Using MARTE in a Co-Design Methodology

Ali Koudri
Thales Aerospace Division
ali.koudri@fr.thalesgroup.com

Denis Aulagnier
Thales Aerospace Division
denis.aulagnier@fr.thalesgroup.com

Didier Vojtisek
IRISA
didier.vojtisek@irisa.fr

Philippe Soulard
Sodius
psoulard@sodius.com

Christophe Moy
Supelec
christophe.moy@supelec.fr

Joël Champeau
ENSIETA
joel.champeau@ensieta.fr

Jorgiano Vidal
Lab-STICC
jorgiano.vidal@univ-ubs.fr

Jean-Christophe Le Lann
Thomson
jean-christophe.le-lann@thomson.net

***Abstract*—The Model Driven Architecture is a promising approach aiming to fill the productivity gap due to the increasing technology and time to market pressure. In the field of real time embedded systems, this approach requires the use of well-adapted formalisms in a reliable process that guarantees the quality of the products. MARTE, the new standardized UML profile, provides those formalisms that are applied in the process defined in the MoPCoM SoC/SoPC research program, aiming to develop SoC and SoPC applications. In this paper, we discuss the main phases of this process and their use of MARTE.**

I. Introduction

Nowadays, embedded equipments rely more and more on components such as System on Chip(SoC) or System on Programmable Components (SoPC) based on Field Programmable Gate Array(FPGA) for low volume of production.

Embedded applications require more and more computing resources. These needs can be fulfilled by SoC / SoPC integrating on the same component analog circuits, hardwired ASIC Gates, programmable logical cells, memory blocks, high speed links, DSP and processor cores. Nevertheless, the SoC/SoPC development requires challenging productivity improvement to be able to address the component design complexity[ITR07].

Moreover, some embedded applications are constrained by certification norms like ARP-4754 (system), DO-178B (software) and DO-254 (hardware). In such case, SoC/SoPC developments have to be handled by rigorous methodologies based on well-adapted formalisms[KCA07].

In this paper, we discuss a new methodology to develop SoC/SoPC applications. This methodology is based on UML and MDA (Model Driven Architecture), and takes into account the achievements of codesign community. MDA[OMG03], promoted by the OMG (Object Management Group) replaces classical waterfall process (specification, design, coding), by a succession of models transformations, among these models:

- CIM (Computation Independent Model) models the requirements for the system and how the the system will be used,
- PIM (Platform Independent Model) is the view of the application independent from the target platform and can be considered as a functional description,
- PM (Platform Model) represents the target architecture used to implement the application,
- PSM (Platform Specific Model) results from the mapping of the PIM on the PM.

Actually, several PM of the SoC/SoPC Platform with different levels of abstraction are required to achieve the final implementation. For each PM, the process of allocation results in new PSM and analysis models. This approach enables to increase flexibility, reuse and separation of concerns.

The MDA approach was initially proposed in the context of software development and can be applied in the codesign domain. Indeed, "MDA notions" have been introduced long time ago in this field. For example, the Y-Chart methodology proposed by Gajski and Kuhn identifies behavioural, structural and geometric viewpoints[DR83].

Compared to conventional codesign methodologies, the use of UML/MDA for SoC/SoPC development enables to take advantage of the standards and the tooling developed by the UML/MDA community, such as the new UML profile for Modeling and Analysis of Real Time Embedded Systems (MARTE)[OMG07]. This profile enriches UML adding new notions for real time systems such as Non-Functional Properties or Time Access. Those features are seen in the section III.

In this paper, we present a new Co-Design methodology defined in the MOPCOM research project, aiming to provide a full MDA compliant tool to design SoC/SoPC applications. In this methodology, system analysis follows the Telelogic Harmony$^{\text{TM}}$ System Engineering process, based on the use of SysML[OMG06], improved with the use of MARTE. It is then refined to address platform and allocation modeling. Three levels of abstraction have been defined to describe the platform on which the application is mapped (see fig.1):

- APA (Abstract Platform Architecture) considering the platform as a set of basic services without notion of time,
- EPA (Execution Platform Architecture) refining the APA model taking into account software and hardware nature of APA components and timing constraints,
- DPA (Detailed Platform Architecture) detailing the EPA model, taking into account other constraints (power consumption, surface, etc).

In order to favor analysis, reuse or separation of concerns (computation / communication), the description of the com-

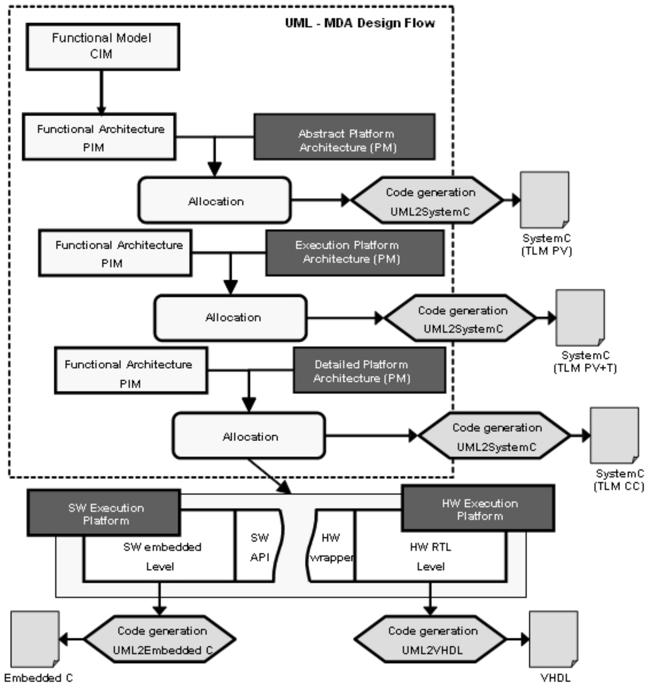

Fig. 1. MoPCoM overview

munications are described following the TML 2 standard as defined by the OSCI [Ini07]. This allows the same design space exploration allowed by different TLM levels (PV, PVT, CC) and the separation of concerns allowed by the MDA approach. The Programmer View (PV) level captures the SoC/SoPC architecture without clock or explicit timing. It focuses on functional correction based on the execution of the underlying Model of Computation. Timing concern is taken into account in the Programmer View + Time (PVT) level where data size and computation duration are described following the non functional requirements. Only the Cycle Callable (CC) level allows the fine verification of all those properties.

At each level, several analysis models (performance, power consumption, etc.) can be generated from each allocation in order to validate design choices. This approach allows early errors detection and design space exploration, fastering simulation speed. Moreover, in order to add meaningful properties to our methodology (predictability, maintainability, improvability, etc.) and automate it, we have used the SPEM profile[OMG05] to formalize it.

In this paper, we present our use of the UML MARTE profile through each phase of our methodology and we discuss the encountered problems from implementation or conceptual point of view.

II. Related works

The relevance of model based approaches in Co-Design have been widely discussed by the litterature. In [BDH06], authors enumerate the main advantages of such approaches: cost decrease, silicon complexity handling, productivity increase, etc. Athough several methodologies based on the use of UML/MDA have been applied, UML were not tailored to design SoC/SoPC applications. Then, it was extended and tooled by each of those methodologies, thanks to the profiling mechanism provided since UML 1.3. For example, in [BDDM04], authors propose extensions to describe Intensive Signal Processing Applications in a profile called ISP UML Profile. In [RSRB05], authors apply a SoC Design Methodology based on the use of a UML profile called *UML for SystemC Profile*. This profile allows direct mapping between the models and the concepts of the SystemC language. The same kind of approach is presented in [WZZ$^+$06]. A SystemC UML profile is used to generate a TLM/PV SystemC code skeleton. In [CSL$^+$], a design environment called *Metropolis* and aiming to design embedded systems is proposed. It tailors UML in order to describe execution platforms and their refinments. They provide set of stereotypes representing structural parts of the platform and corresponding model of computation.

In [RSRB07] a development process for embedded systems called UPES[1], based on the use of SystemC profile, is presented.

It is interesting to notice how, through all those approaches, common needs like platform and allocation description at several levels of abstraction are filfulled by different approaches. Some of them propose a direct mapping of the platform concepts to the target languages while others use target languages as concrete syntax in purpose of generation or execution. The problem with those approaches are the dependency to the tools implementing very specific profiles or DSL (Domain Specific Languages). With the standardization of the MARTE profile, we think this drawback will be handled by the use of common formalisms to describe real time and embedded systems.

III. Use of MARTE in the MoPCoM SoC/SoPC research project

The MoPCoM process is a Co-Design Process based on the use of models and transformations. Since one of the main features of the MDA is the separation of concerns, there is a need to dispose of well-adapted formalisms to describe each part of the MDA model. For several reasons, such as communication improvements or tooling, we have made the choice to use standardized metamodels and profiles. The new standardized UML MARTE Profile add semantics to UML for Real time Embedded System Modeling. This profile is organized around several packages capturing different concerns :

- The Non Functional Properties (NFP) package enables to describe features that are not directly related to the business model (performance, memory usage, power consumption, etc.) and mechanisms to attach them to model elements,
- The Time package enables modeling of time structure and access (continuous time, clock, etc.),

[1]Unified Process for Embedded Systems

- The Generic Resource Modeling (GRM) package enables modeling platform resources from high level perspective and mechanisms to manage access to those resources,
- The Allocation package enables modeling of spatial and temporal allocations,
- The Detailed Resource Modeling (DRM) package allows modeling of hardware and software resources at several levels of granularity; it contains two sub-packages enabling the modeling of Software Resource (SRM) and Hardware Resource (HRM),
- The Generic Quantitative Analysis Modeling (GQAM) package allows several types; It contains two subpackages for modeling Performance Analysis (PAM) and Schedulability Analysis (SAM).

The array in Table I sum up the requirements of the process and formalisms that satisfy those requirements. The system properties, mesurable or not, like power consumption or latency, can be expressed using the Value Specification Language (VSL) provided by MARTE at each stage of the process.

MoPCoM Process Requirements	**Solutions**
Model Computational Independent Model	SysML + MARTE NFP and RTE-MoCC
Model Platform Independent Model	SysML + MARTE NFP and RTE-MoCC
Model Abstract Platform Architecture	MARTE GRM and NFP
Model Execution Platform Architecture	MARTE DRM + NFP + Time
Model Detailed Platform Architecture	MARTE DRM + NFP + Time
Model Allocation (PIM to APA/EPA/DPA)	MARTE Allocation + NFP
Model Analysis Models	MARTE PAM and SAM

TABLE I
MoPCoM PROCESS REQUIREMENTS

A. *System Analysis*

The system analysis goes from the requirements analysis to the system design and is covered by the SysML profile [OMG06]. Since this part is not really in the scope of this paper, we will just give a quick overview in the following points:

- requirements elicitation,
- static validation of the requirements,
- manual transformation of the requirements into use cases and interactions,
- use cases validation through simulation,
- identification of the subsystems,
- operational contracts allocation,
- interaction refinements,
- system design validation.

During this stage, we use some concepts of the NFP packages in order to describe the required quality of services in term of real-time features of behaviors and interactions. At the end of this stage, an annotated functional architecture is provided for following steps, including allocations and refinements.

B. *Abstract Platform Architecture*

The platform is defined as a set of resources like processing resources, communication media or storage resources. The purpose of this level is to describe applied mechanisms in order to realize high level functions. Communications between APA blocks are point-to-point and realized through abstract channels providing set of basic services allowing data transport. Actually, what we aim to describe at this level is the underlying model of computation supporting the execution of the application and since we just want to demonstrate the functional correction of this model, it is thus untimed. Synchronization mechanisms ensure the characterization of the causal relations between processing elements and then ensure the predictability of the system behavior. The fact that the model is untimed doesn't mean that timing constraints inherited from the application model are not taken into account. Functional delays can be then inserted into the APA model, regarding the functional constraints.

Once the platform has been defined by the software or hardware engineering teams, allocation can be modelled using well-adapted formalisms provided by the allocation package of MARTE. The allocation can be of spatial, temporal or hybrid nature. Since many allocations with different characteristics (performance or schedulability) are possible, models of allocation are concerned with the functional correctness issue. The figure 2 summarizes issues related to the allocation at the APA level. The application is modelled through business components communicating through (a)synchronous operation calls or events send/reception while the platform is modelled through APA blocks communicating through channels implementing data transport services. The execution semantic described in the business model must be verified through the allocation. Since the APA platform is untimed, there is no notion of protocol at this level.

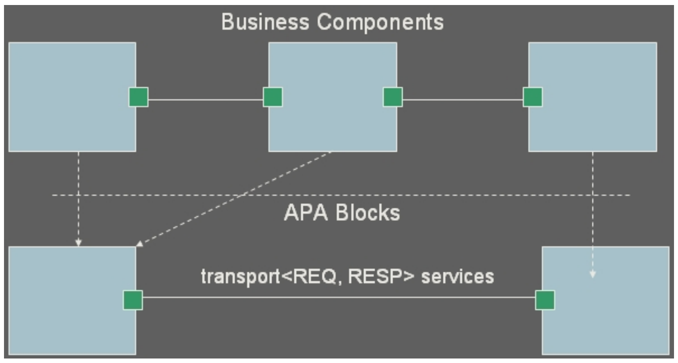

Fig. 2. APA allocation

In order to describe the APA platform, one need to describe resources from a high level perspective, without prejudging of their hardware or software nature, and take into account their quality of service. The GRM package of MARTE provides all needed stereotypes and tagged values to represent resources (computing resources, storage resource, communication media, etc.) and their use. This package is used in conjunction with the NFP package for the quality of services and Time Modeling package for timing constraints. The GRM package provides also mechanisms of synchronization and resource management that fulfills the needs for APA modeling. The package *Allocation Modeling* provides all needed stereotypes to allocate business model to APA plaform. Since an allocation comprises at least one application end and one execution platform end, we need to constraint the set of possible mappings for this level refining the allocation definition.

C. Execution Platform Architecture

The MOPCOM EPA platform level aims at providing a more precise platform than the APA level : while APA focused on point-to-point communication, EPA provides more details with respect to the expected final *topology* on the system. The right methodological tools at this level are clearly MARTE sub-profile HRM – *Hardware Resource Modeling* on which MOPCOM relies. Concerning the usage of MARTE, the stereotypes used here are hwComputingResource (processor,...), hwCommunication(bus,...).

EPA is geared towards architects willing to depict global scenario of data movement over the chip and computing resources, without comitting to final IP choice. EPA is thus fundamental to preserve the representativity of application with respect to the platform, while enabling fast design space exploration and high-speed simulation, based on generic high-level components.

In this sense, many hardware resources must be adequately identified at this level, without over-specifying them, which is often the case when architects are not provided with apropriate tools to help their specification. A natural companion modeling concept of EPA is transaction-level modeling, as promoted by Gajski and SystemC community in general.

Among those resources, communication media like busses, NoCs and buffering medium need to appear in EPA (the right stereotypes here are HWI/O and HWStorage). Beyond the communication aspects, computing resources are also included in this model, possibly still in an abstract (behavioral) version. For instance, instead of describing a real RISC processor or accelerator, EPA proposes to aleviate the need for deep hardware description and invites to keep the functions executed by these processing elements as the representative of the blocks themselves.

Thus a wrapper needs to ensure that the function can indeed have access to the communication media : here we prone a simplified version of generic DMA at the I/O of the functionnal data-intensive blocks, possibly assisted with control/status registers that ensure the high-level control of the application in a processor-like manner. Using this approach, it is possible to set up dummy experiments to test these notions and their adequacy to a broader set of other experiments. In the following drawing, we depict typical experiment within reach of an EPA model : the goal is to move data from DDR zones to other DDR zones, after some modifications executed on the set of data. A global controler, mimicking a micro-controler/processor, observes the completion of the various data transfers in order to enable the next transfers. Entire data structures can be moved from one location to another without any serialization mecanism to low level data representation. As a consequence of this separation of concerns, early application porting debug activity can start, while studying the underlying lower level mecanisms constraints that need to be fullfilled to respect the expected final behavior.

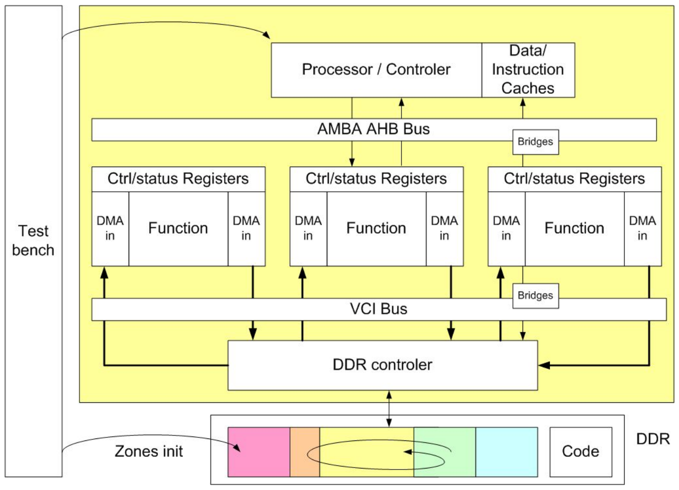

Fig. 3. EPA typical experiment : data are moved from one location to another, through well identified communication links, possibly simulated at a behavioral level, abstracting away details about finer details.

Clearly here, the EPA level envisions the mapping process as a simple binding of application functions to processing elements and point-to-point communication links to concrete communication elements, with no further notion of compilation to dedicated binary format, nor the resort to ISS/interpreters : functions can still be seen as abstract tasks, just as in the APA level. Note that some such blocks or units may be in charge of executing several functions : in this case, this coarse grain resource sharing then assumes the presence of a basic abstract arbitration mecanism.

The performances of the final hardware cannot be measured at the EPA level. However, the unambiguous identification of computing resources *types* allows for analysis and parameterized estimations : we keep as much room as possible for architectural exploration, by tuning these parameters adequately. The metrics on which it is possible to focus on are the channels required bandwidth and relative computation power of the processing units. *MARTE SAM package* (schedulability analysis model) are clearly in line with the requirements at the EPA level. For instance we reinforce the EPA structural model with *behavioral scenarios* with the help of Message Sequence Charts : synchronization aspects of locking/unlocking busses can be handled through (Metaclass) BasicInteractions::Message, even if it has been promoted with OS in mind. In the same manner, many elements of PAM package seem adequate to analyze non-functional properties especially concerning communication side. Their exact concrete usage is still under study.

As emphasized here, EPA prones to resort to behavioral descriptions instead of deeply detailed descriptions. For instance, concerning the processors, the further refinement to real binary code (on the embedded SW side) and cycle & pin accurate hardware will be treated in the 'next' level, i.e DPA level, where *measures* becomes meaningfull. The same applied for communications, where the complete protocols will need to be fully modeled.

One interesting question is about the minimum set of hardware features to embed in this EPA level so that the model of computation can be implemented either manually or handled by the MDA toolflow : register control, DMA control etc.

D. Detailed Platform Architecture

The DPA is the last level before generate the RTL code. At this level, all information about the RTL system is specified, so it can be generated. The platform is defined as a set of IPs that communicate among them through busses, as in EPA level. The purpose is to have a cycle accurate platform model, where all transaction is done by a refined and precise protocol, defined here at a cycle level. The internal state of the IP blocks can be observed at each clock cycle. There are three main improvements from the EPA level:

- Existence of hardware clock – All synchronous elements must be connected to a clock.
- Definitinion of a RTL communication protocol – All communications must be well defined in terms of signals and clock cycle.
- Physical hardware details – card layout and power consumption.

To fulfill these improvements, MARTE elements are used. The hwClock MARTE stereotype is used to model clocks. The clock values are then defined with NFP package (frequency, counter, ...). MARTE does not provide an explicity way to precise a protocol, it just have the elements to define a protocol, that means: bus (with their signals) and the behavior is modeled with a UML state machine. The protocol constrains is done by NFP and RTMoCC MARTE elements (RtAction,...). One possibility is to use a protocol library/stereotype to speed up the design process, such solution is under study.

The figure 4 shows the main issues related to the allocation at the DPA level. The application is refined to allow cycle precising and the platform contains all hardware details to support these definitions. The stereotypes used at EPA level are now refined to be detailed at implementation level, i.e. a processor type, ISS and registers will precise elements to have a cycle accuracy. This is achieved with the tags in HRM MARTE sub-profile.

The platform is modelled through DPA blocks communicating through buses with a detailed cycle-accurate protocol. The protocol must be able to implement at a RTL level the Model of Computation defined at higher levels. Here, the exploration is done at protocol level, with a clock, bus/protocol to achieve the desired performance. Now, the meaning of a communication implementation is important, where complex data structure can not be moved from blocks, it must respect a communication protocol and bus specification (address and data size).

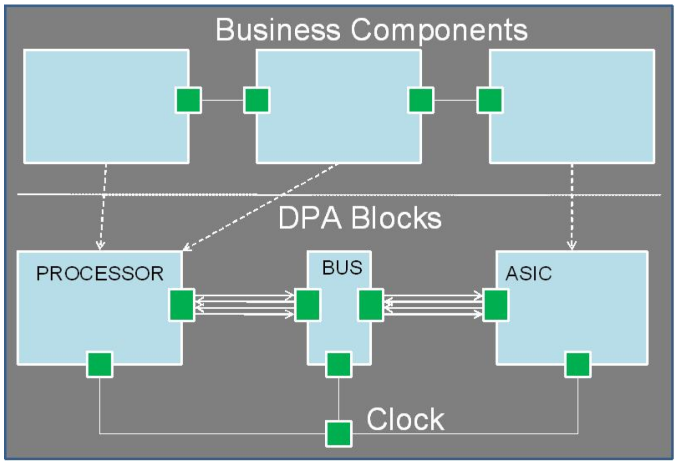

Fig. 4. DPA allocation

The real performance of the system can be measured at DPA level, possibiliting the last refinements to achieve an optimal configuration of the PEs (physical characteristics and logical refinements) and how they communicate. All protocol details allows the observation of the system at the precision of the physical clock cycle tick. All the real-time constrains specified at application level must be verified here.

The MARTE sub-profile HRM – *Hardware Resource Modeling* – is used to specify the DPA elements, but, refined from the EPA model, it must be detailed. Thus this profile we can specify all the hardware elements and, for that, the final physical hardware characteristics is done by the HWPhysical (HRM sub-profile) and is composed by:

- HwLayout: Component (card, channel, chip, port)
- HwPower: power supply, cooling supply

An element, already stereotyped with its logical characteristic (processor, bus, IO, ...) and detailed in DPA level is now stereotyped also with the physical characteristics. Such definition allows the final allocation to be made. For instance, with the HwPort stereotype it is possible to define its pin number, area, weight, etc (from HwComponent), which allows the application port to be allocated into hardware ports. Such allocation allows a really refined unbiguous hardware platform definition, where each application element is allocated into a final hardware element, RTL-detailed.

We precise the time with the Timing sub profile. The stereotypes are "tagged" with NFP elements to define the details of the platform. The allocation is done by the Alloc sub-profile. Also, the use MARTE PAM – *Performance Analysis Modeling* – and SAM – *Schedulability Analysis Modeling* – sub profiles allows the specification of the system performance.

Modeling constraints must be defined to allow an automatic code generation, for now, the constraints are:

- Must use of HRM stereotypes to all hardware elements.

Need to define which tags are or not optional for each stereotype.

- Protocol specification. For now, the only way is to use a UML state machine to handle communication. The use of a library and/or new stereotypes is in progress.
- Every synchronous element MUST be connected to a clock.

With a refine cycle accurate specification, it is possible to measure the final system performances. As the final hardware is detailed at DPA level all the communication details can be verified. The RTL model of the system can be automaticaly generated. This model can be used to measure and generate the final system in the target desired languages (for Hardware and Software).

IV. Discussions

The development methodology we have presented above is being experimented on a video application for SoC and a RF digital receiver application for SoPC. Despite we think that MARTE can fulfil most of the requirements of a modelling languague for SoC/SoPC design and analysis, we faced some limitations applying MARTE. Those limitations are either related to the implementation of the profile or missing concepts in the profile.

Allocation: MARTE proposes to specify the "allocate" relationship between business components and resource components with notation based on a dashed line with an open arrow head. This graphical approach can be very cumbersome and messy for large models. Some tooling, using widgets for example, could be developed to facilitate the allocation activity.

Model of Computation and Communication: MARTE foundation enables to define alternative MoCC. Modelling or implementation of embedded applications generaly rely on heterogenous MoCCs such as Synchronous Data Flow (SDF), Communication sequential process (CSP), Petri Net (PN), Kahn Process Network (KPN). These specific MoCCs should be introduced in MARTE.

Action Semantics: In the purpose of executing and testing models, UML has introduced the Action Semantics to provide software independent language specification for actions in their models. Nevertheless, those action semantics have been defined in the context of software development, and then do not integrate concerns from other domains like system or real time domain. Typically, actions defined by the UML specification are related to other UML concepts and since new concepts have been introduced in MARTE, we think it would have been interesting to add meaningful actions related to them and also provide associated concrete textual syntax.

V. Conclusion and future works

In this paper, we have discussed a co-design process called MoPCoM based on the use of the MARTE profile. For each phase of this process, we have presented the selected stereotypes of the profile and constraints related to their use. We have outlined the main encountered problems from conceptual and implementation perspective. Future works are related to VHDL et SystemC code generation after the allocation of the application on the Detailed Platform Architecture. This development is done using MDWorkbench transformation tooling from Sodius[2]. Further works will be carried out on modelling and implementing dynamically reconfigurable applications.

Acknowledgments

The research program MOPCOM[3] SoC/SoPC is supported by the French National Research Agency (ANR), the "cluster of clusters" and "Media and Networks" from Brittany and Pays de la Loire regions. The program partners are the end-user companies Thales, Thomson, Sodius as an UML tool provider and the academic research institutions ENSIETA, LESTER, Supelec and INRIA.

References

[BDDM04] Pierre Boulet, Cédric Dumoulin, J-L Dekeyser, and Philippe Marquet. Uml 2.0 structure diagram for intensive signal processing application specification, 2004.

[BDH06] Pierre Boulet, Cédrid Dumoulin, and Antoine Honoré. Model driven engineering for system-on-chip design. In *From MDD Concepts to Experiments and Illustrations*, pages 91–109. ISTE ltd, 2006.

[CSL+] Rong Chen, Marco Sgroi, Luciano Lavagno, Grant Martin, Alberto Sangiovanni-Vincentelli, and Jan Rabaey. Uml and platform-based design.

[DR83] D.D.Gajski and R.Khun. Guest editor introduction : New vlsi tools. In IEEE, editor, *IEEE Computer*, volume 16, pages 11–14. IEEE Computer Science, 1983.

[Ini07] Open SystemC Initiative. Osci tlm2 user manual, 12 2007.

[ITR07] ITRS. The international technology roadmap for semiconductors : Design, 2007.

[KCA07] Ali Koudri, Joel Champeau, and Denis Aulagnier. Fpga rf transceiver development based on uml mda approach. DAC, 2007.

[OMG03] OMG. Mda guide version 1.0.1. Technical report, Object Management Group, 2003.

[OMG05] OMG. SPEM 1.1. Technical Report ptc/05-01-06, Object Management Group, 2005.

[OMG06] OMG. Sysml profile. Technical Report ptc/06-04-03, Object Management Group, 2006.

[OMG07] OMG. Uml profile for marte, beta 1. Technical Report ptc/07-08-04, Object Management Group, 2007.

[RSRB05] E. Riccobene, P. Scandurra, A. Rosti, and S. Bocchio. A soc design methodology involving a uml 2.0 profile for systemc. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 704–709, Washington, DC, USA, 2005. IEEE Computer Society.

[RSRB07] Elvinia Riccobene, Patrizia Scandurra, Alberto Rosti, and Sara Bocchio. Designing a Unified Process for Embedded Systems. In *Fourth International Workshop on Model-Based Methodologies for Pervasive and Embedded Software*, pages 77–90. IEEE Computer Science, mars 2007.

[WZZ+06] Ting Wang, Xue-Gong Zhou, Bo Zhou, Liang Liang, and Cheng-Lian Peng. A MDA based SoC Modeling Apporach using UML and SystemC. In *Proceedings of the sixth IEEE International Conference on Computer and Information Technology (CIT'06)*, pages 245–245. IEEE Computer Society, september 2006.

[2]http://www.sodius.com

[3]http://www.mopcom.fr