High Level Design of Adaptive Real-time Embedded Systems
A Survey

Mouna Ben Said, Yessine Hadj Kacem, Nader Ben Amor and Mohamed Abid
University of Sfax, ENIS, CES Laboratory, Soukra km 3.5, B.P. 1173-3000 Sfax, Tunisia

Keywords: Adaptation, MDE, UML/MARTE, RTES, Timing Constraints.

Abstract: Real time embedded systems (RTES) know a growing complexity due to modern applications requirements and architectures complexity especially with the addition of the multiprocessor feature. They are equally subject to a variety of constraints due to their mobility. They need to react to environment variability, resource limitations and timing constraints. An emergent solution to deal with this complexity is the integration of adaptation strategies in embedded systems design flow. The design space of multi-layer adaptation decisions is becoming increasingly vast and difficult to explore. Development of such systems at low system levels is therefore increasingly tedious especially with the limitations of computer aided design tools. Using MDE approach and the UML/MARTE profile for high abstraction level design is becoming a promising solution to ease the design of RTES. In the present paper we recall and classify existing works built around adaptive embedded systems. We concentrate on a set of criteria to help highlighting the shortages of existing approaches on modern adaptive RTES design. We focus on the design environment, adaptation features, online time constraints verification and performance assessment. Finally, we present our future works to cope with the limits of existing solutions while taking into account the observed criteria.

1 INTRODUCTION

Compared to conventional desktop and server systems, real-time embedded systems are often subject to numerous constraints due to their mobility. These include (i) limitations on local resources, imposed by weight and size constraints, concern for power consumption, and lowered trust and robustness resulting from exposure and motion, and (ii) environment variations such as unpredictable variation of network load. This makes it impossible to yield an offline optimization. Thus, RTES are asked to dynamically self-adapt internally to their limited computational and energy resources as well as to changes in their surrounding environment in order to offer the required quality of service. Self-adaptive systems change their behavior or structure, at run-time, in response to variations in their execution environment and according to adaptation engine decisions. Researchers have focused on the integration of adaptation strategies in embedded systems which further hardens designers' task. Another complexity dimension is added to such systems' development through their prominent evolution to increasingly complex multiprocessor architectures. In this way, their design space containing adaptation decisions at different system layers (architecture, application, operationg system and network) is becoming increasingly vast and difficult to explore, especially with the limitations of computer aided design tools. Therefore, developing such complex systems using classic design approaches, which handle technical details at low system levels, is no more an efficient solution.

There have been numerous contributions in adaptation for RTES at both hardware and software (operating system, application) layers. However, there is a lack of standard formalization of the adaptation logic that is understandable by everyone, platform-independent and reusable. At present, there are great improvements of high level design methods based on Model Driven Engineering (MDE) methodology (Schmidt, 2006) and specifically using the UML/MARTE (Modeling and Analysis of Real-Time and Embedded Systems) profile (Group, 2011). Authors in (Gogniat et al., 2010) assert the strong need for abstraction mechanisms to cope with the limits of existing design flows and associated tools that are too tied to hardware platforms and permit designers to develop more efficient systems. Model based approaches permit to decrease the complexity of modern systems using the abstraction mechanism while increasing productivity and reusability and minimizing

Ben Said M., Hadj Kacem Y., Ben Amor N. and Abid M..
High Level Design of Adaptive Real-time Embedded Systems - A Survey.
DOI: 10.5220/0004381703410350
In *Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development* (MODA-2013), pages 341-350
ISBN: 978-989-8565-42-6

cost and Time to Market. The MARTE profile permits to jointly model both hardware and software parts of RTES which eases the communication between designers and reduces design errors. It allows, as well, performance and schedulability analysis. However, existing MDE based adaptation approaches do not deal with the complete process of adaptation in RTES. They do not offer, equally, a complete and obvious support of timing constraints guarantee which is a key feature in RTES. Additionally, MARTE profile does not fully support adaptation features for RTES such as inter-processor tasks migration.

In the present paper, we recall and classify state-of-the-art approaches around adaptive embedded systems. While studying these approaches, we concentrate on a set of comparison criteria related to design environment, adaptation process and support of online temporal and performance analysis. In Section 2, we start by defining the considered comparison criteria. In section 3, we recall low-level adaptation approaches which do not consider a modeling step. Then, in Section 4, we look through adaptation approaches based on MDE paradigm. In Section 5, according to the comparison criteria, we discuss and compare related works in order to summarize the limits of existing approaches on MDE based design of adaptive RTES. Then we present our proposed solution and future works to cope with these limits and allow taking the adaptation behavior into account early in RTES design flow. Finally, we conclude in Section 6.

2 COMPARISON CRITERIA

During our study, we focus on a number of criteria in which we are interested for our ongoing work. The first group of criteria relates to the design environment. We are interested to mention if the approach is designed at a high level. We are particularly concerned with the UML/MARTE standard which fits our future works direction (please refer to section 4.1 on MDE/MARTE). We are also interested in platform dependency and multiprocessor support criteria. The second group of criteria is linked to the adaptation process where we are interested in the modularity of design and cross-layer adaptation support. We finally focus on the support of online temporal verification and performance evaluation.

There is no need to recall the importance and benefits of modularity in complex systems development, especially in terms of scalability and productivity. The modular structure of embedded system adaptation process includes:

- Observation: needs to specify resources (ex. CPU time, network bandwidth or power consumption observer), trigger events and associated thresholds, and monitoring period.
- Analysis: generates adaptation requests according to upcoming events. The used analysis test per constraint has to be defined (such as scheduling analysis for real time constraint) as well as thresholds to demand adaptation action.
- Decision: needs the specification of a set of adaptation strategies and mechanisms, system configurations and selection algorithms to be used to generate the adaptation solution that best meets the adaptation requests.
- Action: acts on the system elements to run the new adaptation solution. It is generally application/platform dependent.
- Assessment: required to measure adaptation cost-effectiveness and make adjustments if needed to meet the required performances. Metrics have to be defined, and feedback, estimation and update techniques need to be settled. Adaptation frequency and granularity, and system stability need to be considered. Performance models describe how to measure and interpret application performance.

3 LOW-LEVEL ADAPTATION APPROACHES FOR EMBEDDED SYSTEMS

There have been numerous contributions to integrate adaptation strategies in embedded systems development process. In the present section, we recall low-level adaptation approaches in different layers of embedded systems. These approaches do not consider a modeling step in their development process. We classify those works in two classes: hardware layer and software (operating system and application) layer adaptation.

3.1 Adaptation in the Hardware Layer

We present in the following sub-sections a number of low-level approaches proposing adaptation techniques which are representative of common state-of-the-art adaptation techniques.

3.1.1 Power Management Techniques

Portable systems are mostly battery driven and oftentimes have to run for considerable time periods. The

most serious limitation on these devices is the available battery lifetime. Therefore power management has become a critical issue for those systems. Diverse efforts have been made to improve power management in mobile devices. Dynamic Voltage scaling (DVS) (Pillai and Shin, 2001) and Dynamic Power Management (DPM) techniques have been widely studied as methods for optimizing the power consumption.

3.1.2 Dynamic Hardware Reconfiguration as a Key Adaptation Feature

Profiting from technological advances, the current trends show that dynamically reconfigurable architectures are considered a key element in self adaptive embedded systems development. Researchers in the domain have focused their works on the development of new hardware adaptation techniques taking advantage of dynamic and partial reconfiguration (DPR) capability offered by modern FPGA (e.g. Xilinx devices). The DPR support enables the system to swap several designs by reconfiguring co-processors or accelerators at run-time. Such feature is beneficial as it permits to deal with system optimization at run-time and improve performance. However, it has a negative effect on surface and energy consumption. So, to address area, power and performance trade-offs, a solution is to decide when and how a running application has to be accelerated. In (Ye et al., 2010), authors aim at optimizing multi-processor architecture through HW resources reconfiguration according to variable applications' needs in terms of standard functions. In fact, authors have noticed a prominent use of common standard functions in embedded systems such as video/image processing and data encryption. Their approach is based on a reconfigurable MPSoC architecture model and libraries of hardware and software implementations of intensively used standard functions. They define a reconfiguration decision algorithm which decides at run-time which configuration best fits with application requirements according to recorded execution time.

3.2 Adaptation in Software Layers

We classify software adaptation in two categories: adaptation in the OS layer and application-aware adaptation.

3.2.1 Adaptation in the OS Layer

The role of the OS in an embedded real-time adaptive system is to sense external events, monitor and allocate scarce resources. Generally, OS adaptation consists in changing resources allocation or modifying the scheduling policy in response to application and resource variations. In (Vahdat et al., 2000), CPU resource managers are defined to provide soft real-time performance guarantees. In (Banachowski and Brandt, 2002), schedulers adapt the scheduling policy to handle the variations of application at runtime. K. Nahrstedt et al. (Yuan and Nahrstedt, 2006; Chu and Nahrstedt, 1999) have equally dealt with scheduling policy adaptation. They have proposed in (Yuan and Nahrstedt, 2006) an energy-efficient soft real-time CPU scheduler for multimedia applications running on a mobile device. The EScheduler seeks to minimize the total energy consumed by the device while meeting multimedia timing requirements. To achieve this goal, they integrated the DVS into traditional soft real-time CPU scheduling: it decides at what CPU speed to execute applications in addition to when to execute what applications. EScheduler makes these scheduling decisions based on the probability distribution of cycle demand of multimedia applications and obtains their demand distribution via online profiling.

An important allocation adapting technique, which is increasingly used in modern systems with multiprocessor architecture, is inter-processor tasks migration (Goossens et al., 2003). It consists in reallocating tasks on execution units following a resource variation event. The event can be a workload variation due to a task entry or exit event in order to respect timing constraints. This is a classic problem often encountered and extensively studied in distributed systems such as web servers or local networks. It is however not well tackled in embedded systems especially in the case of heterogeneous multiprocessor architectures. Different strategies can be envisaged. The simplest is to execute the most complex tasks on the most efficient processors. The operating system to be used has to manage processors heterogeneity issue which presents a significant challenge when the processors have different instruction sets.

3.2.2 Adaptation in the Application Layer

Modern applications, specifically multimedia ones, are becoming more and more complex and have increased computational demands that may exceed existing embedded systems capacity. Application-aware adaptation is then an important capability of embedded systems that has been largely tackled in the literature. Application level adaptation techniques have been essentially developed for multimedia systems since they fit very well multimedia applications, such as H264 and JPEG2000 codecs, which are highly

configurable. It is hard to conceive a universal algorithm that can perform well for all kinds of contents. However, if important characteristics of applications can be identified and utilized to trade-off output quality for resource usage and user preferences (ex. Trade-off output video quality for network bandwidth availability), one can design an adjustable algorithm that can tune its parameters (algorithms or configuration parameters) to adapt to environment variations. This tuning technique has been widely used in literature (Said et al., 2011) (Mesarina and Turner, 2003) (Ngoc et al., 2002) (Hsia, 2003) (Satyanarayanan et al., 1995). Mesarina and Y. proposed in (Mesarina and Turner, 2003) a method to reduce the energy for a specific multimedia application, the «decoding MPEG», using parameter modifications. Authors in (Ngoc et al., 2002) presents a QoS framework for interactive 3D applications where the QoS management relies on high-level QoS parameters (e.g. PSNR) of quality scalable 3D objects. This framework aims at guaranteeing the user specified interactive frame rate through degrading the 3D objects quality in such a way that minimal overall quality degradation over the scene is obtained.

3.3 Cross-layer Adaptation

The above adaptation techniques have been shown to be effective for both QoS provision and energy saving. However, most of them adapt only a single layer or two joint layers. More recently, researchers have proposed cross-layer adaptation frameworks where different system layers adaptations are coordinated in order to fully exploit the adaptation benefits (Yuan et al., 2006) (Vardhan et al., 2009) (Diguet et al., 2011) (Loukil et al., 2009). Some of these cross-layer approaches adapt only at coarse time granularity (Pillai et al., 2003)(Mohapatra and Venkatasubramanian, 2003), e.g., when an application joins or leaves the system. This infrequent adaptation is proven to be insufficient to deal with small changes in the system environment and processed data. Then, some other cross-layer adaptive frameworks have been proposed to adapt systems at both coarse and fine time granularities. We present hereafter some recent cross-layer adaptation approaches.

3.3.1 A Closed Loop Adaptation Approach

J. Philippe et al. have presented in (Diguet et al., 2011) an OS based adaptation approach for reconfigurable embedded systems. The main objective of this work is the implementation of reconfiguration managers with runtime decision and configuration control capabilities. As illustrated in Figure 1, a hierarchy of local and global configuration managers (LCM/GCM) is used to permit the separation between application-specific and application-independent configuration decisions. Authors have been interested in both architectural and algorithmic reconfigurations. The algorithmic reconfiguration is performed by the LCM which selects an application-specific algorithmic configuration independently of implementation details. However, the architectural reconfiguration is in charge of the GCM. It consists in tasks migration from software to hardware on a multiprocessor heterogeneous architecture based on a master GPP processor, a specific processor and reconfigurable hardware accelerators. Equally, a RTOS is used for transparent execution of tasks with their different HW and SW implementations, and the reconfiguration engine has been implemented as new OS services. A configuration table, which is built at an offline design space exploration step, contains a set of configurations among which one configuration is selected at run-time. A configuration is a combination of local and global parameters which correspond to both algorithmic and architectural (HW/SW tasks partitioning) configurations. Three constraints are supposed to be respected in this work: application QoS, execution time and power consumption. A simple timer, provided by the RTOS, is used for tasks execution time monitoring.

In addition to adaptive behavior integration, this work is one of the few works that were interested in studying system stability when self-adapting. Authors define a configuration period equals to Ne application iterations in order to minimize control and reconfiguration overhead. Every period, local and global managers collect metrics, and then the next configuration is selected so that the required QoS is respected and power consumption and execution time are optimized. This work has nearly covered the different steps required in adaptation process but has the limitation of being developed at a low level which makes it difficult to handle and requires a high level of expertise at hardware implementation. In addition, Target systems run a static set of tasks and reconfiguration decisions are limited to a pre-characterized set of configurations which does not fit to modern systems with dynamic tasks set. Furthermore, the predefined configurations have to be permanently loaded on the chip which increases static power consumption. Finally, Altera devices, without dynamic reconfiguration capability, were considered in this work.

3.3.2 GRACE Project

The GRACE project (Global Resource Adaptation through CoopEration) (Yuan et al., 2006) (Vardhan

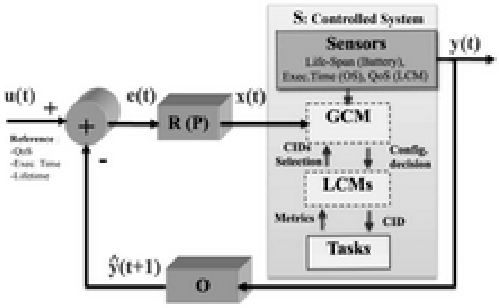

Figure 1: Closed loop adaptive system structure (Diguet et al., 2011).

et al., 2009) proposes a cross-layer adaptation framework that adapts multiple system layers at multiple time granularities. It addresses the conflict of adaptation scope and frequency through a hierarchical solution (c.f.Figure 2) which combines different modules at different adaptation levels:

- An infrequent expensive global adaptation that optimizes energy for all applications in the system. It only occurs occasionally at large system changes such as application entry or exit
- A frequent inexpensive limited-scope per-application adaptation that optimizes for a single application at a time. It is invoked every frame, adapting all system layers to the application's current demands. An internal adaptation adapts only a single system layer and may be invoked several times per application frame.

GRACE's first generation implementation, called GRACE-1 (Yuan et al., 2006) coordinates the adaptation of the CPU speed in the hardware layer, CPU scheduling in the OS layer, and multimedia quality in the application layer, in response to system changes at both fine and coarse time granularities. It focused also on the coordination between these cross-layer adaptations in order to reap their full benefit. GRACE-1's focus was on cross-layer global adaptation, for which it showed significant energy benefits. It reported a few experiments with hierarchical adaptation in the CPU and scheduler, but showed only modest benefits over global adaptation when running multiple applications.

Later, a second generation prototype, GRACE-2 (Vardhan et al., 2009) was developed to demonstrate the benefits of the hierarchical adaptation. This prototype implements a global adaptation in the CPU and application layers, and soft real-time scheduler, per-app adaptation in both CPU and application layers, and internal adaptation in the scheduler. It respects the constraints of CPU utilization and network bandwidth (assumed to be constant), while minimizing CPU and network transmission energy. GRACE-2 is network-aware. It adds a network bandwidth constraint in the global and per-application controller and considers global and per-application adaptations that are driven by the tradeoff in CPU time and network bandwidth usage. It has demonstrated that the lack of any network awareness results in very modest benefits from the hierarchical adaptation while running multiple applications. It has also shown that in a network bandwidth constrained environment, per-app application adaptation yields significant energy benefits over and above global adaptation. This project is very interesting since it has covered almost all system layers. However, it has been essentially applied to servers which are not embedded systems and daptation cost has not been taken into account.

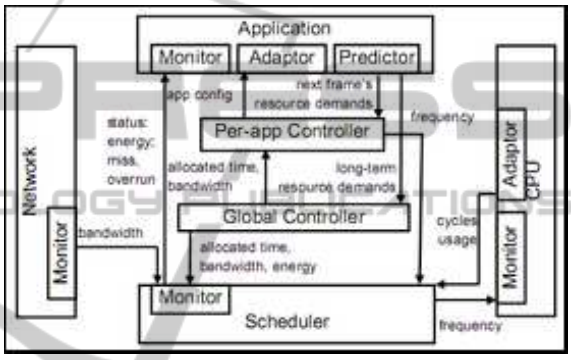

Figure 2: GRACE adaptation structure (Vardhan et al., 2009).

4 MDE BASED ADAPTATION APPROACHES

Researches on embedded systems modeling using MDE approach have begun several years ago and have shown the gain of using UML/MARTE standard (Schmidt, 2006) (Dekeyser et al., 2005). However, those particularly tackling adaptive embedded systems have recently begun. In the present study, we are interested on adaptation modeling for embedded systems using UML/MARTE profile. Currently, most of state-of-the-art works are focused on modeling hardware level adaptation through system reconfiguration, but few are interested in software level adaptation, or furthermore, in cross-layer adaptation modeling. In the sub-sections below, we present an overview of the MDE methodology and the MARTE profile, and then we recall works related to MDE based adaptation modeling that we classify in hardware and software levels.

4.1 Model Driven Engineering Overview

Several co-design methodologies shown in literature prove MDE to be well appropriate to embedded systems design. The model development process aims at decreasing the growing complexity of real time systems as well as the verification of their correctness. In fact, the abstraction view presented by the model avoids dealing with details and favorites reusability. An MDE based methodology starts with a high abstraction level and leads to a targeted specific model through a set of transformations and refinements. The produced models can be executed to ensure simulation, verification or execution. They can be used for another objective by being an input to produce other models.

One variant of MDE is the extension or the restriction of UML views for specific domains. It is presented through the notion of UML profile. From current times, the Object Management Group (OMG) has voted for a new standard for model driven development and analysis of real time systems. The adopted MARTE (Group, 2011) profile provides mechanisms to model appropriate specification in order to perform specific analysis. In fact, MARTE consists of three major packages. Foundation Package represents the foundational concepts for RTES design. It allows the specification of basic real time concepts such as non-functional properties (NFPs), time constraints and useful resources. The other two packages are refined from the first one. The second package named MARTE Design Model is dedicated for a detailed hardware and software description. As for the third package, MARTE Analysis Model package, it offers annotations for generic basis of quantitative performance and schedulability analysis. However, although it covers a large set of RTES features, MARTE still lacks concepts for an entire specification of adaptive RTES, specifically adaptive real-time task specification and tasks migration for multiprocessor architectures.

4.2 Hardware Level Adaptation Modeling

Reconfigurable systems' modeling is treated by several researchers and teams (Rafiq Quadri et al., 2009), (Rafiq Quadri et al., 2010), (Quadri et al., 2010), (Vidal et al., 2010), (Vidal et al., 2011). A recent project called FAMOUS (Famous Project,) is interested in this topic and involves many partners from research and industry. It aims at defining a complete design flow for embedded systems which integrates the dynamic reconfiguration of the hardware. We recall in the subsections below a number of approaches that have been developed under renowned projects and essentially dealt with dynamic and partial reconfiguration in embedded systems.

4.2.1 Reconfigurability under GASPARD Framework

Imran et al. proposed in (Rafiq Quadri et al., 2009), (Rafiq Quadri et al., 2010) a SoC co-design approach, developed in the GASPARD (Gamatié et al., 2008) framework, where they integrate control models based on mode automata to express DPR in modern FPGA. Xilinx FPGA is the targeted technology. They present some basic control semantics, such as Mode Switch Component and State Graphs that are used in a compositional manner to form mode automata. Then they propose A UML/MARTE design flow for automatic RTL code generation to implement dynamically reconfigurable FPGA. They have extended the MARTE profile by a set of stereotypes to model DPR features in Xilinx FPGA. To have a more complete set of concepts for high level FPGA modeling, they brought, for example, modification to the HwProcessor stereotype by adding the attribute Implementation_Type to specify wether the processor is implemented as a hardcore or a softcore IP. Then, to integrate DPR features, they made modifications to the HwComponent stereotype such as adding the attribute areatype to specify wether the hardware component is a static or dynamically reconfigurable region or other. They have also defined other extensions to support some Xilinx specific concepts such as bus macro, which permits routing between static and reconfigurable regions, the Internal Reconfiguration Access Port (ICAP), which assures the configuration memory read/write at run-time and the Reconfigurable Hardware Accelerator (RHA). Some control mechanisms have been added in (Cherif et al., 2011). A distributed modular reconfiguration controller has been presented. A reconfiguration controller can be defined in different forms such as a HDL hardware component or a soft-core processor. It can also be integrated in different system layers. Here, the controller has been integrated at the architecture layer. The complete modeling of a DPR-supported architecture has been elaborated using the extended UML/MARTE profile and targeting the Xilinx FPGA XC2VP30 Virtex-II Pro chip. Model transformations are applied to move to lower RTL detailed models arriving to automatic code generation of target application and the reconfiguration controller.

Although it provides rich extensions of the MARTE profile to get a more complete support for

dynamically reconfigurable systems, this methodology presents some limitations. First, the proposed system models only focused on the hardware properties; the MARTE package that has been most examined and modified is the Hardware Resource Model (HRM) package. Second, the approach is linked to the Gaspard framework, and thus limited to a specific application domain, that of complex intensive data-parallel applications. Finally, the work is platform-dependent. It is based on the Xilinx EAPR (Early Access Partial Reconfigurable) design methodology and proposes DPR models at a low level using Xilinx specific modules. This demands a high level of expertise and limits its large adoption in future reconfigurable architectures.

4.2.2 DPR for MPSoPC under MoPCOM Project

Another co-design methodology targeting the design and code generation of dynamically reconfigurable embedded systems has been developed under the MoPCOM project (Koudri et al., 2008). Authors tried to be less target-dependent by hiding technology specific details at high level models and allowing their adding at code generation level via a target-dependent code generation tool. In (Vidal et al., 2010) authors aim at modeling DPR in case of Multiprocessor System on Programmable Chip (MPSoPC). Their approach is based on run-time reconfiguration of co-processors connected to embedded processors. They define the three common Y-chart models, the application, architecture and allocation models using standard UML/MARTE stereotypes. The dynamic reconfiguration behavior is integrated in the allocation model. Specific stereotypes are used to give reconfiguration indications for the code generation tool. A «Reconfig» stereotype with an ID tag are added to application components to order the integration of a reconfiguration service in the allocated processor code.

In (Vidal et al., 2011), a component-based modeling methodology is presented. UML/MARTE elements and some predefined design patterns (Strategy and state patterns) have been used to reconfigurable systems. A network based reconfiguration service has been developed. The proposed reconfigurable MPSoPC is illustrated in Figure 3. One of the processors is identified as a «manager processor» which executes the reconfiguration API. When a processor wants to reconfigure its co-processor, it sends a reconfiguration request to the manager processor. This latter loads the requested bitstream either from a SDRAM cache or, if it doesn't exist locally, from an IP server connected to the chip through an Ethernet connection. Using the PlanAhead design and analysis tool, a bitstream is generated for the initial system configuration and a set of partial bitstreams are made for each possible configuration.

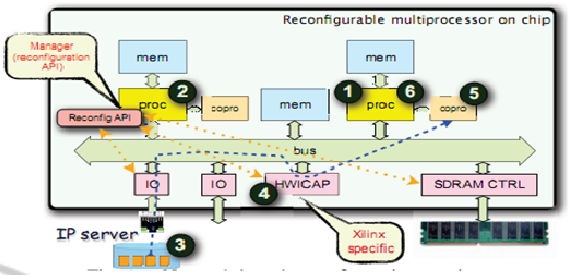

Figure 3: Reconfiguration service integration in MPSoPC (Vidal et al., 2011).

Furthermore, while authors claim to consider real-time components by using the «RTUnit» stereotype, they express no special care for the real-time constraint. Their allocated model is manually performed, so it gives no guarantee about the optimality of the selected solution. A design space exploration step and a schedulability analysis are thus necessary to be added in order to fit to modern complex systems which present dynamic tasks sets with timing constraints.

The presented approaches set important basis for dynamic reconfiguration support in embedded systems using high level design methodology. However, in order to efficiently uphold this capability, some control and decision making mechanisms need to be considered. These approaches are only interested in DPR modeling without tackling the adaptation engine which lies behind the reconfiguration decisions such as events, adaptation rules, system monitors, adaptation managers and assessor. Additionally, the reconfiguration capability is only developed for the hardware platform. The application behavior is unchanged and tasks scheduling is static. Furthermore, these works exhibit no support for real-time constraints and performance evaluation.

4.3 Software-level Adaptation Modeling

We present hereafter two recent works among the few ones tackling MDE based adaptation modeling at software level.

A model based approach is provided in (Krichen et al., 2010) to deal with software reconfiguration in distributed real-time embedded (DRE) systems with respect to non-functional properties. Authors started from the fact that MARTE and AADL standards present the limitation of permitting dynamic reconfiguration modeling for RTES by using only static sets of predefined modes (also called configurations)

Table 1: Comparison of adaptation approaches for RTES.

	UML/MARTE	Platform independent	Multiprocessor	Modularity	Hw/Sw Adaptation	Design-time verif.	Online time verif.	Online performance eval.
(Ye et al., 2010)		*	*					
(Diguet et al., 2011)		*	*		*		*	*
(Vardhan et al., 2009)		*		*	*		*	
(Rafiq Quadri et al., 2010)	*		*					
(Cherif et al., 2011)	*		*	*				
(Vidal et al., 2011)	*		*					
(Krichen et al., 2010)	*	*	*			*		
(Boukhanoufa et al., 2011)	*	*				*		

and transitions between them. Thus, they proposed a reconfiguration approach with non-predefined set of possible configurations by capturing them using mode structure concept. To describe this concept, they defined a simple MARTE and AADL inspired meta-model for reconfigurable software architectures in RTES which is a combination of model and component paradigms. The reconfigurable DRE system is modeled by a set of mode structures. A transition between mode structures specifies a reconfiguration activity which is activated by an event trigger and related to some constraints. The proposed meta-model is composed of three packages permitting to define three adaptation features: the configuration structures, the possible events and the reconfiguration activities. A UML profile for reconfigurable DRE was derived from this meta-model. This work was enhanced in (Krichen et al., 2011) by adding a fourth package to specify Structural and non-functional constraints and allocation constraints (policies of meta-modes allocation on execution supports). Another improvement was added in (Krichen et al., 2012) which consists in a verification approach for reconfigurable DRE systems permitting to verify, at design time, a set of non-functional properties such as CPU usage, memory and bandwidth usage,tasks deadline meeting, and deadlock and livelock freedom. They used the RMS scheduling algorithm, the Cheddar framework and defined algorithms for verification of the mentioned properties. This approach adds precious features to adaptation support using MARTE profile but is only limited to software modeling. The hardware architecture is considered unchanged. It also lacks support for online temporal verification and performance evaluation of adaptation.

(Boukhanoufa et al., 2010) (Boukhanoufa et al., 2011) were also interested in software adaptation modeling using UML/MARTE. They essentially focused on the validation of adaptation rules with respect to real-time constraints at design time. A state machine is used to model the application configurations and transitions between them. Time constraints verification is performed via a simple comparison between the worst case execution time of adaptation operations with the application allocated time slot. This work is among rare researches that have dealt with timing constraints at model analysis step. However, it is an off-line step where time constraints verification is performed on the basis of a set of unchanged timing properties, which does not fit dynamic systems with variable input data and resource constraints (variable execution times and WCET).

5 DISCUSSION

In this section, we present a comparative Table 1 which summarizes related works on adaptive embedded systems and compares them according to the criteria described in section 2. We derive from the previous study that most of MDE-based adaptation approaches tackle only hardware level adaptation modeling, essentially dynamic hardware reconfiguration (Reconfigurable architectures, modes and transitions modeling), whereas few interests exist in modeling adaptation at the software level. Cross layer adaptation is absent in MDE approaches. Along with this deficiency, MARTE does not support, among others, adaptive real-time tasks specification and interprocessor tasks migration. We therefore need to add extensions to MARTE to permit a generic and complete adaptivity features modeling in different system layers. We need to proceed towards efficient implementations of adaptive embedded systems through a well defined design flow which uses a common language to avoid misunderstandings between designers,

and permits the elevation of abstraction level, separation of functional and extra-functional concerns, and reusability.

Furthermore, most of existing approaches do not support the complete process of adaptation for real time embedded systems and lack modularity. They present a few interest in temporal properties control, which present a major constraint in RTES, as well as performance evaluation. More efforts need then to be done in the assessment step through temporal and performance analysis models in order to build more efficient and reliable systems.To the best of our knowledge, none of existing MDE approaches jointly dealt with the following features: adaptive RTES HW/SW modeling, MDE, UML/MARTE profile, modular adaptation process, online temporal verification and performance analysis. The gathering of these features in a complete design flow, from high abstraction level modeling to implementation, sets the originality of our future works.

6 CONCLUSIONS

According to our study around adaptive embedded systems, we notice a gap between adaptation techniques development and their utilization in embedded systems. This gap is due to compatibility and reusability issues and the complexity of integration of adaptive behavior in increasingly complex systems. Unfortunately, the limitation of computer aided design tools addressing adaptive embedded systems design further widens the gap. To deal with these issues, specialists in the field have resorted to MDE based design methodologies. The UML/MARTE profile is the most upcoming standard for model-driven development of embedded systems. It permits to model the properties of software and hardware parts of a RTES and the relations between them. It also offers some extensions such as performance and scheduling analysis. However, although it offers a large set of RTES features, MARTE still lacks a full specification of adaptive RTES.

Our work-in-progress does not consist in proposing a new adaptation approach, but lies in drawing up a complete design flow that raises the adaptation RTES development to a high abstraction level, based on MDE and UML/MARTE standards. Our aim is to set a generic modeling tool for adaptive RTES through MARTE extension which (a) permits models reusability and design complexity decrease and (b) covers the adaptation process modules, with special focus on online temporal verification and performance assessment. This approach can then be utilized by designers who contribute to the embedded systems adaptation field.

REFERENCES

Banachowski, S. A. and Brandt, S. A. (2002). The best scheduler for integrated processing of best-effort and soft real-time processes. In *In Proceedings of Multimedia Computing and Networking 2002 (MMCN Š02*, pages 46–60.

Boukhanoufa, M.-L., Radermacher, A., and Terrier, F. (2010). Towards a model-driven engineering approach for developing adaptive real-time embedded systems. In *NOTERE*, pages 261–266.

Boukhanoufa, M.-L., Radermacher, A., and Terrier, F. (2011). Offline validation of real-time application constraints considering adaptation rules. In *Proceedings of the 2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, TRUSTCOM 11, pages 974–980, Washington, DC, USA. IEEE Computer Society.

Cherif, S., Trabelsi, C., Meftali, S., and Dekeyser, J.-L. (2011). High level design of adaptive distributed controller for partial dynamic reconfiguration in fpga. In *DASIP*, pages 308–315.

Chu, H.-H. and Nahrstedt, K. (1999). Cpu service classes for multimedia applications. In *ICMCS, Vol. 1*, pages 296–301.

Dekeyser, J.-L., Boulet, P., Marquet, P., and Meftali, S. (2005). Model Driven Engineering for SoC Co-Design. In *NEWCAS'05*, Quebec, Canada. IEEE.

Diguet, J.-P., Eustache, Y., and Gogniat, G. (2011). Closed-loop–based self-adaptive hardware/software-embedded systems: Design methodology and smart cam case study. *ACM Trans. Embed. Comput. Syst.*, 10(3):38:1–38:28.

Famous Project. Anr famous overview.

Gamatié, A., Le Beux, S., Piel, É., Etien, A., Ben Atitallah, R., Marquet, P., and Dekeyser, J.-L. (2008). A Model Driven Design Framework for High Performance Embedded Systems. Research Report RR-6614, INRIA.

Gogniat, G., Vidal, J., Ye, L., Crenne, J., Guillet, S., de Lamotte, F., Diguet, J.-P., and Bomel, P. (2010). Self-reconfigurable embedded systems: From modeling to implementation. In *ERSA*, pages 84–96.

Goossens, J., Funk, S., and Baruah, S. (2003). Priority-driven scheduling of periodic task systems on multiprocessors. *Real-Time Syst.*, 25:187–205.

Group, O. O. M. (June 2011). A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems, ptc/2011-06-02. Object Management Group.

Hsia, S.-C. (2003). An adaptive video coding control scheme for real-time mpeg applications. *EURASIP J. Appl. Signal Process.*, 2003:244–251.

Koudri, A., Vojtsiek, D., Soulard, P., Moy, C., Champeau, J., Vidal, J., and Le Lann, J.-c. (2008). Using marte in the mopcom soc/sopc methodology. In *workshop MARTE*.

Krichen, F., Hamid, B., Zalila, B., and Coulette, B. (2010). Designing dynamic reconfiguration for distributed real time embedded systems. In *NOTERE*, pages 249–254.

Krichen, F., Hamid, B., Zalila, B., and Jmaiel, M. (2011). Towards a model-based approach for reconfigurable dre systems. In *ECSA*, pages 295–302.

Krichen, F., Hamid, B., Zalila, B., and Jmaiel, M. (2012). Design-time verification of reconfigurable real-time embedded systems. In *HPCC-ICESS*, pages 1487–1494.

Loukil, K., Amor, N. B., Said, M. B., and Abid, M. (2009). Os service update for an online adaptive embedded multimedia system. In *ISCC*, pages 721–725.

Mesarina, M. and Turner, Y. (2003). Reduced energy decoding of mpeg streams. *Multimedia Syst.*, 9(2):202–213.

Mohapatra, S. and Venkatasubramanian, N. (2003). Parm: Power aware reconfigurable middleware. In *Proceedings of the 23rd International Conference on Distributed Computing Systems*, ICDCS '03, pages 312–, Washington, DC, USA. IEEE Computer Society.

Ngoc, N. P., van Raemdonck, W., Lafruit, G., Deconinck, G., and Lauwereins, R. (2002). A qos framework for interactive 3d applications. In *WSCG*, pages 317–324.

Pillai, P., Huang, H., and Shin, K. G. (2003). Energy-aware quality of service adaptation. Technical report, UNIV. OF MICHIGAN.

Pillai, P. and Shin, K. G. (2001). Real-time dynamic voltage scaling for low-power embedded operating systems. *SIGOPS Oper. Syst. Rev.*, 35(5):89–102.

Quadri, I. R., Meftali, S., and Dekeyser, J.-L. (2010). Designing dynamically reconfigurable socs: From uml marte models to automatic code generation. In *DASIP*, pages 68–75.

Rafiq Quadri, I., Meftali, S., and Dekeyser, J.-L. (2009). A Model based design flow for Dynamic Reconfigurable FPGAs. *International Journal of Reconfigurable Computing*.

Rafiq Quadri, I., Yu, H., Gamatié, A., Rutten, E., Meftali, S., and Dekeyser, J.-L. (2010). Targeting Reconfigurable FPGA based SoCs using the MARTE UML profile: from high abstraction levels to code generation. *Special Issue on Reconfigurable and Multicore Embedded Systems, International Journal of Embedded Systems (IJES)*.

Said, M. B., Amor, N. B., Taher, F. B., Diguet, J. P., and Abid, M. (7-9 April 2011). A bi-constraints adaptation technique for embedded multimedia systems. In *International Conference on Multimedia Computing and Systems (ICMCS), 2011*, pages 1 – 6.

Satyanarayanan, M., Noble, B., Kumar, P., and Price, M. (1995). Application-aware adaptation for mobile computing. *SIGOPS Oper. Syst. Rev.*, 29(1):52–55.

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

Vahdat, A., Lebeck, A., and Ellis, C. S. (2000). Every joule is precious: the case for revisiting operating system design for energy efficiency. In *EW 9: Proceedings of the 9th workshop on ACM SIGOPS European workshop*, pages 31–36, New York, NY, USA. ACM Press.

Vardhan, V., Yuan, W., III, A. F. H., Adve, S. V., Kravets, R., Nahrstedt, K., Sachs, D. G., and Jones, D. L. (2009). Grace-2: integrating fine-grained application adaptation with global adaptation for saving energy. *IJES*, pages 152–169.

Vidal, J., de Lamotte, F., Gogniat, G., Diguet, J.-P., and Guillet, S. (2011). Dynamic applications on reconfigurable systems: From uml model design to fpgas implementation. In *DATE*, pages 1208–1211.

Vidal, J., de Lamotte, F., Gogniat, G., Diguet, J.-P., and Soulard, P. (2010). Uml design for dynamically reconfigurable multiprocessor embedded systems. In *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE 10, pages 1195–1200, 3001 Leuven, Belgium, Belgium. European Design and Automation Association.

Ye, L., Diguet, J.-P., and Gogniat, G. (2010). Rapid application development on multi-processor reconfigurable systems. In *FPL*, pages 285–290.

Yuan, W. and Nahrstedt, K. (2006). Energy-efficient cpu scheduling for multimedia applications. *ACM Trans. Comput. Syst.*, 24(3):292–331.

Yuan, W., Nahrstedt, K., Adve, S. V., Jones, D. L., and Kravets, R. H. (2006). Grace-1: Cross-layer adaptation for multimedia quality and battery energy. *IEEE Transactions on Mobile Computing*, 5(7):799–815.