From high-level modelling of time in MARTE to real-time scheduling analysis

Marie-Agnès Peraldi-Frati[1], Yves Sorel[2],

[1] I3S, UNSA,CNRS/INRIA, Sophia Antipolis Méditerranée, B.P. 93, 06902 Sophia Antipolis, France.

[2] INRIA, Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France.
map@unice.fr, Yves.Sorel@inria.fr

Abstract. An important challenge in the domain of automotive control design is to provide a seamless flow for modelling conjointly with the behaviour, the temporal characteristics and the timing constraints of a system at different abstraction levels. In addition, this flow should provide analysis phases for validating the real-time behaviour of the functional models in regard to these constraints and a specific execution platform, To achieve this goal, we adopt a model-based approach, based on the UML MARTE profile [1], that allows the modelling of a system, with a separation of concerns between the software (functional model) and the execution platform resources (non-functional model) as well as the timing constraints (non-functional model). The temporal characteristics (offset, period) and timing constraints (deadline) are modelled with a high level notion of time called the logical time at the functional level down to the physical time called the chronometric time at the implementation level. From these high-level models we extract the temporal characteristics and the timing constraints of the application relatively to the execution platform, and we apply scheduling analysis techniques in order to provide an implementation which satisfies the timing constraints.

Keywords: UML, MARTE, Methodology, Time modelling, Scheduling Analysis,

1 Introduction

The ever increasing complexity of real-time embedded systems raises the problem of merging inside the development process, different concepts and techniques coming from the domain of software development and others concepts developed for real-time systems design.

UML (Unified Modelling Language) [2] and its domain specific extensions are becoming accepted notations to cope with the design of complex automotive real-time embedded applications.

The recent OMG standardization of the MARTE profile (Modelling and Analysis of Real-Time and Embedded systems) is an important step for modelling non-functional characteristics of real-time embedded systems. An important contribution of MARTE lies in its time model, centred on the notion of multiform time (logical or chronometric), that enriches UML with explicit time model elements (clocks, clocks type …). This is a real advance in regard to the SPT [3] (Scheduling Performance and Time) profile which considers time as an implicit notion closed to the physical time, modelled by a simple annotation onto UML models. In addition, MARTE provides two other models for describing execution platforms, and the allocation of application functions onto the resources of the platform..

Scheduling analyses are based on well founded theory widely used in the domain of real-time control systems. Applying such analysis techniques to a UML design requires extracting from functional and non-functional models the temporal characteristics of every function of these models. The main characteristics are the period, the deadline possibly equal to the period. Analysis techniques need also, as input, the Worst Case Execution Time of the function (WCET). The WCET of a function depends on its execution platform.

In this paper we shall not focus on the transformation of these temporally characterized functions into tasks which depends on the Real-time Operating System (RTOS). Later on, we shall use the term task, which is more usual in the real-time community, instead of function as soon as we shall speak of these temporally characterized functions.

The UML profile MARTE is an interesting answer to the problem of a specialization of UML for real-time embedded systems modelling.

In our approach, we use MARTE for building four models. One for representing the *functional* part of the system mainly composed of activity diagrams, state machine and structure diagrams. Another model called the *time* model contains the main time entities (clocks, clock type, clock constraints) as units for expressing the non-functional temporal characteristics (period) and constraints (deadline). Two other models represent the *resources* and the *allocation* of the functional parts onto these resources. The allocation model allows in particular, an identification of the tasks of the system by exploiting, in addition to non-functional temporal characteristics, other temporal characteristics such as the WCET which actually is hardware dependent.

The time model is composed of multiple clocks which are the inputs of a scheduling analysis of the tasks deduced from the application. As these clocks can be of different nature (logical or chronometric), the model integrates *clock constraints* for merging the clocks and for obtaining a common notion of time in order to verify the schedulability of these tasks according to a specific execution platform.

These complementary models capture different views of an application. Their relationships are based on the semantic of MARTE. This profile is recent but an experimental implementation of the profile is available in the UML editor Papyrus [4]. As we use this editor, we obtain an intermediate format which can be transformed into an input format whose syntax must be compliant with a scheduling analysis tool,

such as Cheddar [5], SynDEx [6], etc. These model transformations are out of the scope of the work presented here.
In this paper, we discuss and illustrate some methodological aspects on using MARTE at the different steps of a modelling approach that integrates both functional and non-functional modelling. Another contribution is to show how it is possible to exploit these models by extracting the temporal information and the implementation characteristics in order to provide a schedulability analysis.
The paper is organized as follows. The section 2 gives an overview of the capabilities of UML and ADL-based languages for modelling time. The section 3 presents the concepts defined in the time model of MARTE. We show how to build a time model by using the clock and the time concepts of MARTE and how to express in a non-ambiguous manner the time units and relations between the clocks of the system. We establish the link between the clock and a functional description. We illustrate the usage of the profile with the example of an ignition control system.
The section 4 presents the transition from a high level notion of time (logical time) to the real-time (chronometric time). We call this step the refinement of time which is performed by integrating and resolving the constraints between clocks and by considering the resources of the intended platform.
The last section is devoted to the exploitation of the models in order to apply a scheduling analysis. The result of such analysis is a starting point for a manual or automatic implementation solution of the functional model onto the execution platform model.

2 High level modelling of time

The adoption of a model based design approach conjointly with UML and ADL (Architecture Description Languages) is becoming promising for real-time embedded systems design [7], especially in the automotive domain. In such domain, a critical issue is to develop separate views of the software independent from the execution platform and to provide mechanisms for the projection/allocation of software onto this execution platform. In addition, both models must be endowed with timing characteristics (functional and non-functional properties).

2.1 UML and time

Applying a model based design approach to real-time embedded system design requires considering the modelling of time. UML is a modelling language for specifying, visualizing, constructing, and documenting software systems and business processes. Several profiles have been standardized by the OMG to cope with real-time.
A first profile called SPT (Scheduling Performance and Time) [3] extends UML1.4 and allows the expression of quantitative temporal information onto structural and behavioural UML models such as activity diagrams, sequence diagrams, events, and structure. This information is defined as instants, duration and observations. They

referred to an implicit notion of time so the semantic of this “implicit” time is not defined.
UML has integrated these concepts in the release 2.0. After that, a new RPF (Request For Proposal) for a UML profile focusing on embedded real-time system modelling has been proposed by the OMG. The result of the RFP heavy process is an official OMG profile called MARTE [1].
MARTE is a UML profile composed of different packages which provide adapted concepts for modelling and analyzing a real-time embedded system. Among these packages, the timing model of MARTE provides a concrete representation of clocks into UML model and, in that way, the access to duration, deadline, period and observation linked to these clocks. Another major concept introduced in the timing model is the different *nature* of clocks dense (the physical continuous time) or discrete (a discretized view of this time) and their different *types*, i.e.: chronometric (concept closed to the physical time) or logical (any repetitive event can be modelled as a clock). These different concepts are presented in the domain view chapter of the MARTE document. The implementation and the usage of these concepts is the profile itself which relies on the UML view. The user of the profile is mostly concerned by the UML view.

2.2 EAST-ADL and Autosar

Architecture Description Languages has been adopted in the automotive domain. EAST-ADL [8] and Autosar [9] are respectively a language and the standard that allows a separation of concerns between the functional view of an application, and the non-functional ones (execution platform, environment and implementation views). This separation of concerns is compatible with a Model Based Design approach, [7][10].
Another orthogonal concept of these ADL is the decomposition of a design in domain-oriented modelling levels. To that aim, EAST-ADL provides specific model elements for catching vehicle features description, control/command modelling, software-oriented design, whereas Autosar covers the implementation level.
EAST-ADL and Autosar focus on a structural description of a system. The behavioural parts are differed to external formalisms (native languages such as Matlab/Simulink diagrams or C code…). The temporal characterization of the structural parts is limited to the expression of requirements associated with ADLFunctionType/Prototypes which are the elements for modelling the structure in EAST_ADL and Autosar software components. Such modelling of temporal aspects is not sufficient for applying timing analysis onto these models. A scheduling analysis requires a precise association of *temporal characteristics* (period and WCET) and *constraints* (deadline) to the different functionalities. Worst Case Execution Time of computations or communications between functions depends on the execution platform resources such as the CPU, the network, the bus, etc. Results on the transformation of AADL models for scheduling analysis are presented in [11]
Projects such as ATESST [12] and TIMMO [13][14] consider the problem of extending EAST-ADL2 and Autosar with MARTE temporal features.

3 Principles of Time modelling with MARTE

We adopt the recent UML-MARTE profile and particularly its timing model [15] to capture the various forms of repetitive event that trigger the functional parts of an automotive embedded system, and to model them at a high level of abstraction. As it is shown in section 3.1, the rotation of the camshaft is the main trigger of the ignition control system. In the initial requirements of the ignition control system, the value of the period and the deadline are expressed with the unit camshaft degree.
The semantic attached to these multiform time (clocks, period, deadline, jitters…) makes it possible to transform this high level model of time to the real time.

3.1 Clocks as model elements

In MARTE, time is modelled with clocks. A `clock` is an instance of a `clockType` (see Figure 1) which has multiple attributes such as the nature of the clock (discrete or dense), the unitType that is an enumeration of the different units, and the relationships between them. For instance, the relationship between *s* and *ms* is a conversion factor of 0.001. The boolean attribute isLogical indicates whether the clock is endowed with a classical unit such as *chronometric* unit (closed to the classical notion of time -UTC time) or with *logical* units (concepts of time related to logical events). A `clockType` has also resolAttrib, maxValAttrib and offsetAttrib attributes. The resolution indicates the granularity of the clock (minimal interval between two values). The maximal value corresponds to a modulo-value and the offset denotes a temporal shift at the first instant. The different operations (*getTime*, *setTime*, *indexToValue*) are defined to access the clock.

Figure 1: ClockType stereotype in MARTE

A clock with a logical unit may represent any repetitive logical event in a system. Intervals between two consecutive occurrences of these events denote the instants (ticks) of the clock. Intervals are not necessarily equal. A duration represents the number of ticks between two events linked to this clock.
An example of such a logical clock is the camshaft revolution in an automotive engine. The different instants of this clock are linked to teeth detection onto the axis. The temporal distance between two consecutive teeth may vary and depends on the

engine rotation speed. This means that a logical clock is not necessarily periodic; nevertheless, it is possible to quantify a duration based on this clock, which represents the duration between two ticks on this clock. The unitType of this clock is the angle degree.
The Figure 2 shows an example of a time model with two *clockTypes*: MyIdealClockType (chronometric) and AngleClock (logical) and their *UnitKind* (MyTimeUnitKind and AngleUnitKind). Three instances of clocks are defined in the model (crkClk, camClk and myIdealClock) with different properties. We consider that the clock myIdealClock inherits, from its type, the default values for the resolution and offset. The crkClk and camClk have the same clockType but the maxValue attribute for their periods are different. One tick on camClk corresponds to two ticks on crkClk.

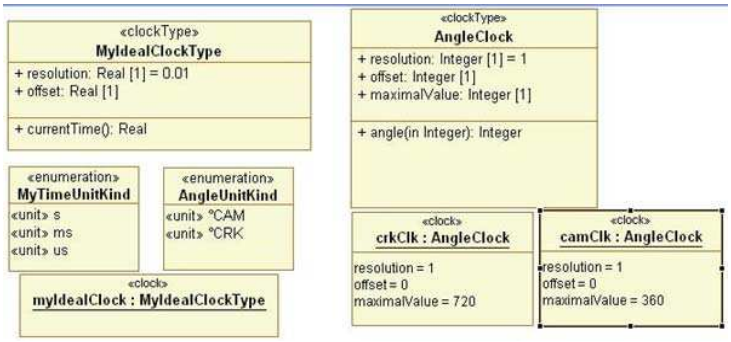

Figure 2: Elements of the Time Model

3.2 Relations and constraints between Clocks

As multiple heterogeneous clocks may be necessary in a model, MARTE allows the expression of relations between them. These relations are expressed by the way of `clockConstraint` expressions. In the MARTE profile, a specific language called CCSL (Clock Constraint Specification Language) [16] has been defined to express the different relationships between clocks.
From these constraint expressions, an automatic analysis based on partial order calculus can be applied that provides a solution for merging the clocks onto a common on considered as the reference. We call this phase the refinement of clocks that consist in the "projection" of the instants of logical and heterogeneous clocks onto a common one. In our example the relation between the crkClk and camClk clocks is given by the following CCSL expression:

```
camClk = crkClk filteredby 0b(10)
```

This means that camClk is a subclock of crkClk, and that the ticks of the camClk are coincident with the ticks of the crkClk when applying the binary mask 10 as a filter.

For a given application, the analysis of a set of clock constraints provides a solution, if this solution exists, of the interleaving of these clocks.
Clock constraints can be also used simply to transform a duration from one unit to another one. In the case of the crkClk, the relation between a second (s) and a °CRK depends on the rotation speed of the Crankshaft (Rotation Per Minute). Equation 1 expresses this relation.

$$1_{\circ CRK} = \frac{1}{RPM \times 6} s \,. \qquad (1)$$

3.3 UML Behaviour modelling using Clocks

A MARTE model containing clocks makes it possible to associate instant (occurrences) to events, or intervals (durations) to behaviours of a UML model. To achieve this goal, UML behaviours can be stereotyped with `timedProcessing`.

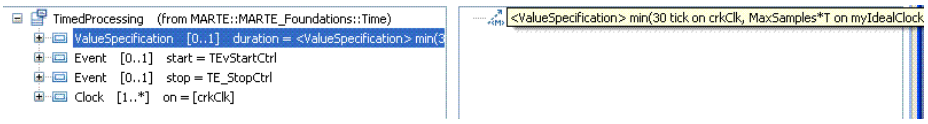

Figure 3: timedProcessing attributes

A timedProcessing may be any UML behaviour such as an activity diagram, a sequence diagram, a state machine and is linked to a clock (*on* attribute).
The main effect of such association is the possibility offered to the designer for expressing temporal characteristics and constraints onto behaviours. In MARTE these constraints are *timedValueSpecification* which represents either a duration or an interval, in between which, the behaviour should execute.
The Figure 3 shows the attributes of a TimedProcessing stereotype. The *clock* attribute indicates the reference clock on which the durations and events are measured. Two significant *timedEvents* can be associated with the start and stop of this behaviour. The *start* attribute indicates the event that triggers the behaviour. The *stop* attribute is an event produced by the behaviour at the end of its execution. This event materializes the duration of the activity and can be used in a *timedValueSpecification* to express a temporal constraint.
Of course, multiple factors influence the temporal distance between these two events. An advantage of using these events is the possibility of reasoning on the end of a behaviour independently to its physical execution support.
A *timedValueSpecification* expresses a constraint onto the activity duration. Notice that different clocks can be involved in a duration expression. In the equation 2 extracted from the paper [17], the duration is expressed as follows:

$$\text{duration} \prec MIN\left(\begin{array}{l}\text{30 tick on crkClk,}\\ \text{MaxSamples* T on MyIdealClock}\end{array}\right) \qquad (2)$$

The duration depends on two values measured onto two different clocks. Duration must be less than the minimum between 30 tick measured onto the crkClk clock and MaxSamples*T tick measured onto myIdealClock. MaxSamples is a variable of the application specification.
This equation can be solved after integrating the clocks constraints as explained in the paragraph 3.2. In this case, it consists in translating the crkClk unit to the second unit.

3.4 Illustration on an ignition control system

We illustrate the MARTE Time Model usage on the example of an automotive engine control system. We focus on a particular part of this control that is the correction of the ignition. The activity diagram on Figure 4 shows that the ignition of the spark plug in an engine depends on different corrections imposed by physical phenomenon such as the knock, the temperature variation and the warm-up of the engine. Multiple sensors and actuators participate to this correction. The correction controllers must be executed periodically and their executions may overlap. The period unit is the crkClk clock. The period, the offset and the deadline of these actions are also linked to this logical time base.

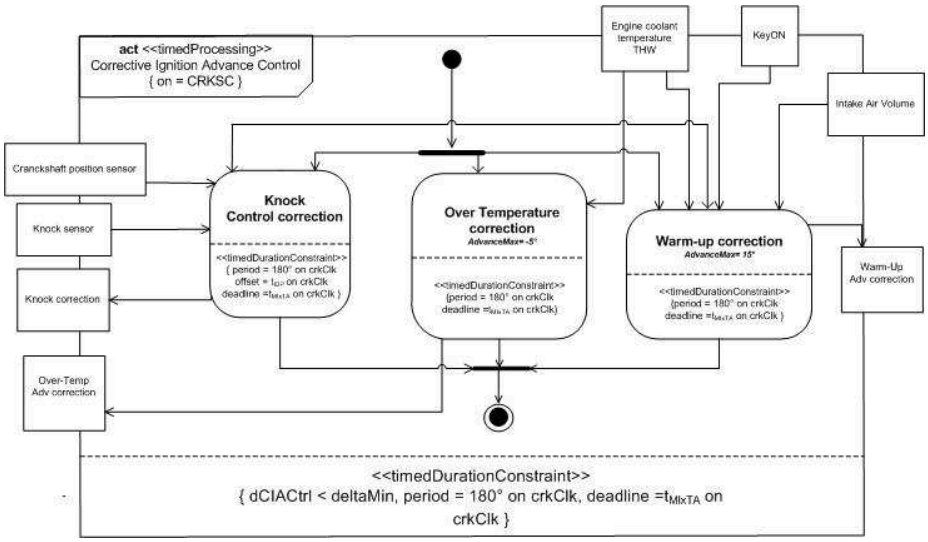

Figure 4: Correction activity of a control engine system

We associate to each correction controller a temporal characteristic such as its period and a timing constraint such as a deadline. For the sake of visibility we have represented these constraints at the bottom parts of the actions and the activity. In an actual model using tools, these constraints are modelled with timedValueSpecification expressions which are elements of the design.
A consequence of the activity diagram hierarchy is that time constraints can be expressed at the different levels of this hierarchy. Thus, time constraints at the lower levels must comply with those of the higher level.

A verification of such constraints can be done by applying mathematical rules and reasoning on a high level of abstraction of time. At this level we can obtain results about the feasibility of this control according to the different constraints and some other characteristics of the system.
Examples of the feasibility results are: all the correction controls can be executed on time in the case of a 4-stroke engine with a maximum engine rotation speed of 4000 rpm but cannot be executed if the rotation speed exceeds this value. These results, detailed in the paper [17], were obtained by a manual calculus applied to functions executing sequentially (non preemptive cyclic scheduling). On the example presented figure 4 multi-task scheduling is intended. For that purpose the tasks has been enriched with the temporal characteristics requested for this type of analysis.

4 From logical to physical real-time

4.1 Clock refinement

The modelling of multiple heterogeneous clocks in the time model allows a high level modelling of temporal aspects of a system. The goal is to apply scheduling analysis onto these models. Such analysis requires two conditions. The first one is, at the behavioural modelling level, the necessity to cope with a common notion of time. The second condition is the integration of the execution platform model and the allocation constraints in order to integrate the physical time (the one of the CPU processor).
The paragraph 4.2 addresses this second aspect. Concerning the common notion of time, it is obtained at the behavioural model level by the integration of temporal constraints. Equation 1 gives the relation between the crkClk and the idealClock (s). The parameter *RPM* represents the engine rotation speed. This parameter depends on the automotive phases (acceleration, deceleration …). To evaluate the schedulability of the different tasks, we have to consider the worst case execution time corresponding to the maximum value of the RPM (MAXRPM=4500). By considering this value, we may use the clock constraint expressed in Equation 1 to transform the period values which time unit is the crankshaft degree °CRK to period expressed in second. The Table 1 gives the result of such transformation.

4.2 Integration of the execution platform characteristics

In a MDE design process, the execution platform model is built separately from the application model. The MARTE profile provides a set of resources model elements that supports the modelling of the application. The association between an application model and an execution platform model is obtained through an allocation model.
Figure 5 shows a structural description of the architecture of the ECU on which the engine controller will be executed. The execution platform is modelled as a set of microcontrollers and a memory; all the elements are interconnected by a bus.

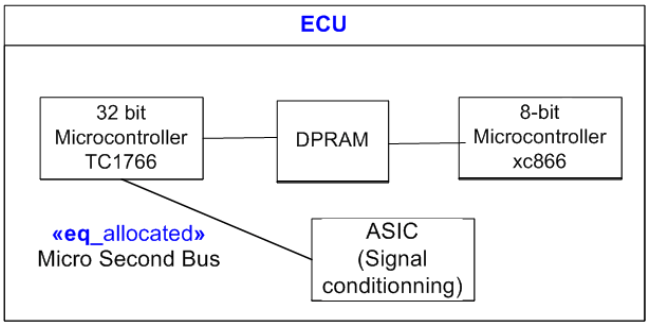

Figure 5: ECU execution platform

The execution platform can be annotated with temporal information based on logical or chronometric clocks. Figure 6 provides a timed-view of the different elements of the execution platform. A computation resource can be seen as a clock with a resolution that corresponds to the duration for executing one instruction cycle on this processor. As the Infineon TC1766 is a PCP2 (single cycle instruction), it means that one instruction on this processor takes a duration of 125 10^{-10} second. The temporal characteristic of a bus is generally its transmission rate.

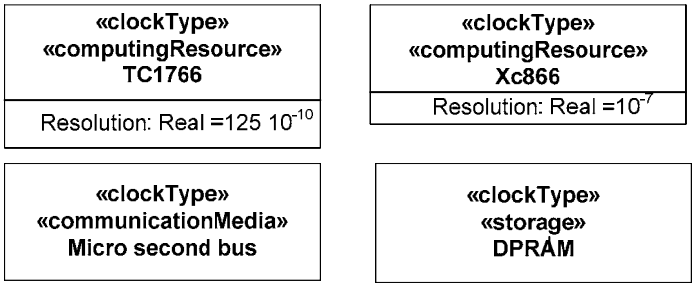

Figure 6: Time model of hardware

This abstract view of the execution platform can be enriched with others similar non-functional characteristics such as the memory cost, the power consumption. Such model allows a verification of the execution platform capabilities in regard to the constraints and the characteristics of the application parts.

4.3 Allocation of functionalities onto execution platform resources

The association between the application model and the execution platform model is expressed by the allocation model. An allocation associates every element of the application model to a resource of the execution platform model. In MARTE allocations, are annotated with time constraints and temporal characteristics. The allocation modelling establishes the link between the model views of time and the actual physical time.

Allocations can map structural to structural elements, and behavioural to behavioural or structural elements. The Figure 7 represents the potential allocations of sensors, actuators and actions of Figure 4 onto the possible computation resources of the ECU. The sensors and actuators behavioural parts are allocated to the ASIC

(signalConditioning). The functions are allocated to the Infineon TC1766. In some cases, a behavioural part may have several possible allocations onto different resources. The allocation of a function onto a physical resource implies a temporal cost (the WCET). This cost is a new information that appears on the allocation diagram at the bottom part of each action.

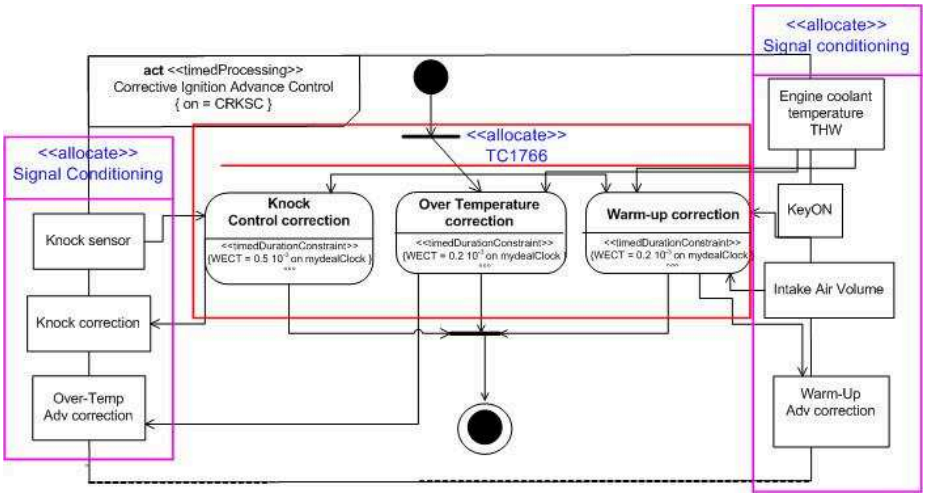

Figure 7: Partition of activities

Table 1 is a summary of the different temporal characteristics of the behavioural parts. The period, offset and deadline values have been extracted from the behavioural models.
The period expressed initially with the time unit °CRK has been translated in second by applying the time constraint relation on equation 1.
The offset and deadline were expressed as relative dates in the behavioural model with the time unit °CRK. We give in Table 1 the actual values of these parameters obtained by calculating the corresponding values in milliseconds. The last row of the table represents the duration of the deadline which depends on the offset and the deadline date.

	Knock	**Over Temp**	**Wam-up**
WCET (on TC1766 in ms)	0,5	0,2	0,2
Period(°CRK - *ms*)	180 - *6,66*	180 - *6,66*	180 - *6,66*
Offset date (°CRK - *ms*)	24 – *0,888*	0 - *0*	0 - *0*
Deadline date (°CRK / *ms*)	50 – *1,851*	50 – *1,851*	50 – *1,851*
Deadline duration (°CRK / *ms*)	26 – *0,962*	50 – *1,851*	50 – *1,851*

Table 1: Temporal information associated to tasks

The WCETs are obtained either by emulating or profiling the tasks corresponding to the correction controllers represented in the behavioural model. The time base in this case is the time base of the processor on which these tasks will execute, i.e. the

physical clock. The value of the deadline has been extracted from the book [18] which gives some actual parameters values of an ignition engine controller.
The next step consists in exploiting these characteristics in the scheduling analysis phase.

5 Scheduling analysis

5.1 Principles

The scheduling analysis may start as soon as the application models with its associated timing model, and the architecture model, are available. Mainly, it consists in exploiting the timing information, i.e. the temporal characteristics (periods and WCETs, the latter depending on the allocation) as well as the timing constraints (deadlines), attached to each temporally characterized function of the application model. The allocation model allows the designer to determine the actual timing characteristics which vary with the various possible computing resources each application element can be allocated to. Multiple potential allocations can be considered for a function when several computing resources are able to implement it. In our case the computing resource is unique since we address uniprocessor architecture. Nevertheless several versions of this unique processor may be considered. In this case the schedulability analysis must be iterated according to the different processors which induce different timing values of the WCET.
The processor of the execution platform is supposed to provide a RTOS, e.g. OSEK in the case of our example. This RTOS is the standard for the automotive domain. The schedulability analysis assumes also that the given RTOS supports the scheduling policy the analysis is based on. This will guarantee that the real-time behaviour of the application, running on the chosen architecture, will satisfy the real-time constraints.
Actually, this assumes that the WCET were carefully determined and enough margins were taken to approximate the cost of the RTOS itself, i.e. the cost of the scheduler including the cost of the preemption if it is allowed by the scheduling policy.

5.2 Illustration on an ignition controller

In the ignition control system example, the three behaviours: Knock control correction, Over temperature correction, and Warm-up correction are associated to a couple of temporal characteristics (deadline, period), and a WECT that was determined according to the execution platform resources it was allocated to. These associations lead to a system of three periodic tasks: Knock, OverTemp, and Warm-up. Their periods initially expressed in °CRK have been translated in milliseconds (from the idealClock) by applying the time constraints between the two clocks. The resulting values are listed in Table 1.

With these data, it is possible to perform a scheduling analysis based on a fixed priority scheduling policy. Every tasks is periodic, has a WCET, and a deadline. In addition, preemption is allowed. As the deadline of each tasks is less than its periods, the system of tasks can be scheduled according to the Deadline Monotonic (DM) scheduling algorithm [19], i.e. the task with the smallest deadline has the highest priority, assuming these tasks are independent. If they are not independent a more complex schedulability analysis must be performed, but that does not change anything to the proposed approach. We do not focus on the schedulability analysis itself but on the way it is possible to perform it from the previous models, manually or possibly automatically. In this context, the scheduling analysis using DM algorithm amounts to verify the following sufficient condition. As mentioned before, to be consistent the RTOS running on the considered uniprocessor must use also this algorithm.
The system is schedulable if:

$$\sum_{i=1}^{n} \frac{WCET_i}{Deadline_i} \leq n\left(2^{\frac{1}{n}} - 1\right) \text{ with } Deadline_i \leq Period_i \qquad (3)$$

We chose here the DM fixed priority policy instead of the Earliest Deadline First (EDF) variable priority policy because the scheduler is simpler, and thus its cost is easier to approximate. This approximation is a fundamental hypothesis used in the DM schedulability analysis. Usually the industrial designers take a margin equal up to 30% of the task WCET. With these assumptions our automotive example with the three tasks mentioned above is schedulable if:

$$\frac{WCET_{KC}}{deadline_{KC}} + \frac{WCET_{OT}}{deadline_{OT}} + \frac{WCET_{WU}}{deadline_{WU}} \leq 3(2^{\frac{1}{3}} - 1) = 0.779 \qquad (4)$$

Considering the values of Table 1 we can conclude that this system of three tasks is schedulable with the assumption of a MaxRPM equal to 4500. In this case, the left part of the equation is equal to 0.735 < 0.779. On the other hand, if we consider a MaxRPM equal to 6000, the left part of the equation is equal to 0.980 > 0.779 and the system of tasks is not schedulable.
Another constraint to be verified is the timedValueSpecification of the activity diagram (CorrectionAdvanceControl). According to the timedDurationConstraints expressed on Figure 4 the equation 5 must be verified.

$$t_{IDP} + WCET_{KC} + WCET_{OT} + WCET_{WU} \leq t_{MIxTA} \qquad (5)$$

This equation is also valid.

6 Conclusion

In order to cope with the complexity of real-time embedded systems, the Model Based Design approach promotes a separation of concerns between the model of the application (functions) and the model of the execution platform.
UML and its profiles are largely used to model both parts. The recent standardization of the UML MARTE profile extends UML with temporal information and physical resources modelling capabilities. Applying this profile to a model based design makes it possible to enrich the application and execution platform models with precise and semantically well founded temporal information. As this information corresponds to explicit model elements endowed with a clear semantic, they can be extracted from the model. Consequently, MARTE models can be used as a starting point for methods and tools intended for schedulability analysis. They take into account temporal information and timing constraints for verifying deadline constraints.
In this paper, we presented the MARTE model elements associated with the time model package of MARTE, and we illustrated their uses on an automotive case study. Four models were presented, the *functional* model, the *time* model, the *allocation* model, and the *execution platform* model. While temporal information (periods) and constraints (deadlines) are associated with the functional model and are independent from the execution platform model, other timing information (WCET) are dependent of the platform, i.e. the computing resources.
In this approach, there are two notions of time, the time linked to the functional model logical time and the physical time related to both the allocation model and the execution platform model. We showed how to establish the link between logical time and physical time through the allocation model.
From these models we extracted the physical timing information and used them to straightforwardly perform a schedulability analysis. We use an analysis based on the DM algorithm, but others types of analyses are possible, which concludes that the illustrative application, characterized with the temporal characteristics and constraints, is schedulable onto the given execution platform.

References

1 MARTE OMG Specification. *A UML Profile for MARTE, Beta 1*.OMG Adopted specification ptc/07-08-04. August 2007.
2 OMG. *UML 2.1 Superstructure Specification*, April 2006.OMG document number: ptc/2006-04-02.
3 OMG. *UML Profile for Schedulability, Performance, andTime Specification*, January 2005. OMG document number: formal/05-01-02 (v1.1).
4 *Papyrus*: graphical UML2 modelling editor, http://papyrusuml.org
5 *Cheddar*: http://beru.univ.brest.fr/~singhoff/cheddar
6 *SynDEx*: http://www-rocq.inria.fr/syndex/
7 T. Schattkowsky, W. Muller, *Model-based design of embedded systems*, IEEE symposium on Object-Oriented real-time distributed computing, , pp113-128, Vienna May 2004
8 ITEA project. *EAST-ADL: The EAST-EEA Architecture Description Language*, June 2004. ITEA Project Version 1.02.

9 *Autosar* specifications, www.autosar.org
10 H. Gronniger, J. Hartmann, H. Krahn, S. Kriebel, L. Rothhardt, B. Rumpe, *View-centric Modeling of Automotive Logical Architecture*, Tagungsband des Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme IV. Informatik-Bericht 2008-02, CFG-Fakultät, TU Braunschweig, 2008.
11 O Sokolky, I. Lee and D. Clarke, *Schedulability analysis of AADL models*, 14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'06), Island of Rhodes, Greece, April25-26, 2006.
12 ATESST, *ATESST project reports of the Advancing Traffic Efficiency and Safety thought Software Technology* (ATESST) project. *http://www.atesst.org*.Final versions, 2007
13 *TIMMO* project, www.timmo.org
14 Oliver Scheickl, Michael Rudorfer , *Automotive Real-Time Development Using Timing-augmented AUTOSAR Specification*, BMW Car IT, Proc. of the 4th European Congress Embedded Real-Time Software (ERTS 2008), Toulouse, France, January 29 - February 1, 2008
15 Ch. André, F. Mallet, and R. de Simone, *Modeling Time(s)*. In MoDELS'07, 10th ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems. LNCS 4735, 2007.
16 *CCSL Clock Constraint Specification Language*, MARTE OMG Specification [1] Annex C3 pp419-430.
17 F. Mallet, C. André M-A. Peraldi-Frati *Multiform time in UML for Real-Time Embedded Applications*, 13th IEEE Inter. Conf. on Embedded and Real-Time Computing Systems and Applications, Daegu (Korea) August 2007.
18 R. Bosch GmbH, *Automotive Handbook*, 7th edition, Bentley Publishers
19 C.L. Liu and J.W. Layland, *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*, Journal of the ACM, Vol. 20, No. 1, pp. 46-61, january 1973