depends on the capacity of the communication channel (in this case 16 bits for transmission); packedTime, which allows to explicit the communication transmission defined here as “low-speed”; and the transmMod which allows to specify the transmission knot and attributed values as, for instance, simplex, halfduplex, and full-duplex.

All ports, in addition to requests from the driver and passengers, can be locked or unlocked by means of signals transmitted over wireless networks, and for that reason the WCC component (embedded component able to capture wireless transmissions) is also connected to the Communication Controller Area. When receiving a certain signal, through the network, it will be able to retransmit for the main ECU which is responsible for sending the commands to the physical device in question. This WCC component, being a physical entity that provides blocking and unblocking services of the doors through wireless communications is also stereotyped as deviceResource (from MARTE:GRM::ResourceTypes).

Peripheral ECUs, related to the control of doors, are responsible for transferring information related to the state of doors and, also, the windows of each port. The ECU Wiper encompasses information to be sent to the control module of the Windshield Wiper Control operating with receptors such as, for example, the automatic operation of wipers (perceiving the existence of rain), the actuators and pre-established settings for the windshield wipers. ECUs Right Front Light and Right Back Light cooperate with the Light Control module, able to control the firing of the front and back light, of the brake Light and the Emergency Brake Light. ECUs of each door are stereotyped with SecondaryScheduler, from MARTE:GRM::Scheduling, once these ECUs directly receive demands for each component of the vehicle (lock/unlock, window up/down, seat up/down). This stereotype represents the concept of processing division, that is, the capacity of processing, acting and controling are not expected in this stereotyped component, but performed by other resources/components (in this case, those defined as SchedulableResource).

All information representing the status of doors, windows, lights and Windshield Wiper can be presented, to the driver, through the information that is transmitted by means of main ECU for the dashboard ECU via low-speed data network (not shown in this modeling). Textual restriction

NfpConstraint (from NFP_Annotation) was added to the element to specify the persistence time of a message/alert to the vehicle's driver. This restriction allows to add in the structural model of the system temporal assertions to warning messages (displayed for 1 minute) and, also, anomaly messages displayed when a problem is detected.

Stereotype TimedElement, from TimeRelatedEntities::TimedElements, is added to the preexisting association between the Communication Controller Area block and the ECU Dashboard block. This stereotype describes a relation between a model element with other element that has, within its definition, temporal definition (that is, communication with physical or logical clock). In this case, it is important to highlight that the ECU dashboard component will have temporal definitions, in its implementation, to represent the information to passengers in timed intervals.

In this section, the structural model of the system presented in Fig. 4 is enriched, in Fig. 5, with different stereotypes of packages GRM and NFP. Overall, these constructors allow the derailment of important characteristics of a component, establishes temporal criteria in modules which possesses these restrictions, present definitions of communication and transmission of data resources and, also, delimits different physical and logical application resources.

## 6.2. Requirements Specification of the Operational Module

SysML Requirements diagram provides support for modeling different types of individual requirements and their relationships, and also allows to model requirements relationships in various manners. These include relationships for defining a requirements hierarchy, deriving requirements, satisfying requirements, verifying requirements, and refining requirements.

Original SysML Requirement diagram allows one to describe for each requirement the identifier (ID) and a textual information, usually in natural language, with a global description of the requirement. As highlighted in [17], those descriptions are commonly presented in natural language, making the specification intuitive, but may culminate in ambiguous and less precise requirements specifications [19]. With the purpose of minimizing ambiguities in the requirements specification, it is possible to combine MARTE constructors with an atomic definition of a requirement. SysML Requirements diagrams are enriched with MARTE constructors which are described below, as presented in a simple example model for the Control Module Door Control (presented in Fig. 4). The set of general requirements considered in this specification are:

1) RB1: To check of periodically ECU Door Front Right.
2) RB2: To check of periodically ECU Door Front Left.
3) RB3: To check of periodically ECU Door Back Right.
4) RB4: To check of periodically ECU Door Back Left.
5) RB5: Locking the doors in a maximum of 50 ms and wait for acknowledge for 30 ms.
6) RB6: Unlock doors and wait for acknowledge for 30 ms.
7) RB7: Check internal inactivity and after 180000 ms must activate locking. Receiving acknowledge must not exceed 30ms.
8) RB8: Processing request WCC, with a maximum delay of 50 ms.
9) RB9: After acceleration more that 25 km/h or after 120000 ms in motion activate doors' locking.
10) RB10: Scheduling simultaneous requests.

Complete requirements specification for the door control module, the Comfort Body domain, is depicted in Fig. 6.

According to the definition of the MARTE profile, stereotype TimedProcessing, from *TimeRelatedEntities::TimedProcessingModels::TimedProcessings*, represents the model elements that own or perform activities whose period, time and duration are known. Requirements $RB1$, $RB2$, $RB3$, $RB4$, $RB5$, $RB6$, $RB7$ and $RB8$ are stereotyped as *TimedProcessing* to perform temporary and periodic actions that relates to a clock. As can be observed in Fig. 6, each of the requirements described above relates to the *Chronometric clock* by means of *TimedElement*, from TimeRelatedEntities::TimedElements. A *TimedElement* is a stereotype that allows a model element associated with one chronometric clock. Element Chronometric clock represents definition of the clock that implicitly refers to physical time.

Stereotype *TimedInstantObservation*, from *TimeRelatedEntities::TimedObservations* is also employed in the models in order to describe the observation of events and activities in a period of time (i.e., it refers to physical clocks). Attribute *obsKind* is an enumeration to define the type of event that is being observed. In this case, this attribute is set to "receive", since that it is expected to receive values of vehicle speed or of the elapsed time after the vehicle is in motion. In addition to MARTE stereotypes that were applied to describe temporal characteristics to requirements of the Control Module Door and that were previously contextualized, this research proposes, yet, non-functional restrictions to define the function for each modeled requirement. A *NfpConstraint* (from *NFP_Annotation*) is a mechanism which allows the insertion of conditional information or restrictions of elements of the model. In this work, all the definitions for a requirement that are explained by attribute "*Text*" were complemented with semantics provided by stereotype *NfpConstraint*. Being that, such adaptations, in the form of constraints are important to standardize and to minimize ambiguity.

Keywords were used, in the functional description of each requirement, referring to major actions which it executes. For example, terms **AFTER**, **BETWEEN**, **WAIT** and **SUPERIOR** explicit, respectively, values after a time interval, values within a time interval, temporal values of waiting and values of space/time exceeding limit in milliseconds.

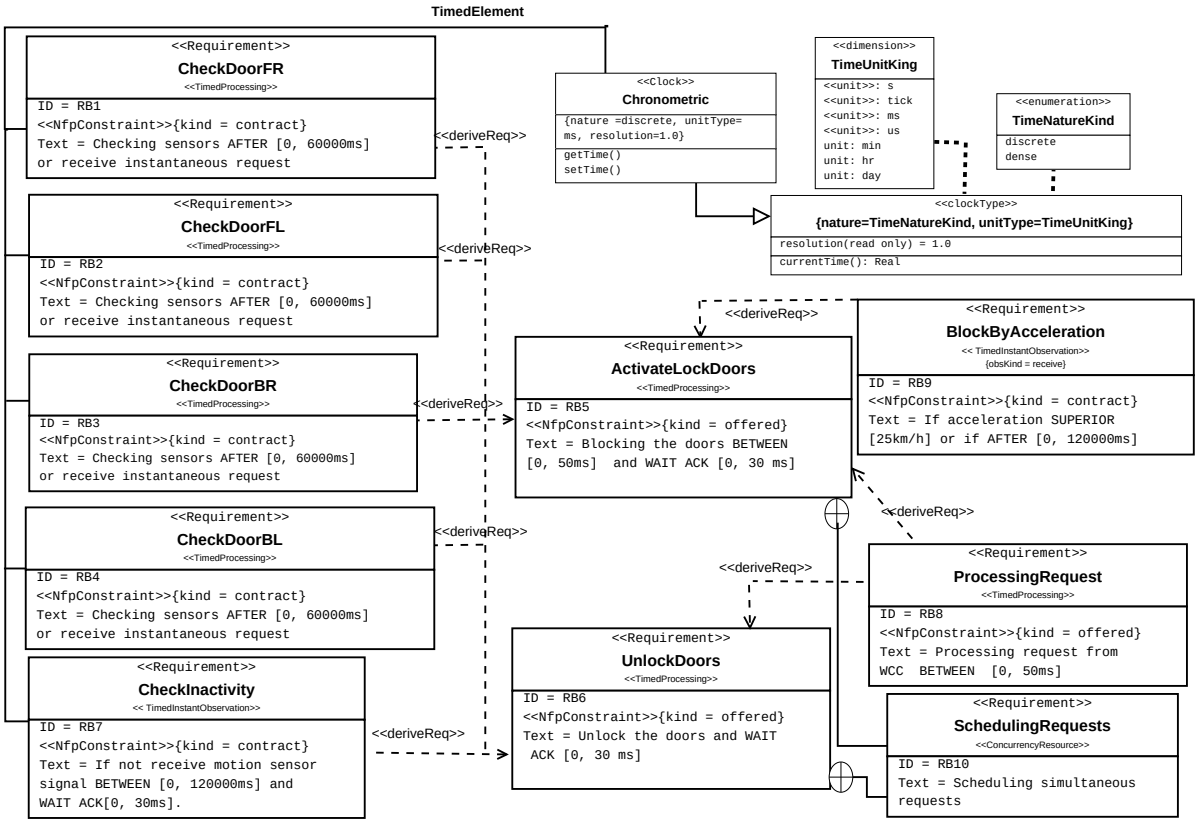


Figure 6. Requirements Model with Stereotypes from the MARTE Profile.

Moreover, stereotype *NfpConstraint* allows qualifying NFP constraints by either *required*, *offered*, or *contract* nature for a model element. Requirements that are labeled with attribute *offered* describes the space of temporal values for a model element. Requirements $RB7$ and $RB9$, which are of type *contract*, specify a conditional expression that the requirement must satisfy on defining its functions.

Requirement $RB10$, stereotyped with *ConcurrencyResource*, from *MARTE:GRM::ResourceTypes*, shows that the achievement of this non-functional requirement of the system should present logic and/or physical concurrence strategies. Thus, it describes strategies that will be needed to attend to and schedule simultaneous requests of various embedded components, in accordance with certain policy.

Temporal and non-functional constraints, in general, when added in early specification models of a system will positively guide other stages of development. In fact, the designer must prove that the response time is always acceptable and, therefore, that the responses for each stimuli are made in a limited interval.

## 7. Discussion

Research strategy adopted in this work is justified by the expressive power of the MARTE profile. Packages NFP, Time and GRM, used in conjunction with SysML models, allow, respectively, to explicit non-functional properties in models that were strongly related to RTES, to show different temporal concepts, implicit to RTES, and to detail characteristics of an embedded platform.

Analysis and comprehension of the domain model and of the constructors/stereotypes presented in package NFP becomes relevant in this research because the explained stereotypes describe relevant concepts in RTES domain. Therefore, diverse non-functional properties and particular aspects of NFPs of model elements can be achieved, through restrictions annotated in the models and, also, definition of different types of relations between modeled elements.

Comprehending the domain model of the Time Package become crucial to this project, considering that, through its constructors, it is possible to explicit, in the models, requirements and temporal concepts (delay, periods, duration, *clocks*, physical time, logical time) inherent and necessary for documentation, treatment and comprehension of real-time applications. In this work, stereotypes were used in the structural model and, mainly, in the system requirements models. Different temporal restrictions imposed to requirements were added, in a simple way, already in the requirements diagram.

Proposed research, when compared to correlated works, presents some improvements. Two specific diagrams for requirements engineering and system design were used. In this case, it was chosen not to extend the profiles to avoid unnecessary information doubling. Furthermore, it

was presented in a succinct and intuitive way the conformity in the application of stereotypes of packages GRM and Time. In this way, it was clearly presented which MARTE elements were used and, consequently, it becomes possible to comprehend and replicate a proposed modeling strategy.

## 8. Conclusion

This paper presents applicability of MARTE and SysML profiles in the context of detailed design of RTES. These models have been drafted by combining annotations provided by the MARTE profile, more specifically by packages NFP, Time and GRM, associated with SysML diagrams that are well-known in the process of requirements and system design.

SysML Block diagram is employed in order to introduce the relationships, the dependencies and the main functional components of the control system of the body control doors domain. Stereotypes from GRM package establish the first step towards the model-based design approach for RTES. When evaluating the elaborated models, one can observe that each component of the modeled system is characterized in a detailed way, and that constructors of the GRM package and NPF package increase the amount of information provided. In addition, restrictions are considered already in the early stages of system specification. In general, understanding the system design is more detailed already at the architectural level.

SysML Requirements diagrams have been employed, as described in the literature review, on different projects and purposes. However, it is possible to note that non-functional settings as, for example, restrictions on modeled elements, temporal values, delays, overheads, scheduling policies are not possible to be represented in the original SysML Requirements diagram. Thus, applying in proposed models the annotations and stereotypes available in Time, NFP and GRM packages contributes to describe functional and non-functional requirements in the initial models of a system.

As for future work, we plan to establish the adoption of VSL (Value Specification Language) for all specifications and marked values inserted into the models. This proposition is justified by the need to specify requirements in a formal way, i.e., to formalize and to express it in the body of a requirement and, also, in other existing annotations in the modeled elements and their relationships. With this, it becomes possible to verify the correctness and consistency of models developed in accordance with the presented specification.

## Acknowledgment

The authors would like to thank CNPq (www.cnpq.br) Grant 445500/2014-0 for the financial support.

## References

[1] H. Espinoza, D. Cancila, B. Selic, and S. Gerard, "Challenges in Combining SysML and MARTE for Model - Based Design of Embedded Systems." in *5th European Conference*, ser. Lecture Notes in Computer Science, vol. 5562, 2009, pp. 98–113.

[2] F. Noyrit, S. Gerard, F. Terrier, and B. Selic, "Consistent Modeling Using Multiple UML Profiles." in *Model Driven Engineering Languages and Systems*, ser. Lecture Notes in Computer Science, D. Petriu, N. Rouquette, and y. Haugen, Eds., vol. 6394, Oslo, Norway, 2010, pp. 392–406.

[3] F. Oquendo, B. Warboys, and R. Morrison, Eds., *Software Architecture: The Next Step.*, ser. Lecture Notes in Computer Science, vol. 3047. Berlin, Heidelberg: Springer, 2004.

[4] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice.*, 3rd ed. Boston, MA, USA: Addison-Wesley Professional, 2012.

[5] M. Shaw and D. Garlan, *Software Architecture: Perspectives on an Emerging Discipline.*, 1st ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

[6] N. Medvidovic and R. N. Taylor, "A Classification and Comparison Framework for Software Architecture Description Languages." *IEEE Trans. Softw. Eng.*, vol. 26, no. 1, pp. 70–93, 2000.

[7] U. OMG, *OMG - OMG Systems Modeling Language - version 1.4.*, 2015, tech. Report Formal/2015-06-03.

[8] M. OMG, *Modeling and Analysis of Real-Time and Embedded Systems (MARTE)- version 1.1.*, 2011, tech. Report Formal/2011-06-02.

[9] A. Albinet, S. Begoc, J. L. Boulanger, O. Casse, I. Dal, H. Dubois, F. Lakhal, D. Louar, F. M. A. Peraldi, Y. Sorel, and Q. D. Van, "The MeMVaTEx Methodology: From Requirements to Models in Automotive Application Design." pp. 1–10, 2010.

[10] I. R. Quadri, L. Soares, I. Gray, L. S. Indrusiak, and A. Bagnato, A. Sadovykh, "MADES: A SysML/MARTE High Level Methodology for Real-Time and Embedded Systems." in *Proc. of the 10th Embedded Realtime Software and Systems Conference*, 2012, pp. 1–10.

[11] M. Saadatmand, A. Cicchetti, and M. Sjödin, "UML-Based Modeling of Non-Functional Requirements in Telecommunication Systems." in *The Sixth Int. Conf. on Software Engineering Advances (ICSEA 2011)*, Barcelona, Spain, 2011, pp. 213–220.

[12] F. G. C. Ribeiro, C. E. Pereira, A. Rettberg, and M. S. Soares, "Model-Based Requirements Specification of Real-Time Systems with UML, SysML and MARTE," *Software & Systems Modeling*, pp. 1–19, 2016.

[13] F. G. C. Ribeiro and M. S. Soares, "An Approach for Modeling Real-time Requirements with SysML and MARTE Stereotypes," in *ICEIS 2013 - Proc. of the 15th Int. Conf. on Enterprise Information Systems, Volume 2*, 2013, pp. 70–81.

[14] C. Gomez, J. DeAntoni, and F. Mallet, "Multi-View Power Modeling Based on UML, MARTE and SysML." in *EUROMICRO-SEAA*, Cesme, Izmir, Turkey, 2012, pp. 17–20.

[15] M. R. S. Marques, E. Siegert, and L. Brisolara, "Integrating UML, MARTE and SysML to Improve Requirements Specification and Traceability in the Embedded Domain." in *12th IEEE Int. Conf. on Industrial Informatics*, 2014, pp. 176–181.

[16] B. Selic and S. Gerard, "Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE." in *Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE*, 1st ed. Boston: Morgan Kaufmann, 2014.

[17] S. Walter, A. Rettberg, and M. Kreutz, "Towards Formalized Model-Based Requirements for a Seamless Design Approach in Safety-Critical Systems Development." in *Int. Symp. on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, 2015, pp. 111–115.

[18] N. Navet and F. Simonot-Lion, *Automotive Embedded Systems Handbook*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2008.

[19] M. S. Soares, J. Vrancken, and A. Verbraeck, "User Requirements Modeling and Analysis of Software-Intensive Systems." *The Journal of Systems and Software*, vol. 84, no. 2, pp. 328–339, 2011.