Towards a Model-based Multi-Objective Optimization Approach For Safety-Critical Real-Time Systems

Soulimane Kamni*, Yassine Ouhammou*, Antoine Bertout†, and Emmanuel Grolleau*
LIAS, *ISAE-ENSMA and †Université de Poitiers, Futuroscope, France,
*{soulimane.kamni, yassine.ouhammou, grolleau}@ensma.fr and †antoine.bertout@univ-poitiers.fr

***Abstract*—In safety-critical real-time systems domain, obtaining the appropriate operational model which meets the temporal (e.g. deadlines) and business (e.g. redundancy) requirements while being optimal in terms of several metrics is a primordial process in the design life-cycle. Recently, several researches have proposed to explore cross-domain trade-offs for a higher behaviour performance. Indeed, this process represents the first step in the deployment phase, which is very sensitive because it could be error-prone and time consuming.**

This paper is a work in progress proposing an approach aiming to help real-time system architects to take benefit from existing works, overcome their limits, and capitalize the efforts. Furthermore, the approach is based on the model-driven engineering paradigm and suggests to ease the usage of methods and tools thanks to repositories gathering them as a sort of a shared knowledge.

***Index Terms*—Critical real-time systems, model-driven Engineering, multi-objective optimization, functional-to-architectural mapping, real-time analysis tools.**

I. Introduction

A real-time system consists of a set of parallel and interdependent functions that should be performed on one or several execution nodes in order to accomplish sensing, control and/or command functionalities. The parallelism is often ensured by the multitasking paradigm. Safety-critical real-time systems are widely used in various domains (like avionics, automotive and nuclear), where temporal constraints should be respected (e.g., deadline, end-to-end delays), otherwise a catastrophic failure may occur. For this reason, safety-critical real-time systems require to apply appropriate temporal validation tests to check their schedulability.

The design phase of such systems can take place over several years while designers have to cope with hardware and software specificities. Recently, several works have been proposed to ease the modeling and the analysis of critical real-time systems by proposing approaches, tools and frameworks [10] based on model-driven engineering settings [16]. Indeed, these works allow designers to master the system complexity by shaping the practical system as a set of abstract models. However, on the one hand, most of existing works focus on advanced design phases, especially by capturing the operational part. This latter consists in mapping a functional model (i.e., functions and input-output data) to the workload model (i.e., tasks and messages), scheduled through a real-time operation system (RTOS) and deployed on an execution platform (i.e. hardware model). On the other hand, in a model-driven engineering context, the refinement of the functional model of a real-time application to a specific operational model is a challenging task. This can be error-prone and time consuming if no automated support is provided. Hence, several mapping methods have been proposed in the literature [1] relying on optimization methods. Generally, each method leads to an optimal mapping functional-to-operational solution in order to satisfy one or several kinds of metrics while ensuring the schedulability. Examples of these metrics can be the number of processors, transmission buses, memory consumption, the number of threads, preemption costs, energy, price, memory size, etc. In other words, the optimization of such metrics should enhance the flexibility of the system while ensuring its behaviour correctness. In fact, the limit of the current optimization-based mapping techniques is that they focus on one or several specific metrics with an inter-dependency relationship. Moreover, each of the proposed mapping methods relies on a particular functional pattern (e.g., periodic events, independent functions, precedence relationships,). Thus, the design process becomes more complicated and confusing for real-time architects since each technique solves a particular problem differently. That is, the designer has to master each method separately. The contribution of this work-in-progress is to suggest a blueprint of a model-based framework dedicated to the multi-objective optimization for critical real-time systems. This framework consists in providing a support gathering different mapping methods and easing their usage without being expert in the optimization field. Indeed, this support should help system architects to choose the appropriate methods that match both functional requirements and metrics requiring to be optimized.

The rest of this paper is organized as follows. We present the state of the art and our work positioning in Section II. Section III introduces an overview of our approach and details future directions. Finally, Section IV discusses and concludes this paper.

This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement N. 826610

II. State of the art

As depicted in Figure 1, the real-time system design is generally composed of three parts in the context of model-based design [15].

The functional model consists of functions, their activation patterns, their time budgets, their precedence relationships and also constraints that should be respected. Among these constraints we can find scheduling-related ones (like end-to-end delay) and business ones (for instance, for safety reason two functions should be segregated). The behaviour of the functional part may be represented as directed acyclic graph where vertices correspond to the system function-set $F = \{f_0, f_1, ..., f_n\}$ and edges correspond to the interaction-set $I = \{i_0, i_1, ..., i_m\}$ between functions.

The hardware model consists of a set of execution processors, their core kinds (e.g., unicore or multicore), their provider/frequency kinds (e.g., homogeneous, uniform or heterogeneous cores), and a set of network buses, their topology, their bandwidth and their protocols (like CAN, AFDX, AVB, etc.).

The third part is the operational model which is, in fact, the first result of the deployment process. The operational model is a result of the function-to-task mapping, the task-to-core allocation, the data-to-message mapping, tasks deadlines derived from the functional end-to-end delays, tasks activation patterns derived from functions activation and synchronization, etc. Moreover, the real-time operating system (RTOS) is a cornerstone choice since it impacts the task scheduling (e.g., online, offline, hierarchical) and several properties including priority assignments, preemption costs and shared resource access.

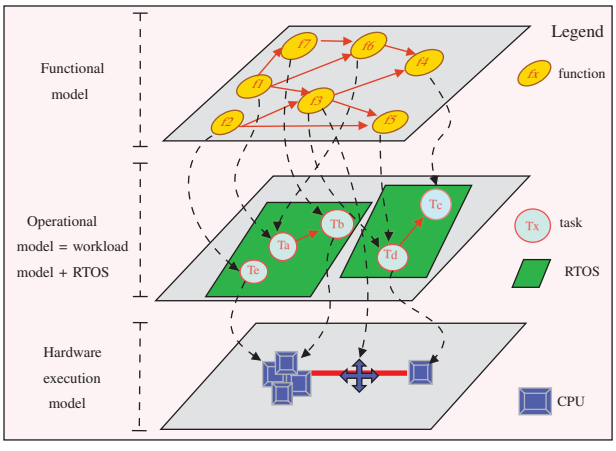

Fig. 1. Real-time system architecture

The deployment of the real-time systems consists of mapping the functional architecture to an operational architecture. Indeed, mapping manually the functions to tasks is time-consuming and may be subject to errors in regards to the significant number of possibilities. In the following, we present existing works regarding the deployment process and we highlight their limits and our work positioning.

A. Background and related work

Instinctively, two naive extreme mapping strategies are identified: i) the single-task that consists of executing the whole functional specification in one and only one task, ii) one-to-one mapping, where one task is defined for each functional block. Both strategies suffer from several drawbacks. While the single-thread approach suffers from delays in input/output procedures, the one-to-one approach suffers from the excessive scheduler overheads due to the frequent context switching and complex two-ways communication mechanisms. Therefore, the system designers need a trade-off solution in order to reduce the impact due to the drawbacks of these two extreme strategies. In the literature, different alternatives exist in order to solve the optimization problems related to the mapping of functions to tasks.

Bartolini et al. [2] have proposed an ad-hoc solution for mapping functions to tasks targeting uniprocessor architecture using Earliest Deadline First (EDF) as a scheduling policy [7]. Their approach consists of generating a unique workload model from a functional model with calculating the timing properties. The authors verify the schedulability of the unique proposed solution using the processor-demand analysis [9].

Mehiaoui et al. [11] have addressed distributed architectures with heterogeneous execution nodes which adopt the fixed priority scheduling policy. Their approach proposes a two-step formulation based on integer linear programming. The first step consists in placing the functions on the execution nodes, while in the second step the functions are assigned to tasks. The resulting tasks are scheduled eventually at the end of the second step. Here, the problem is formulated as a set of linear constraints with an objective function that can be solved afterward using a solver. The authors address hence two objectives, the minimization of both the number of tasks and the worst case end-to-end response times of the external events. Moreover, the schedulability test opted to validate the candidate solutions is the response time analysis [17].

Solving the linear programming problem provides an optimal solution. However, in case of problems with a large number of possibilities and numerous constraints, the solution does not scale. Thus, another way could be possible to solve the optimization problem is via approximate approaches such as heuristics. So, Mehiaoui et al. [11] have tackled the two-step problem using a genetic algorithm in order to overcome the unscalable linear programming solution drawback.

Bertout et al. [4] have suggested to cluster functions into tasks while preserving the schedulability of the system. The authors aim at minimizing the number of tasks in order to reduce the context switching overheads through the exploration of a well-bounded search space. This work is intended for Deadline Monotonic (DM) [3] and EDF scheduling policies on uniprocessor architectures. The worlkoad model considered in this study is composed of periodic and independent tasks. The problem has been solved via a heuristic of type best-first search since the problem was proved to be a NP-Hard. The schedulability test used in order to validate the candidate

solutions is the response time analysis [6].

In [5], the authors target uniprocessor architectures with Rate Monotonic (RM) scheduling policy. They have considered a functional model with periodic and independent functions. This work proposes a double objective optimization. It aims at minimizing the number of preemptions and maximizing the laxity (the amount of time between the worst-case response time of a task and its relative deadline). A meta-heuristic algorithm called Pareto Archived Evolutionary algorithm is used to solve the problem. A simulation-based test is performed on each given candidate solution to check the schedulability.

B. Discussion

The existing works mentioned previously share a common real-time aspect: schedulability of the system. The limit of existing solution is that they address a specific "**context**". By context we mean a specific kind of architecture with a specific scheduling policy and communication protocols and where their schedulability is checked by a specific schedulability analysis method. Moreover, the formalization of optimization methods proposed in these studies is also ad-hoc since it is related to a specific context and aims at optimizing specific metrics. Constraints to be addressed are also fixed: if a business specific constraint is to be taken, a designer would have to manually encode it. Therefore, one of the key objectives of our work is to categorize the different works. Hence, comes the idea of building an all-in-one flexible framework as a key solution to ease the deployment of the real-time systems and capitalize the efforts.

Actually, we are not the first to have thought of building a framework with the same objective of regrouping multiple optimization methods of the software architecture. Le Nabec et al. [8] have proposed a framework that helps to compare the results given by each optimization method as well as the outcome of combining consecutively different strategies for different metrics. The authors proposed an extensible library of deployment methods. However, this work has several limitations in regards to what we are working on. Hereafter we enumerate these limitations. (i) The framework does not support real-time standard languages as input formalisms, i.e. the inputs are not conform to languages such as Capella [14] or SysML [12]. (ii) The framework does not provide any mechanism to describe neither the schedulability-related nor business-related constraints. In contrast, constraints are constant (static) and there in no way to customize, introduce new ones or select the desired ones. (iii) Also, this work provides some existing optimization methods but only for a comparison purpose. It does not orient system architects to choose the right/appropriate methods in order to get the suitable architectural model.

Our mid-term objective is to allow the system architects to express the requirements and help them to: i) identify the optimization methods regarding the context of the optimization problem and the desired metrics, ii) take the right decision in choosing the best optimization method. Finally, it would help them to compare the outcome results and integrating them to prune the design space and to focus on the optimal solutions.

III. Our approach

Our vision behind this work is to provide a framework for multi-objective optimization based on model-driven engineering settings (i.e., meta-modeling, model transformation, code generation). The framework is intended specifically for system architects in order to help them in the elaboration of an optimized operational architecture. The framework would propose the operational model automatically simply after introducing the functional model.

A. Functioning process

The process will operate through two steps. The first step is dedicated to finding a design space of candidate solutions that meet constraints. The second step will be used as a pruning process to reduce the number of possibilities in order to keep only those that meet the metrics. So, as in input of the first step, we will have a real-time system S to be deployed $S =< f, C, H >$, where: f is the functional model, C a set of constraints and H hardware model. The set of constraints allows us to propose the mapping methods. The framework should propose an order relationship between constraints, hence when no solution exists, relaxing some constraints should be possible. Also, the set of constraints allow us to get the objective function. The goal of the objective function is to minimize or to maximize metrics. Yet, the objective function has to call the appropriate schedulability test to check the schedulability of a candidate solution. In order to exploit the mapping methods, a transformation toward the optimization formalism such as MILP, SAT, Heuristic-based or meta-heuristic based formalism should be proposed by the framework. The second step is dedicated to prune the design space. The pruning is driven by the metrics to optimize. Then, regarding the number of metrics to satisfy the design space will be reduced.

B. Framework components

Figure 2 gives an illustration of the framework. The framework will be composed of numerous components as we detail hereafter.

The first component is a repository that stores the optimization methods. It should group all the exiting optimization methods that exists in the literature and should be extensible for the future works. That is the repository should be open and dedicated to be fed in an incremental way by researchers proposing new methods. The second component is the front-end side of the framework (dedicated to system architects) which would present a graphical mechanism in order to express the functional model, plus the description of both schedulability-related constraints and business constraints. The functional specification would have to be conform to a real-time modeling language such as Capella, likewise for the design requirements, they would have been expressed to conform to a requirement modeling language which we plan to propose.

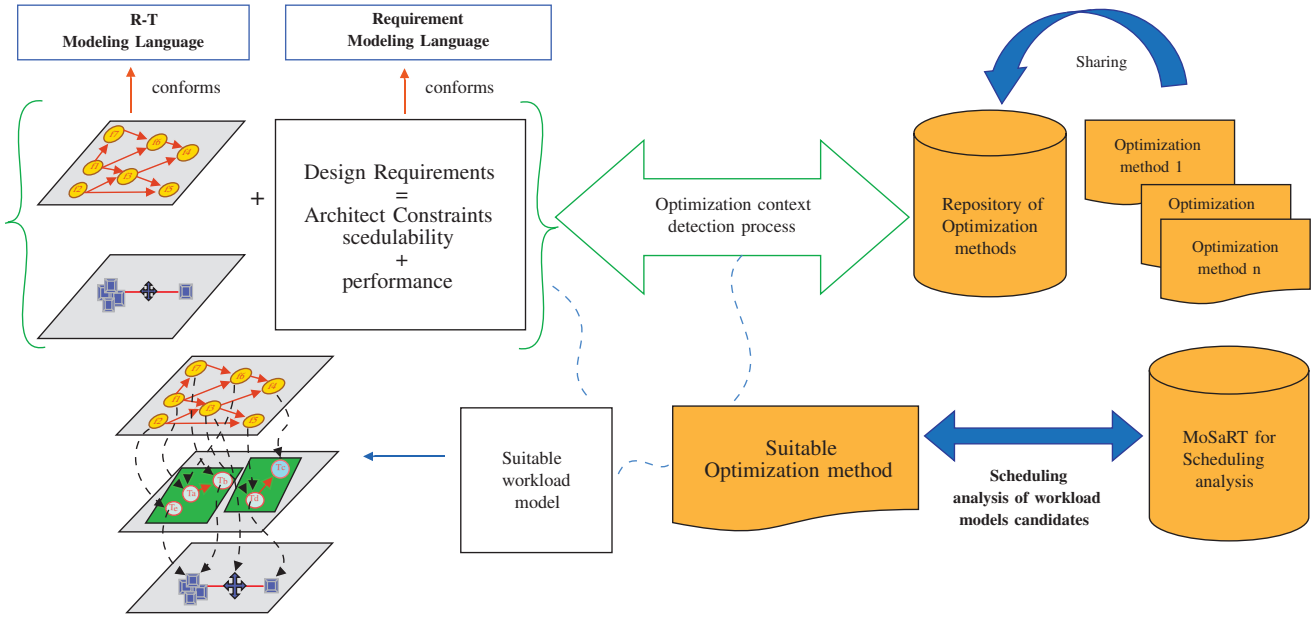

Fig. 2. global illustration of the framework

To ensure the automation of the optimization process, the framework would contain another component called "context detection process". This mechanism should automatically guide the architect to the right optimization methods. The context concerns the set of information provided on the hardware, the task scheduling policy, the functions model, the metrics to optimize. As mentioned above, the analysis tools exist already. However, a connection to these tools is necessary since all the modules must be communicating to ensure the automation of the whole process. The framework would be connected to use in a systematic way the scheduling analysis tools thanks to helper tools like MoSaRT [13], which allows to find the appropriate scheduling tests and get eventually the suitable workload model.

IV. Conclusion

This paper introduced a work in progress. It is a model-based framework helping to deploy a safety critical systems thanks a multi-objective optimization. This work would help the system architects getting an optimized operational model regarding their constraints to meet and metrics to optimize.

References

[1] A. Aleti, B. Buhnova, L. Grunske, A. Koziolek, and I. Meedeniya. Software architecture optimization methods: A systematic literature review. *IEEE Transactions on Software Engineering*, 39(5):658–683, 2013.

[2] C. Bartolini, G. Lipari, and M. Di Natale. From functional blocks to the synthesis of the architectual model in embedded real-time applications. In *RTAS 2005: 11th IEEE Real Time and Embedded Technology and Applications Symposium, Proceedings*, 2005.

[3] S. K. Baruah. Efficient computation of response time bounds for preemptive uniprocessor deadline monotonic scheduling. *Real-Time Systems*, 47(6):517–533, 2011.

[4] A. Bertout, J. Forget, and R. Olejnik. Minimizing a real-time task set through task clustering. In *Proceedings of the 22nd International Conference on Real-Time Networks and Systems*, page 23. ACM, 2014.

[5] R. Bouaziz, L. Lemarchand, F. Singhoff, B. Zalila, and M. Jmaiel. Architecture Exploration of Real-Time Systems Based on Multi-objective Optimization. *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, 2016-Janua:1–10, 2016.

[6] M. Joseph and P. Pandya. Finding response times in a real-time system. *The Computer Journal*, 29(5):390–395, 1986.

[7] M. Kargahi and A. Movaghar. A method for performance analysis of earliest-deadline-first scheduling policy. *The Journal of Supercomputing*, 37(2):197–222, 2006.

[8] B. Le Nabec, B. B. Hedia, and J. P. Babau. QuaRTOS-DSE: A tool for design space exploration of embedded real-Time system. *Proceedings - 2018 IEEE 21st International Symposium on Real-Time Computing, ISORC 2018*, pages 42–50, 2018.

[9] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, 1973.

[10] A. T. B. Long, Y. Ouhammou, and E. Grolleau. Leveraging real-time network analyses by extending a model-based framework. *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*, 2017-October:871–878, 2018.

[11] A. Mehiaoui, E. Wozniak, S. Tucci-Piergiovanni, C. Mraidha, M. Di Natale, H. Zeng, J.-P. Babau, L. Lemarchand, and S. Gerard. A two-step optimization technique for functions placement, partitioning, and priority assignment in distributed systems. *ACM SIGPLAN Notices*, 48(5), 2013.

[12] Object Management Group. Systems Modeling Language. https://www.omg.org/spec/SysML/, 2017.

[13] Y. Ouhammou, E. Grolleau, M. Richard, P. Richard, and F. Madiot. Mosart framework: a collaborative tool for modeling and analyzing embedded real-time systems. In *Complex Systems Design & Management*, pages 283–295. Springer, 2015.

[14] Polarsys. Capella: open source solution for model-based systems engineering. https://www.polarsys.org/capella/.

[15] A. L. Sangiovanni-Vincentelli and M. D. Natale. Embedded system design for automotive applications. *IEEE Computer*, 40(10):42–51, 2007.

[16] D. C. Schmidt. Model-driven engineering. *COMPUTER-IEEE COMPUTER SOCIETY-*, 39(2):25, 2006.

[17] K. Tindell and J. Clark. Holistic schedulability analysis for distributed hard real-time systems. *Microprocessing and Microprogramming*, 40(2):117 – 134, 1994. Parallel Processing in Embedded Real-time Systems.