# Extending UML/MARTE-SAM for Integrating Adaptation Mechanisms in Scheduling View

Mohamed Naija and Samir Ben Ahmed
*Laboratory of Computer for Industrial Systems, INSAT, Tunis, Tunisia*

Keywords: Adaptability, Scheduling Analysis, Real-time & Embedded Systems, MDE, MARTE.

Abstract: The profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) defines a framework for annotating non-functional properties of embedded systems. In particular, the SAM (Schedulability Analysis Model) sub-profile offers stereotypes for annotating UML models with the needed information which will be extracted to fulfil a scheduling phase. However, SAM does not allow designers to specify data to be used in the context of adaptive systems development. It is in this context that we propose an extension for the MARTE profile, and especially the sub-profile Schedulability Analysis Modeling, to include adaptation mechanisms in scheduling view. We illustrate the advantages and effectiveness of our proposal by modeling a FESTO case study as an Adaptive Real-Time and Embedded system.

## 1 INTRODUCTION

The modeling of Real-time & Embedded Systems (RTES) may be stated as a crucial problem in the software engineering domain. RTES are subject to a multitude of constraints (e.g., battery, temperature …) and real-time requirements. Thus, designers are encountering the challenge of resource limitations, time, highly variable environment, etc. The addition of adaptivity to such systems further hardens and delays their modelling and scheduling analysis especially with the current lack of design models and tools for adaptive RTES. Lightening the task of adaptive systems designers and reducing the development cost and time to market represent a major challenge in the field, which requires the use of high-level approaches such as MDE (Schmidt, 2006) and MARTE (OMG, 2008).

MDE is a way to beat the growing complexity of real time systems and verifying their correctness. In particular, Unified Modeling Language (UML) profiles promote an adequate solution to support the whole lifecycle co-design of complex systems. In RTES domain, its adoption is seen promising for several purposes: requirements specification, behavioral and architectural modeling with their real time constraints and performance issues. In this context, the profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) fosters the building of models that support the specification of scheduling analysis problem. This profile has the capacity to model tasks, dependencies between them and events under shape a Workload Behavior of system. Subsequently, it promotes the validation of the system temporal accuracy. Unfortunately, MARTE does not define a clear semantics for modeling and analysis of the adaptation in RTES.

Thus, we propose in this paper the main changes to be made on MARTE profile for supporting adaptation mechanisms. These amendments affect mainly the stereotypes of MARTE/SAM (Scheduling Analysis Modeling) since it is the sub-profile intended to model the schedulability analysis.

Our contribution is to improve the meta-models of the existing annotations. We try to modify in the structure of existing annotations, with no need to add new ones, by referring to their cardinality which makes easy the adoption of the proposed extension in revision of MARTE. This work is the result of previous investigations and published work. Starting from (Naija et al., 2015) a MARTE-based approach was proposed to concurrency model construction at early design stages. In (Naija and Ben Ahmed, 2015) a reconfiguration solution for RTES to meet performance constraints is developed. Notably, we perceive a MARTE semantics limitation in modeling level. Accordingly, we have identified the needed of the new version of the MARTE profile supporting adaptation mechanisms.

This work is to be integrated in a model-based



Naija M. and Ahmed S.
Extending UML/MARTE-SAM for Integrating Adaptation Mechanisms in Scheduling View.
DOI: 10.5220/0005822300840090
In *Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering (ENASE 2016)*, pages 84-90
ISBN: 978-989-758-189-2

approach to guide RTES designers for building and analysing adaptive RTES models. It facilitates complex systems modeling, reduces the development time and cost and improves software process quality. The above benefits have been illustrated through the application of our extensions to different examples of adaptive RTES.

The present paper is organized as follows. Section 2, emphasizes the various related works, section 3 introduces the concept of the adaptation. While section 4 gives a brief definition of the MDE paradigm, its MARTE profile and the SAM sub-profile, section 5 specifies our proposal. To better explain our contribution which is highlighted in section 6, we rely on a case study. Finally, section 7 concludes the paper and sketches some future work.

## 2 RELATED WORK

The design of adaptive RTES presents many challenges due to the complexity of the problem it handles (Said et al., 2014). In the present paper, we limit our study to research works particularly tackling adaptive RTES using the MARTE profile.

Many researchers have benefited from the MARTE profile for the design and verification of adaptive RTES from high-level models. In (Cherif et al., 2010) authors have benefited from MARTE to model reconfigurable architectures such as FPGAs based Systems-on-Chip (SOC). They extended the MARTE profile with some semantics and Xilinx specific concepts, which limits their applicability for diverse systems, to support Dynamic and Partial Reconfiguration (DPR) of FPGA. Unlike this contribution, we aim to propose a new extension to support adaptation which is independent from any specific platform.

In (Boukhanoufa et al., 2010) the authors give a classification of 13 publications that have dealt with the subject of adaptation in the design approach. Following this classification, the authors illustrate using an avionic example the need for the validation of adaptation rules at design-time according to the real-time features of the system. In this context of verification approaches, they have proposed in (Boukhanoufa et al., 2011) an MDE approach for modeling and offline validation of application timing constraints. In fact, this article uses state machine to represent the application configurations and transitions between them to represent adaptation rules. This work is based on the generation of all possible configurations of a system before running, in order to validate timing constraints. The number of configurations varies from one system to another and it can be very large, this combinatorial explosion makes the timing analysis inapplicable.

In (Said et al., 2014), five patterns have been proposed to model and evaluate adaptation. These design patterns are presented in a static form through class diagrams and stereotyped MARTE profile. The adaptive behavior of the system is then designed according to the proposed design pattern without supporting schedulability analysis.

Two major scheduling approaches are available in the literature (Magdich et al., 2014): the partitioned and the global approaches. Originally, MARTE supports only the modeling of the systems to be scheduled according to the partitioned approach. In (Magdich et al., 2012) the authors have proposed various updates for MARTE meta-models of specialization and generalization stereotype in order to support global scheduling approaches, allowing task migrations. Those changes allow a schedulable resource to be executed on different computing resources in the same period (Rubini et al., 2014). Unfortunately adaptation is considered only as a dynamic change in the task allocation, without taking into account operating mode adaptation which is an essential axe of adaptation.

## 3 THE ADAPTABILITY DEFINITION

There are several definitions of adaptation in the literature. In (Bihari, 1991), a software adaptation is defined as any software modification that changes the reliability or timeliness of the software without affecting other aspects of its functionality. Software adaptation encompasses many common software-tuning techniques. These include:

- resource reallocation, such as moving a software component from one processor to another,
- adjustments to processor schedules,
- modification of replication factors for N-modular-redundant software components, and
- modification of retry limits or time-out periods for delivery of a service by a software component.

In (Subramanian and Chung, 2000) and (Lehman and Ramil, 2000), adaptation means change in the system to accommodate change in its environment. More specifically, the adaptation of a software system (S) is caused by a change (Che) from an old environment (E) to a new environment (E’), and results in a new system (S’) that ideally meets the needs of its new environment (E'). Formally, adaptation can be viewed as a function:

Adaptation: E × E' × S → S', where meet (S', need (E')).

In (Oreizy et al., 1999) adaptive system is defined as a system that is able to change its structure or behavior at run-time in response to the execution context variations and according to adaptation engine decisions.

In the present paper, adaptation is defined as any modification in the structure, behavior or architecture of the system to accommodate external or internal change of their operating environment or context and according to predefined adaptation plan and rules.

We distinguish two types of reconfigurations; static reconfigurations (Angelov and Marian., 2005) and dynamic reconfigurations (Khalgui and Hanisch, 2011). Since static adaptation requires stopping the system and restarting it with a new configuration. The dynamic adaptation is based on a set of predefined adaptation behavior, statically designed and verified at design time. At runtime, the system can select an alternative from the available ones in accordance with adaptation rules and context.

# 4 MDE AND RTES DEVELOPMENT

The Model Driven Engineering (MDE) is a software development methodology aiming to increase the level of development and overcome the growing complexity challenge. It covers the entire systems lifecycle, simplifies the design process by using the concept of models and offers independency between different steps of development flow.

In the context of the schedulability analysis, MDE is mainly used in the modeling step and the transformation of scheduling analysis models to the models of the chosen scheduling analysis tool (Magdich et al., 2012). MDE uses the UML profile and especially MARTE.

## 4.1 MARTE Capabilities for RTES Modeling

MARTE is an extension of UML profile providing support for specification, modeling and verification step of real time and embedded systems. MARTE supports the modeling of software and hardware features at a high-level of abstraction. In addition, it offers a rich set of annotations for modeling schedulability analysis. This profile encompasses a lot of sub-profiles such as: SRM (Software Resource Modeling), HRM (Hardware Re-source Modeling), GQAM (Generic Quantitative Analysis Modeling), SAM (Schedulability Analysis Modeling), PAM (Performance analysis Modeling), etc. In this paper, we will focus especially on the SAM sub-profile since it is the package affected by our proposal.

## 4.2 SAM

In order to establish an early validation of the system's temporal behavior a well-formed analyzable model, called SAM, is defined. This profile has the capacity to model tasks, dependencies between them and events. It has the capacity to predict if all tasks meet their time constraints by defining the workload behavior. This is a chain of operation activations representing executions scenarios for the application.

# 5 SAM EXTENSION FOR A BETTER SCHEDULING ANALYSIS MODELING STEP

The scheduling analysis modeling is performed through the MARTE/SAM profile. The idea of performing scheduling analysis based on MARTE models assumes that all the information that is needed for the analysis is already part of the MARTE model (Naija et al., 2015). In fact, SAM meta-model supports the modeling of different systems as it models all the temporal features needed in the scheduling step except those used to model adaptation constraints. Consequently, we seek to improve SAM meta-model in order to support modeling and early analysis of adaptation process. The amendments to be done on the SAM sub-profile affects also the GQAM sub-profile since some classes of the sub-profile SAM inherit from GQAM sub-profile.

## 5.1 The Workload Behavior

The end-to-end computation represents the processing load of the system. It represents the different steps (operations) executed in the system and triggered by one or more external stimulus. Each step can be linked to a successor step with a control flow. «*SaStep*» is a stereotype annotating an action/operation and contains both timing description (computational budget, activation event, etc.) and timing constraints (operation deadlines). A set of steps specify the so-called *Schedulable Resources*. These are units of execution taken into account by the scheduler of the system, called tasks in scheduling literature (Radermacher et al., 2010).

## 5.2 Amendments to be Done for the Workload Event

The workload behavior of the system (OMG, 2008) is characterized by their workload events and behavior scenarios. Workload events annotating *UML AcceptEventActions* introduce the semantic of event sequence arrivals for the execution of each *callBehaviorAction*. Originally event triggers only one behavior scenario. When adaptation is required, an event triggers all the behavior scenarios of the system workload. For example, let us imagine that for an adaptive event that denoted a low level of battery, this can affect more than one behavior scenario. Thus, the designer is not able to specify this property.

Hence, we propose to modify in the association linking the two classes *WorkloadEvent* and *BehaviorScenario* of the package GQAM Resources (Fig 1 and Fig 2). The multiplicity [1..*] denotes that an event can affect one or more behaviour scenarios. Indeed, GQAM is a generalization of the package SAM. So, this change will be inherited by SAM.

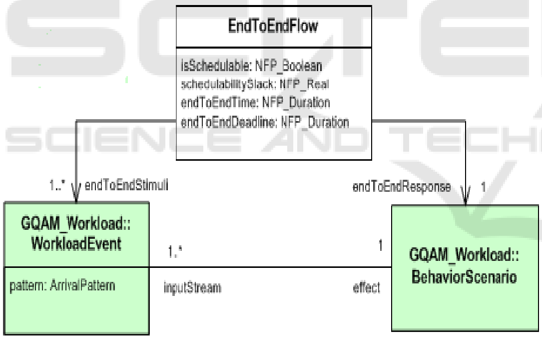


Figure 1: The old meta-model of the GQAM package.

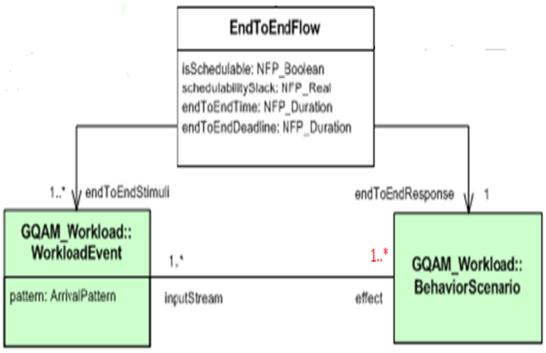


Figure 2: The new meta-model of the GQAM package.

## 5.3 Amendments to be Done for the Step

The class *Step* may represent a small segment of code execution (OMG, 2008). It contains a lot of attributes specifying the temporal features of software resources. In UML MARTE model step can be part of only one behavior. Otherwise, in the case of adaptation process a new *Step* can appear in multiple behavior scenarios to manage adaptation. Thus, MARTE/SAM doesn't allow this specification. Thereby, we propose to change in the association linking the two classes *BehaviorScenario* and *Step* (Fig 4).

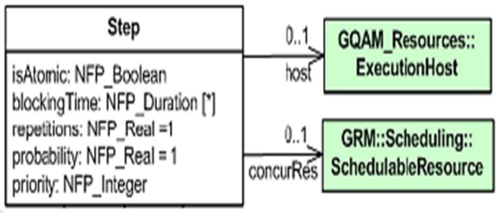


Figure 3: The old meta-model of the SAM package.

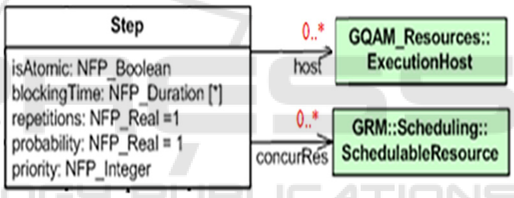


Figure 4: The new meta-model of the SAM package.

Once the workload behavior is performed, it is necessary to identify the so-called *schedulable resources* (called tasks in scheduling literature). Schedulable resources are defined by mapping the execution of the end-to-end computations to them, in order to generate the task model. Different types of mapping exist in the literature (Masse et al., 2003) (Mraidha et al., 2011) (Naija et al., 2015). As explained previously, a software resource can be mapped into more than one thread. Accordingly, the cardinality of the association between *Step* and *SchedulableResource* must be [0..*] (Fig 4).

## 5.4 Amendments to Model Tasks Migration

In literature, three scheduling approaches are presented: the partitioned, the semi-partitioned and the global approaches (Muhammad et al., 2011). Regarding the partitioned approach, it affects each task to be executed on one processor. Accordingly, tasks are not allowed to migrate between processors (Magdich et al., 2014). CPU utilization is therefore not optimal. As for the global approach and semi-partitioned, they enable a tasks migration such that schedulable resource may be allocated, not simultaneously, on different computing resources. The task migration is considered as an adaptive technique that allows improving application performance and achieves optimality. Currently, MARTE/SAM supports only the partitioned approach. Thus, task that can be across multiple processors for different periods of time is not permitted in SAM. Subsequently, the multiplicity of the attribute corresponding to the execution resource (*ExecutionHost*) on which a task (*schedulableResource*) is allocated must be [0..*] instead of [0..1] (Fig 5 and Fig 6). This extension is adopted from the research work (Magdich, 2013).

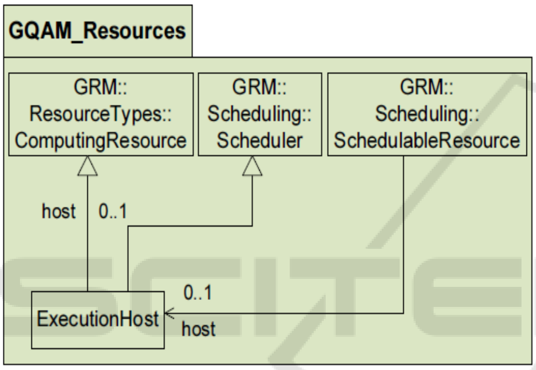


Figure 5: Meta-model of the GQAM package (Magdich, 2013).

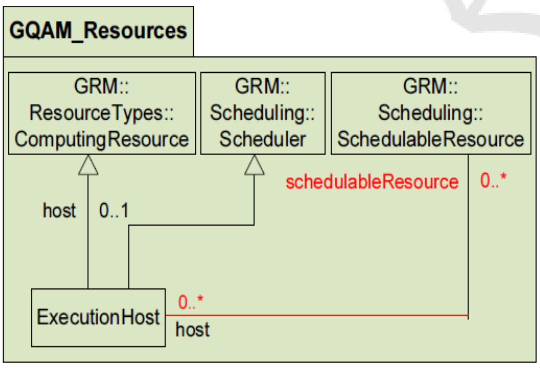


Figure 6: Meta-model of the GQAM package with amendments (Magdich, 2013).

While migrating from one processor to another, the execution time of a task is not the same, then the attribute «deadline» of the stereotype *SaStep* should have a multiplicity of [0..*]. In the same vein, a task can be interrupted several times during one period. Consequently the attribute «preemptT», which refers to the length of time that the step is preempted, must have a multiplicity [0..*] instead of [0..1]. Similarly for the attribute «readyT» which indicate length of time since the beginning of a period. Hence, this attribute must have a multiplicity of [0..*]. The set of values for the attributes «deadline», «preemptT» and «readyT» must be ordered (Fig 7 and Fig 8). This extension is adopted from the research work (Magdich, 2012).

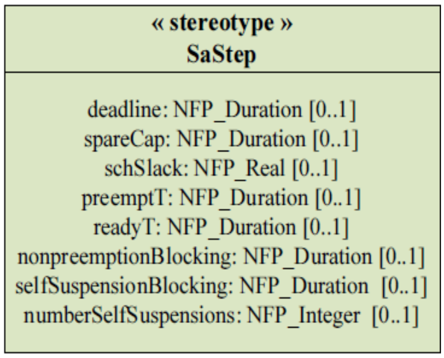


Figure 7: Meta-model of the GQAM package (Magdich, 2012).

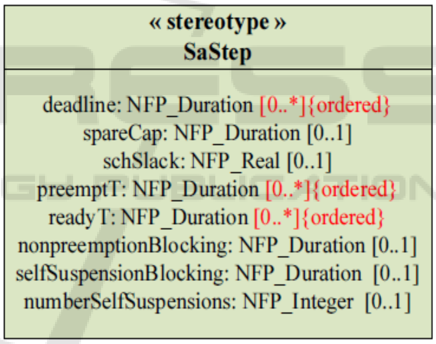


Figure 8: Meta-model of the GQAM package with amendments (Magdich, 2012).

# 6 CASE STUDY

To better explain our proposal, we use a FESTO (Khalgui and Hanisch, 2011) production system as an intact running application in this paper. It is a well-documented laboratory system used by many universities for research and education purposes.

The working process of FESTO is composed of three units: the distribution unit, the test unit, and the processing unit. The distribution unit consists of two steps: a pneumatic feeder and a converter. It forwards cylindrical workpieces from a stack to the testing unit. The test unit consists of three steps: the detector, the tester, and the evacuator. It performs the checking of workpieces for their height, material type, and color. Workpieces that pass the test unit successfully are forwarded to the rotating disk of the processing unit, where the drilling of workpieces is done. The result of the drilling operation is next checked by a checker and finally the finished product is removed from the system by an evacuator.

Note that in this work two drilling machines *Drill1* and *Drill2* are used to drill workpieces. *Drill1* is used in case of medium production. When high production is required, *Drill2* is recommended. According to user requirements, the system FESTO is able to reconfigure automatically at run-time in response to any changed working environment caused by errors or new requirements to improve system performance without a halt. The workload behaviour in Fig 9 represents the processing load of the system, founded on our proposal.

After identifying the behavior model of the system, it is necessary to specify the so-called *schedulable Resources*. For sake of simplicity, we use in this paper the scenario-based mapping (Masse et al., 2003) which is also one of the most used. The idea is to regroup all the operations executed at the same rate and belonging to the same linear end-to-end computation to the same task. In our FESTO system, we obtain three different threads namely task1 (*pieceEjection*, *Convert*, *Test* and *Evacuate*), task2 (*pieceEjection*, *Convert*, *Test*, *Elevate*, *Rotate*, *Drill1*, *Checker* and *Evacuate*) and task3 (*pieceEjection*, *Convert*, *Test*, *Elevate*, *Rotate*, *Drill2*, *Checker* and *Evacuate*).

Following this case study, founded on our proposal. We illustrate that an event (e.g., *epieceEjection*) triggers all tasks. The steps *Convert* and *Test* are part of three tasks. Similar for steps *Elevate*, *Rotate*, *Checker* and *Evacuate*, those participate for the execution of both schedulable resources Task2 and Task3. Compared to original version of MARTE, specifying these properties is not permitted. Note that, tasks can have dynamic properties (*readyT*, *preemptT* and *deadlines*) due to the concept of task migration, but we use the same corresponding values to facilitate our example. Anyway, we can add the different values and they will be ordered to perform scheduling analysis. After scheduling all tasks, we can specify in our SAM view the used allocations through the attributes *Host: GaExecHost* and the corresponding attribute *ExecT: NFP_Duration*.

## 7 CONCLUSIONS

In this paper, we have proposed an extension to MARTE profile since it does not allow modeling and analysis of adaptive systems, which is judged as a hard engineering task. This improvement affects mainly the sub-profile MARTE/SAM and especially



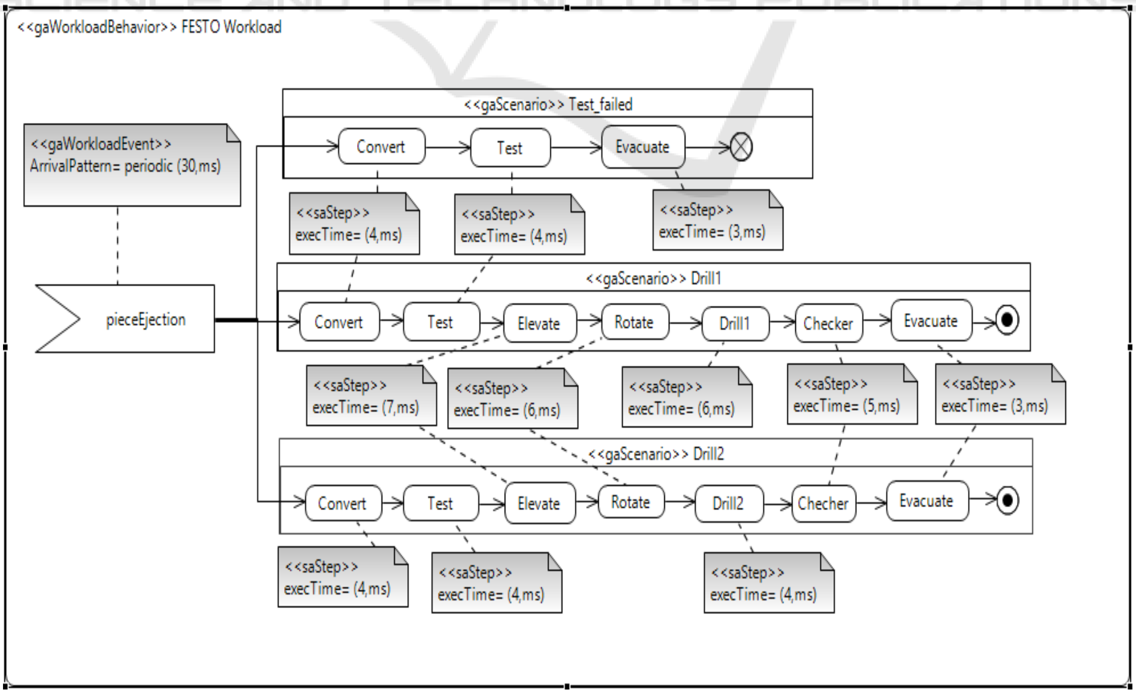


Figure 9: The workload behavior of FESTO system.

GQAM/SAM. This contribution makes MARTE able to stand adaptability at early design stages of RTES. The benefit of our approach is the ability to model adaptive properties which will be extracted, to serve during the scheduling step. Our proposal has already been performed on the papyrus tool, which is an editor of MARTE-based modeling, and validated through a case study.

As future work, we will investigate in analysing and validating of dynamic adaptability, in accordance with adaptation rules and context, at early design stages. This verification step further reduces the development risks of Adaptive RTES.

## REFERENCES


Angelov C, and Marian, N., 2005. Design models for reusable and reconfigurable state machines. In *Proc of Embedded Ubiquitous Comput*.

Bihari T-E et Schwan K., 1991. Dynamic adaptation of real-time software. *ACMTrans. Comput. Syst*., vol. 9, pp. 143-174.

Boukhanoufa M-L, Radermacher A and Terrier F., 2010. Towards a model-driven engineering approach for developing adaptive real-time embedded systems. In *New Technologies of Distributed Systems*, pp 261-266.

Boukhanoufa M-L, Radermacher A and Terrier F., 2011. Offline validation of real-time application constraints considering adaptation rules. In *international Conference on Trust, Security and Privacy in Computing and Communications*, pp 974–980.

Cherif S, Quadri I.R, Meftali S, and Dekeyser J-L., 2010. Modeling reconfigurable Systems-on-Chips with UML MARTE profile: an exploratory analysis. *Proceedings of the 13th Euromicro Conference on Digital System Design*, pp 706-713.

Khalgui M and Hanisch H. M., 2011. Automatic NCES-based specification and SESA-based verification of feasible control components in benchmark production systems. In *international Journal of Modelling, Identification and Control*, vol. 12, no. 3, pp. 223–243.

Magdich A,. Kacem Y. H, Mahfoudhi A., 2013. Extending UML/MARTE-GRM for Integrating Tasks Migrations in Class Diagrams. In *international Conference on Software Engineering Research, Management and Applications*, pp73-84.

Magdich A,. Kacem Y. H, Mahfoudhi A, and Abid M., 2012. A MARTE extension for global scheduling analysis of multiprocessor systems. In *international Symposium on Software Reliability Engineering*, pp 371–379.

Magdich A., Kacem Y. H., and Kerboeuf M., 2014. A uml/marte-based design pattern for semi-partitioned scheduling analysis. In *international Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, pp. 300–305.

Masse, J., Kim, S., and Hong, S. 2003. Tool Set Implementation for Scenario-based Multithreading of UML-RT Models and Experimental Validation. In *RTAS' 03*, *Proceedings of the* 9th IEEE *Real-Time and Embedded Technology and Applications Symposium*. IEEE Computer Society.

Mraidha, C., Tucci-Piergiovanni, S. and Gerard, S., 2011. Optimum: a marte-based methodology for schedulability analysis at early design stages. In *ACM SIGSOFT Software Engineering Notes*, 36(1):1–8.

Muhammad K.B., Cécile B., and Michel A., 2011. Two level hierarchical scheduling algorithm for real-time multiprocessor systems. *Journal of software*, vol. 6, no. 11, pp. 2308–2320.

Naija M, Ahmed B.S. and, Bruel J-M., 2015. New schedulability analysis for real-time systems based on MDE and petri nets model at early design stages. In *ICSOFT'15. 10th International Conference on Software Engineering and Applications*, pp 330-338. 2015.

Naija M, Ahmed B.S., 2015. Aid to design and reconfiguration of the MPSOC architectures. In *CIE'15. IEEE Computer Society 45 th International Conference on Computers & Industrial Engineering*.

Lehman M and Ramil J., 2000. Towards a Theory of Software Evolution _ And its practical impact (working paper). Invited Talk, In *Proceedings International Symposium on Principles of Softw. Evolution*, pp 2-11.

OMG Object Management Group. 2008. A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems, Beta2, Object Management Group.

Oreizy P, Gorlick M. M., Taylor R. N. et al., 1999. An architecture based approach to self-adaptive software. In *IEEE Intelligent Systems and Their Applications*, vol. 14, no. 3, pp. 54–62.

Radermacher, A., Mraidha, C., Tucci-Piergiovanni, S. and Gérard, S., 2010. Generation of schedulable real-time component implementations. In *ETFA'10, 15th IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE Computer Society.

Rubini S, Fotsing C, Dissaux P, Singhoff F, and Tran H. N., 2014. Scheduling analysis from architectural models of embedded multi-processor systems. *ACM SIGBED Review*, vol. 11, no. 1.

Said M, Kacem Y.M, M. Kerboeuf M, Amor N.B. and Abid M., 2014. Design patterns for self-adaptive RTE systems specification. In *International Journal of Reconfigurable Computing*, vol., no. 8.

Schmidt D-C. Model-driven engineering. 2006. IEEE Computer, 39(2).

Subramanian N, Chung L., 2000. Architecture – Driven Embedded Systems Adaptation for Supporting Vocabulary Evolution. Proceedings of International Symposium Principles of Software Evolution International, vol. 0.