MULTI-LEVEL MODELING AND SIMULATION OF COGNITIVE RADIO EQUIPMENTS

Stéphane LECOMTE[1,2]
[1](Technicolor, Cesson-Sévigné, France, stephane.lecomte@technicolor.com);
Christophe MOY[2] and Pierre LERAY[2]
[2](Supélec/IETR, Cesson-Sévigné, France, {firstname.name}@supelec.fr);

ABSTRACT

In this paper we propose a co-design methodology, based on a UML and Model Driven Architecture approach, in order to design embedded reconfigurable systems. We particularly focus on the design of software radio and cognitive radio equipments. A potential hardware target is System on Programmable Chip (SoPC), like FPGA components, taking into account the specific capability of partial reconfiguration. A reconfiguration and cognitive management architecture (HDCRAM for Hierarchical and Distributed Cognitive Radio Architecture Management) is used in addition to the device's radio application. The system (functional architecture and hardware architecture) is modeled in UML using MARTE. Going through Model-to-Model transformation the cognitive radio equipment is first modeled at a high abstraction level first, based on HDCRAM metamodel, and then modeled at a lower level of abstraction in SystemC, which enables to simulate and validate the system at transfer level (TLM). We illustrate the proposed concepts with a cognitive radio case study involving reconfigurable radio signal processing executed on a dynamic reconfigurable target.

1. INTRODUCTION

Real Time Embedded (RTE) systems, such as radio communications systems, integrate more and more functionalities. Reconfigurability or versatility is of particular interest since it may contribute to many emerging and desirable features of future systems, such as power consumption mitigation, quality of service improvement, processing resource use optimization, etc, thanks to an optimal adaptation of the hardware resources at each instant. These are features that can be met in the radio domain for instance (Software Defined Radio (SDR) [1] and Cognitive Radio (CR) [2]), as well as the video processing domain (Reconfigurable Video Coding [3] standardization initiative). The suggested solution to answer the reconfigurability challenges is the sharing and adaption of hardware resources since these domains are very demanding in terms of processing power (up to hundreds of GOPS for SDR for instance). Thanks to the ever-increasing performance of digital electronics, an embedded system can be integrated on a single chip: a System on Chip (SoC) or a System on Programmable Component (SoPC) inside FPGA reconfigurable components. Thus the design of such systems must take into account new challenges.

To understand and define the behavior of such equipments, it appears necessary to model cognitive radio features at a high level of abstraction. Moreover an enhancement of software/hardware design facilities with a reliable process is required to fill the productivity gap. Several approaches have been promoted by Electronic System Level Design (ESL) such as IP reuse [4][5], High-Level Synthesis (HLS) [6], platform based-design [7] and Model Driven Architecture (MDA) [8][9] (which offers a common design framework whatever the design phase and actors).

But there still remains a lot of improvement to be added to design tools in order to answer engineering and design challenges and especially for adaptive system in the SDR and CR context. This is the scope of this paper to propose solutions to the issue.

2. RELATED WORKS

In [10], the Oldenburg System Synthesis Subset + Reconfigurable (OSSS+R) extensions, based on OSSS modeling approach, which used an extension of SystemC, is one of the main approaches to model an adaptive system. This approach is based on polymorphism concept of oriented object features.

In [11], the authors use UML sequence diagrams to model the dynamic reconfiguration. Each call in the sequence diagram is stereotyped with a configuration, which starts a reconfiguration procedure. But the hardware platform is not modeled in this approach. Moreover, the targeted platform, composed of a processor and a hardware accelerator (FPGA), is only considering a static processing inside the FPGA (i.e. not implying partial reconfiguration, so fixed at run-time).

In [12], the authors model the dynamic Partial Reconfiguration (PR) with a model of the hardware platform using an extension based on MARTE profile for the necessary stereotypes, specific to the design problem. This approach is implemented in GASPARD 2 environment tool.

In [13], the authors extended the SystemC kernel in order to simulate the dynamic reconfiguration of a system. They use a dedicated channel to control the different configurations of a function.

Compared to previous works, our approach proposed in this paper covers all the cases of reconfiguration at run-time: software functions (typically inside a processor) and hardware reconfiguration (typically inside a FPGA). The goal is also to use a standard language at high level (UML and SystemC) whatever the target nature ("software" – for processors – or "hardware" – for FPGAs). This is the reason why we call it a high level language as it is not modified depending on the target nature. Based on the MOPCOM methodology [14], an extension can be proposed to model reconfigurable systems. This point is presented in the next section. In section 4 is detailed the simulation of reconfigurable system in SystemC. A simple cognitive radio use case shows this extension in section 5 as a proof of concept. Finally, we conclude.

3. MODELING RECONFIGURABILITY

In a reconfigurability context, which means in SDR domain for instance that the same hardware platform can support several radio applications, systems can be classified in three categories:

- Non-reconfigurable systems: execution of an application without change of mode, and the hardware target is static during the life of the system;
- Functional reconfiguration: the functionality of the system can change, evolve during the life of the system but the hardware target is static;
- Hardware reconfiguration: an area of the hardware target is dynamically reconfigurable.

In order to model reconfigurable systems the necessary concepts have been added to the proposed design process.

3.1. Functional reconfiguration

Like used in the OSSS+R approach [10], in the Platform Independent Model (PIM) of the higher abstraction modeling level of MOPCOM (described in [14]) the polymorphism semantic of object oriented programming is used to model a reconfigurable function (equivalent to operator). For example, in a radio communication context, when a system switches from a standard to another one, the characteristics of modulation of the signal can change. So the different implementations of modulation have the same interfaces, same basic methods and they can be modeled by objects sharing a general base object type (as shown in Figure 1).

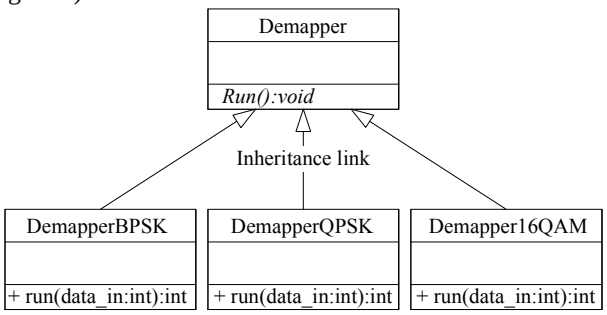

Figure 1. Example to model reconfigurable function with polymorphism without management of reconfiguration.

The class *Demapper* is abstract and its method *run()* is virtual (without implementation). The three classes, which inherit of this one, implement this method *run()* (either the algorithm, or by a different set of parameters, etc).

This static representation, of reconfigurable function, must be completed by adding elements to manage the dynamic of the system. The system must manage the reconfiguration according to the context. For that the design pattern Strategy [15] is well adapted. However we opted for a dedicated architecture in order to cope with real-time constraints of reconfiguration. So [16] and [17] proposed to use a reconfiguration management. As shown in Figure 2, a functional architecture of reconfiguration management called Hierarchical and Distributed Reconfiguration Management (HDReM) adapted to a SDR context is proposed, as SDR is intrinsically including versatility. It proposes a distribution of the tasks of management in three levels of hierarchy *L1_ReM*, *L2_ReMU* and *L3_ReMU*.

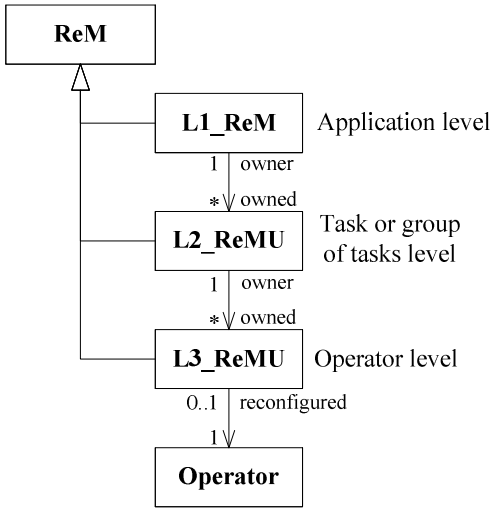

Figure 2. An overview of the reconfiguration management architecture (HDReM).

This type of architecture, deployed in a heterogeneous processing context (i.e. DSP, FPGA, etc.), makes it possible to satisfy the needs for management of reconfiguration to multiple granularities within the system. Indeed, by means of this architecture, we can handle real-time reconfigurations on the scale of a complete change of application (which will be taken care by L1_ReM level), down to reconfigurations of less granularity as the change (or update) of a function. In the latter case, this corresponds to make a local reconfiguration of an implanted operator on any technological target deployed on the platform. Another important point of this architecture of reconfiguration management is that it is particularly adapted to FPGA components reconfiguration management, in the sense of dynamic and partial reconfiguration at run-time. Moreover, this architecture is formalized in a metamodel [16] that can use UML/MARTE models.

This reconfiguration management architecture is integrated in the MOPCOM methodology using the HDReM metamodel. So, when the designers use the MOPCOM methodology to model their system, and when the specifications of this one requires reconfiguration, they must use the polymorphism representation for the reconfigurable function, and associate a dedicated implementation of the HDReM architecture.

With these two elements (polymorphism and reconfiguration manager architecture) a reconfigurable system can be modeled, at a high-level of abstraction, independently of the target platform. However it is imperative to validate this level of modeling in order to verify the functionalities of the system and to observe the impacts of the reconfiguration. As UML models cannot be simulated, we opted for a SystemC simulation (this language allows to make software/hardware co-modeling at different levels of abstraction). SystemC modeling is explained in the section 4.

In this sub-section, functional architecture of the system is modeled at high-level independently of target platform. Of course, the type of target platform also has an impact on the behavior of the system. Thus it is needed to take into account these elements during the modeling phase of the hardware platform.

3.2. Hardware modeling

We particularly focus on the challenging issue of partial dynamic reconfiguration of FPGA components management. This technological breakthrough is available with Xilinx FPGAs Virtex Families [18]. FPGA reconfiguration capabilities have been used for years at design time, but this new feature brings it at run-time. This introduces a new paradigm: hardware processing power efficiency combined with the same flexibility as software [19].

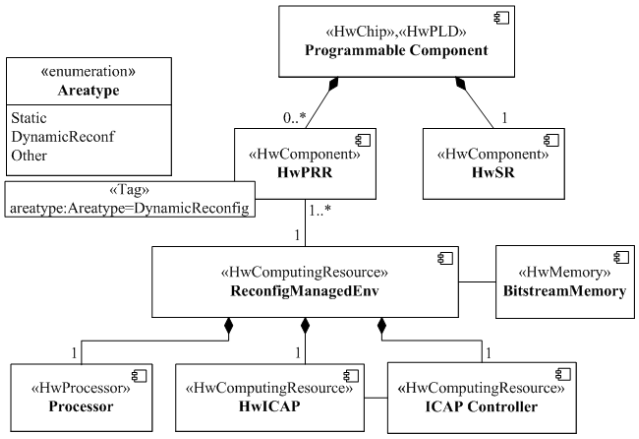

Figure 3. Partial dynamic reconfiguration hardware architecture description for FPGA.

To use this technology, a dedicated primitive, named ICAP, is implemented into the FPGA, as well as a processor to manage the reconfiguration. Moreover a memory is necessary to store the bitstream (binary code that programs the FPGA). So the concepts added to UML and MARTE profile in this paper are specific for this case. But we can also propose a generic modeling or any kind of hardware reconfiguration.

Figure 3 shows different components used for modeling the internal architecture of a FPGA component with partial dynamic reconfiguration technology.

A programmable component can have only one static hardware resource (*HwSR*) and one or several partial dynamic reconfigurable hardware resource (*HwPRR*). If the architecture of the FPGA has at least one *HwPRR*, the model contains a reconfiguration management environment (*ReconfigManagedEnv*). If the programmable component is a FPGA from Xilinx, the *ReconfigManagedEnv* component is composed of a processor, a computing resource which manages the reconfiguration (*HwICAP*) associated to a controller and a memory (to store the different configurations of an operator, i.e. their bitstreams). To differentiate *HwSR* and *HwPRR* the resource is tagged with the dynamic type of area in *DynamicReconfig*.

Moreover, specific stereotypes from MARTE Profile are used to describe hardware platform (Figure 3).

4. MODELING IN SYSTEMC

As explained in sub-section 3.1, the SystemC language is a good candidate to simulate and to validate the model of the system. A manual coding of SystemC model, equivalent of UML model in the higher-level of abstraction of MOPCOM methodology (named Abstract Modeling Level), was developed. Due to SystemC constraints, the model cannot be exactly equivalent to the UML model.

In fact, in the SystemC language a module[1] cannot be destroyed during execution as it is the case for oriented-objects languages such as C++ or Java (the creation and the destruction of modules must be done during the elaboration of deployed architecture step and during the end-of-simulation step).

To bypass this constraint, a generic module for reconfigurable operator named *OperatorReconfig* (as shown in Figure 4), which inherits of the abstract class *Operator* of the metamodel of HDReM, was created.

In this dedicated module, all the possible configurations of an operator (function) are created (during the elaboration of architecture). But only one is active at a given moment following the context in which is the system. When the L3_ReMU, associated to this reconfigurable operator, sends an order to reconfigure it with the right implementation, the *OperatorReconfig* deactivates the current configuration and activates the new configuration which corresponds to the new context. The reconfiguration order and the new configuration (represented by an ID) are sent through a channel between the L3_ReMU level and its associated reconfigurable operator. This last one has a method to activate the good implementation of the function. During the reconfiguration step, the reconfigurable operator cannot receive a new order. In this high level of modeling, no time constraint is specified. It is just to observe and validate the functionalities of the system. But this could be added at lower level of modeling as specified in MOPCOM [14][1]. But this is out of the scope of this paper.

For example, Figure 4 shows a *Demapper* operator which has three operation modes (configurations): Config_1 (BPSK), Config_2 (QPSK) and Config_3 (16QAM).

5. USE CASE

In order to illustrate the modeling concepts of reconfigurable systems presented previously, a simple CR case study is presented here. Figure 5 shows the functional architecture of a transceiver. It is composed of a transmitter and a receiver sub-divided in two functions:

- A *Mapper/Demapper*: this function converts binary data into symbols according to the format of wished modulation. Within the framework of our application, three types of Quadrature Amplitude Modulation are implemented: BPSK, QPSK or 16QAM;
- A *Filter*: this function is a Finite Impulse Response (FIR) filter which filters the signal. Within the framework of our CR application two sets of coefficients are available for two frequency responses.

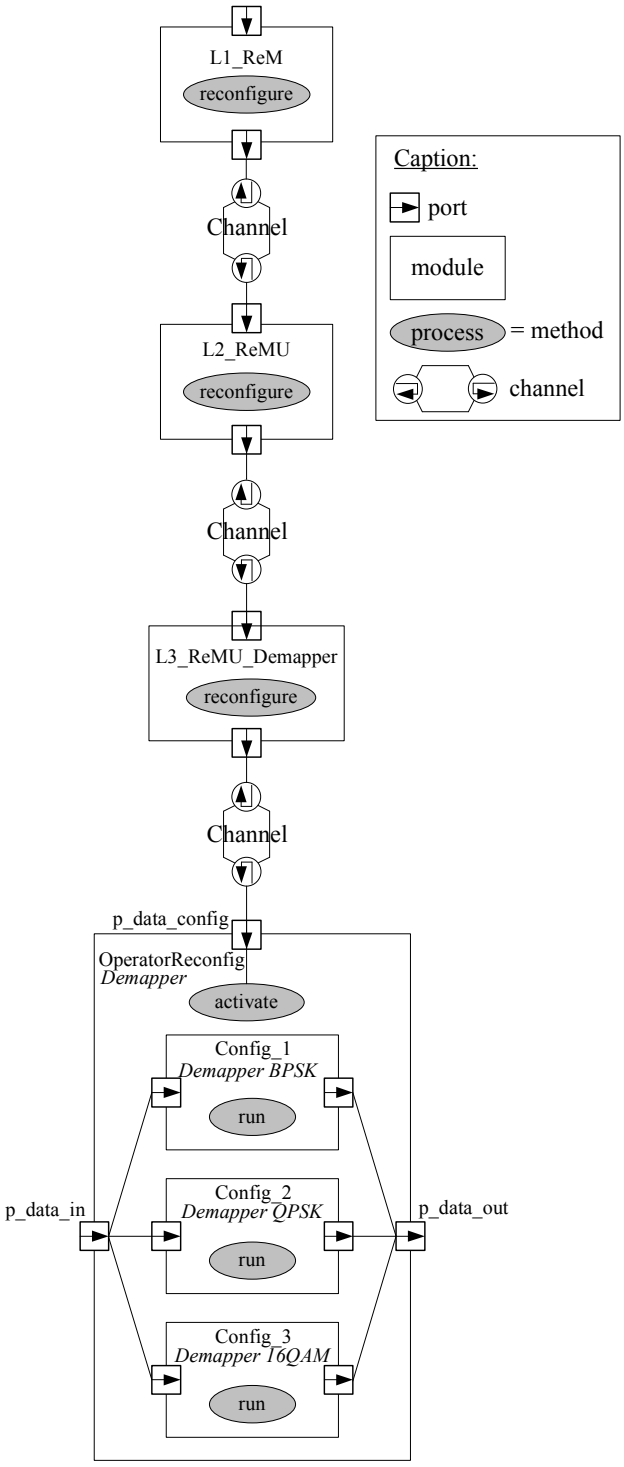

Figure 4. SystemC representation of a reconfigurable function with HDReM manager, extract of the SystemC model of the CR use case.

[1] A module in SystemC is the equivalent of class in object-oriented programming.

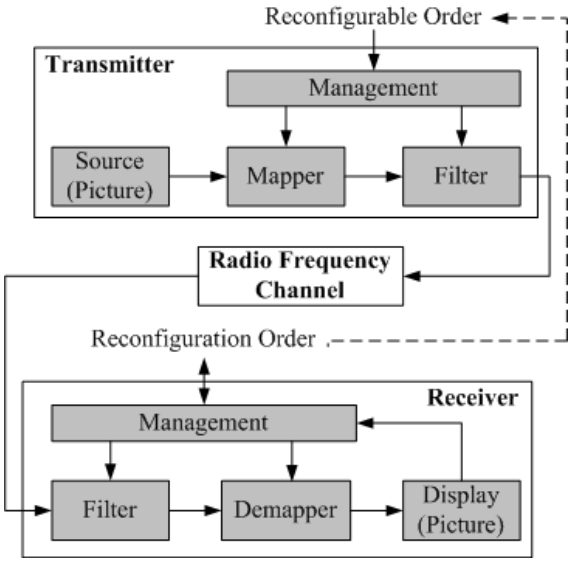

Figure 5. Functional architecture of the case study.

A binary signal representing picture is modulated in the *Mapper*, shaped by the *Filter* and sent through a Radio Frequency (RF) channel to the receiver.

In reception, the received signal is filtered and demodulated by the *Demapper*. Finally, the picture and an IQ constellation are displayed in a PC window (with OpenCV library).

The system user can change the modulation or/and the set of coefficients of the filter during the simulation while sending an order to the system. In a second step, HDReM has been extended for CR management (HDCRAM presented in [16]), in order to be able to capture metrics (for example, a SNR value in our CR use case) and to make a decision (analyze the SNR level) for reconfiguration (according to the level of the SNR).

A hardware implantation, on a Xilinx ML506 prototyping board, based on a Virtex V-SX50 FPGA, has been developed and succeeded in showing the FPGA sub-part implementation pertinence.

5.1. Modeling with UML

The modeling of the system respects the MOPCOM methodology and integrates the elements presented in section 3.1. This sub-section focuses only on functional modeling. The result of the modeling of the hardware platform (FPGA board of prototyping) is not presented in this article.

At the highest level of modeling in the MOPCOM methodology, the Platform Independent Model (PIM) is modeled, i.e. functional model of the system. Figure 6 shows an outline of the functional model (only for the transmitter). This one is independent of a target platform.

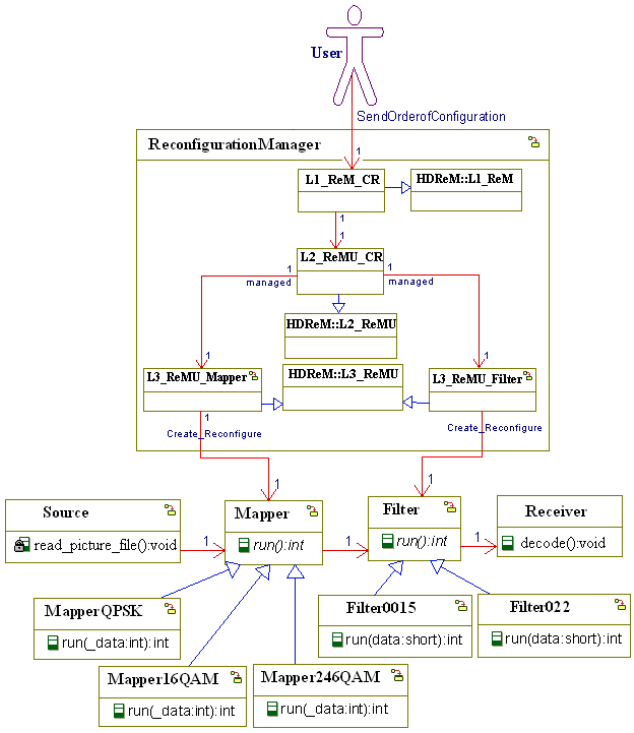

Figure 6. Class diagram of functional architecture of the Transmitter.

In this case study, the *Mapper* and the *Filter* functions have several operating modes (configurations). Thus the model of the functional architecture integrates an instantiation of HDReM architecture to manage the reconfiguration. Each function has a dedicated L3_ReMU unit which manages the reconfiguration process of the corresponding operator (*Mapper* and *Filter*). As the system is simple, only one instance of L2_ReMU unit manages the two specific L3_ReMU units. As in any system the L1_ReM unit is unique and is the general reconfiguration manager of the system.

5.2. Simulation and emulation in SystemC

An equivalent of the UML model previously presented had been developed with SystemC in Programmer View level (highest level of abstraction in SystemC). As explained in section 4, an adaptation was necessary (as shown in Figure 4). In a first step, a simulation on a single computer, without RF transmission, has been tested successfully. In a second step, a complete CR demonstrator was implemented with SystemC transmitter and receiver, and two USRP cards for radio frequency transmission. A more detailed description of this scenario is proposed on the Demonstration track of the conference. This represents some kind of emulation of the system, at a high level of description (so not real-time) but through realistic constraints such as RF and channel media.

6. CONCLUSION

In this paper an extension of MOPCOM co-design methodology for cognitive radio system has been presented. Polymorphism semantic and a generic architecture of reconfiguration management, for all types of reconfiguration on heterogeneous target, are employed to model reconfigurable system.

The future work also, consists in validating the UML modeling in Execution Modeling Level with SystemC PV-T execution in order to analyze the impact of the reconfiguration time on the scheduling of the system. After that, we would like to go to the implementation in reconfigurable FPGA after his modeling through HLS synthesis [20]. Another work consists in improving the integration between UML flow and SystemC simulation with a code generator from UML to SystemC.

7. ACKNOWELDGEMENTS

The MOPCOM methodology has been developed in the RNTL research program MOPCOM SoC/SoPC supported by the French National Research Agency (ANR – contract 2006 TLOG 022 01), the “cluster of clusters”, “Media and Networks” and the Brittany and Pays de la Loire regions. We also thank all participants for their contribution.

8. REFERENCES

[1] J. Mitola, “The Software radio”, IEEE National Telesystems Conference, doi: 10.1109/NTC.1992.267870, 1992.

[2] J. Mitola, “Cognitive Radio: An Integration Agent Architecture for Software Defined Radio”, Ph.D. dissertation, Royal Institute of Technology, Sweden, May 2000.

[3] S. Bhattacharyya, J. Eker, J. W.Janneck, C. Lucarz, M. Mattavelli and M. Raulet, "Overview of the MPEG Reconfigurable Video Coding Framework", In Journal of Signal Processing Journal, Springer, doi: 10.1007/s11265-009-0399-3, July 2009.

[4] A. Koudri, S. Metafli, and J.L. Dekeyser, “IP Integration in embedded system modeling”, in 14th IP Based SoC design conference (IP-SoC), Grenoble, France, 2005.

[5] C. Moy and M. Raulet, “High-Level Design for Ultra-Fast Software Defined Radio Prototyping on Multi-Processors Heterogeneous Platforms”, in Journal on Advances in Electronics and Telecommunications – Radio Communication Series: special issue on Recent Advances and Future Trends in Wireless Communications, vol. 1, n° 1, pp. 67-85, April 2010.

[6] G. Stitt, F. Vahid, and W. Najjar, “A code refinement technology for performance-improved synthesis from C”, In proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), ACM, pp. 716-723, New-York, USA, 2006.

[7] A. Sangiovanni-Vincentelli, L. Carloni, D. Bernadinis, and M. Sgroi, “Benefit and challenges for platform-based design”, In proceedings of the 41st annual Design Automation Conference (DAC), ACM, pp. 409-414, New-York, USA, 2004.

[8] OMG, “MDA Guide Version 1.0.1”, Report num. omg/2003-06-01, Object Management Group, 2003.

[9] S. Rouxel, J.P. Diguet, N. Bulteau, J. Carré-Gourdin, J.E. Goubard, and C. Moy, “UML Framework for PIM and PSM Verification of SDR Systems”, in proceedings of SDR Forum Technical Conference, Anaheim (USA), November 2005.

[10] A. Schallenberg, F. Oppenheimer, and W. Nebel,”Desining for Dynamic Partially Reconfigurable FPGAs with SystemC and OSSS”, Advances in Design and Specification for SoCs, Springer, pp. 183-198, 2005.

[11] C.H. Tseng and P.A. Hsiung, “A UML-Based Design Flow and Partitioning Methodology for Dynamically Reconfigurable Systems”, Workshop UML-SOC'06, San Francisco, CA, USA, July 2006.

[12] I. Rafiq Quadri, S. Meftali and J.L. Dekeyser, "High Level Modeling of Partially Dynamically Reconfigurable FPGAs with MDE and MARTE", Reconfigurable Communication-centric SoCs (ReCoSoC'08), Barcelona, Spain, July 2008

[13] A. Raabe and A A. Felke, “High-Level reconfiguration Modeling in SystemC”, Selected Contributions on Specification, Design, and Verification from FDL’08, vol.36, pp. 227-240, Springer, 2009.

[14] S. Lecomte, S. Guillouard, C. Moy, P. Leray and P. Soulard, “A Co-design Methodology based on Model Driven Engineering for SDR equipments”, in proceedings of SDR Technical Conference, Washington, D.C. USA, 2009.

[15] G. Erich, R. Helm, R. Johnson, and J. Vlissides, “Design Patterns: Elements of reusable Object-Oriented Software”, Addison-Wesley, ISBN 0-201-63361-2, 1995.

[16] L. Godard, C. Moy, and J. Palicot,"An Executable Meta-Model of a Hierarchical and Distributed Architecture Management for the Design of Cognitive Radio Equipments", Annals of Telecommunications, Special issue on Cognitive Radio, vol. 64, pp.463-482, number 7-8, August 2009.

[17] C. Moy, "High-Level Design Approach for the Specification of Cognitive Radio Equipments Management APIs", Journal of Network and System Management, vol. 18, n° 1, pp. 64-96, March 2010

[18] Xilinx, “Programmable Logic”, www.xilinx.com

[19] J. Delorme, A. Nafkha, P. Leray, and C. Moy , “New OPBHWICAP interface for real-time Partial reconfiguration of FPGA”, in proceedings of International Conference on ReConFigurable Computing and FPGAs (ReConFig'09), Cancun, Mexico, 9-11 December 2009.

[20] S. Lecomte, S. Guillouard, C. Moy, P. Leray and P. Soulard, “A co-design methodology based on Model Driven Architecture for Real Time Embedded systems”, in Journal of Mathematical and Computer Modelling: special issue on Telecommunications Software Engineering: Emerging Methods, Models and Tools, Elsevier, doi:10.1016/j.mcm.2010.03.035, publication in 2010.