From a Configuration Management to a Cognitive Radio Management of SDR Systems

Loïg Godard, Christophe Moy, Jacques Palicot

***Abstract*—This paper proposes a functional management architecture for Cognitive Radio systems. It relies on a previously defined configuration management architecture for multi-standard SDR systems, and complement it to support cognitive radio features. This paper explains the requirements of Cognitive Radio systems in terms of reconfiguration, smartness and sensing capabilities. A configuration management architecture capable of dealing with the hardware heterogeneity and a wide range of reconfiguration scenarios expected with SDR systems is presented. The management is distributed over the system and a hierarchical dependency is set on 3 layers, each having a different level of knowledge of the system and the associated hardware constraints of the elements it supervises. Then a cognitive management functional architecture is derived from the previous one, copying the 3 layers of hierarchy. The roles of the elements of each layer are discussed, as well as their respective interactions and their relationships with the elements of the configuration management architecture.**

***Index Terms*—cognitive radio, cognitive management, configuration management, multi-standard terminal**

I. Introduction

THE number of baseband processing contexts of a Software-Defined Radio (SDR) system [1][2] rises increasingly as the number of standards to handle in a single handsets increases. Consequently, configuration management becomes a mandatory part of a reconfigurable system [3]. But most of the proposals in reconfiguration management didn't consider both aspects of the reconfiguration: hardware reconfiguration capabilities, and reconfiguration schemes of application. It is necessary though to adapt reconfiguration mechanisms capabilities of hardware resources to the switching needs of the applications. This reconfiguration adequacy is essential to reduce reconfiguration overhead. We proposed in [4] a first step of such a global approach of reconfiguration management of multi-standard SDR systems. We concluded that reconfiguration management should be distributed and hierarchical, in order to support both HW heterogeneity and its associated reconfiguration capabilities on the one hand, as well as the reconfiguration schemes induced by SDR usage scenarios on the other hand.

As detailed in [5] a CR equipment should take decisions based on information given by sensors on the one hand and on rules and self learning capabilities on the other hand. These considerations have direct consequences on the hierarchical configuration management of [4]. In fact each management level should be driven by an associated cognitive module. That is why we propose, in this paper, a distributed hierarchical cognitive reconfiguration management for SDR systems.

The remainder of the paper is organized as follows. In section II, we describe properties of Cognitive Radio systems. Then we deal in section III with the hierarchical configuration management architecture. In section IV, we describe our proposal for a hierarchical cognitive management mapped on the previous configuration management architecture. It is particularly stressed how these two hierarchical approaches are inter-related. Finally, in section V we conclude this work.

II. Cognitive Radio Systems

A. A reconfigurable radio with sensing means

The Cognitive Radio concept, here tackled in its widest and most unrestricted sense, refers to a communication system that is able to observe its environment, analyze it, and react to it in some way, as illustrated in Fig. 1.

A CR system is a smart SDR system. The target of multi-standard radio is of particular interest for this approach. In addition to its multiple communication parts, the CR system indeed features sensing means, adapting means and a smart sub-system dedicated to analyzing *stimuli* and making decisions [6].

Sensing means refer to all the possible methods the CR system has at its disposal for observing its environment, which can be categorized in four main families described below:

- *electromagnetic environment*: spectrum occupancy, Signal to Noise Ratio (SNR), multi-path propagation, etc.
- *hardware environment*: battery level, power consumption, computational resources load, etc.
- *network environment*: telecommunication standards (GSM, UMTS, WiFi, etc.), operators and services available in the vicinity, traffic load on a link, etc.
- *user-related environment*: position, speed, time of day, user preferences, user profile (access rights,

Manuscript received February 28, 2006.
L. Godard, C. Moy and J. Palicot are with IETR, UMR CNRS 6164, Automatic and Communication lab / Supélec SCEE team, Rennes, France (+33 2 99 84 45 00; fax: +33 2 99 84 45 99; e-mail: loig.godard@supelec.fr, christophe.moy@supelec.fr, jacques.palicot@supelec.fr).

contract…), video and audio sensor (presence detection, voice recognition), etc.

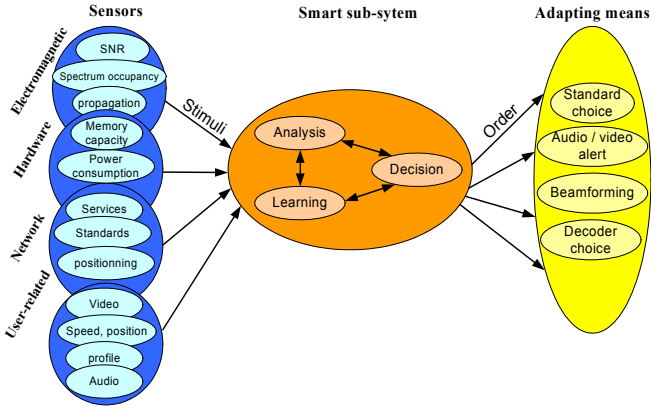

Fig. 1 - Schematic functional view of a Cognitive Radio system

As shown in Fig. 1, all the stimuli received by the sensors from the different layers (from physical to application) provide metrics that must be merged and analyzed jointly in order to reach the best possible reconfiguration decision. Reconfiguration is supported by a software radio or at least a SDR architecture, which integrates both adequate HW capabilities and SW architecture. Our previous research lead to the definition of a hierarchical configuration management [4], whose major features are recalled in part III.

B. *Smartness*

To make a reconfiguration radio become a CR, it is necessary to add some smartness in order to make it self-aware [5]. We assert this smartness itself needs also to be distributed all over the radio system. It may be foreseen indeed that a CR will have to face high-level decisions as well as local decisions. High-level decisions are to be understood in the sense that network considerations or constraints on the overall CR system may take benefit from reconfiguration. This is what is considered in End-to-End Reconfigurability [7]. Local decisions may concern for instance the reconfiguration of a single digital processing block unit (PBU) in order to adapt its behavior to a specific context (increase or decrease the number of fingers of a RAKE receivers depending on the number of path of the propagation channel). But high-level decisions moreover should impact the low-level layers of the system through the low-level decision entities. Which leads to the notion of a hierarchical repartition of the smartness through the CR system.

In a CR system, it is consequently straightforward to merge the cognitive management we consider here, with the previously defined configuration management of part III, due to their identical features in terms of distribution and hierarchy. This paper is all about describing the combination and the relative interactions between them.

III. Reconfiguration Management Architecture

The great diversity of processing types in a multi-standard application implies a large number of processing configurations to be managed. The configuration management is complex and we believe that a hierarchical approach of configuration control and management could simplify it [4].

The benefits of clustering the transmission chain have been presented in [8], particularly to manage partial reconfiguration of an application. In addition to this, the proposal of a hierarchical view enables to manage multi-granularity of configurations, which is of particular interest for heterogeneous architectures. The proposed model is composed of 3 levels of hierarchy detailed in Fig. 2. A system architecture compliant to this functional model includes one *Configuration Manager* at level 1 (L1_CM), several *Configuration Manager Units* at level 2 (L2_CMU), each of them being responsible of one or several *Configuration Manager Units* of level 3 (L3_CMU), which directly manage the processing components.

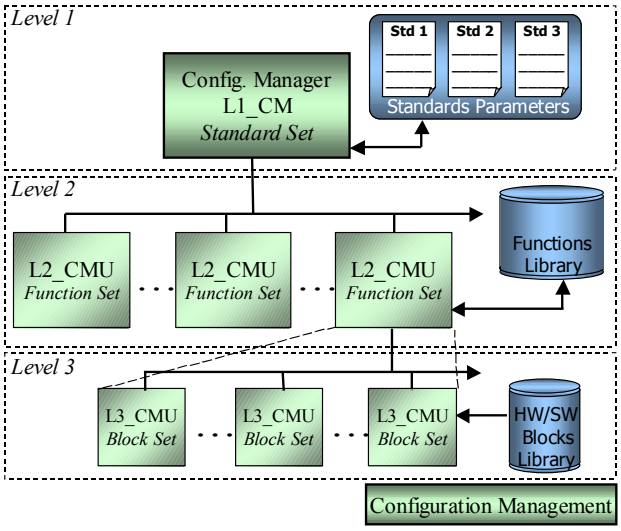

Fig. 2 – Hierarchical distributed configuration management functional architecture

A. *L1_CM*

There is one L1_CM in the system. L1_CM is the orchestra's chief. It has the global knowledge of the state of the system thanks to databases. It can update these databases with the information it receives from the network on the one hand, and from the system itself on the other hand. In the latter case, reconfiguration procedure comes up to L1_CM as soon as several L2_CMU are concerned. The L1_CM watches over the coherency and the synchronization of the orders between the involved L2_CMU.

L1_CM gives *abstract orders* of reconfiguration to the underlying L2_CMU when this is needed, as for a standard hand-over for instance. "*Abstract orders*" here is to be understood as orders that are completely independent from the nature of the HW processing devices they will impact. The only concern of L1_CM is to know where are the L2_CMU

(on which HW devices) and how to reach them (through which method and communication media).

B. L2_CMU

The generic functions selected in level 1 through *abstract orders* are specialized at level 2 in accordance with standards specifications. The set of attributes of each function is handled by the L2_CMU to create each functional context of the entire processing chain. For example, a generic function "bit to symbol mapping" is set at level 2 to fit a selected standard. In the case of 802.11g standard this setting could be BPSK, QPSK, 16-QAM, 64-QAM constellation. At this level, in accordance to performance requirements of the function, the execution target is chosen. But the L2_CMU does not keep in charge the effective instantiation of the context onto the hardware. Here, functions are handled through a few attributes to completely characterize them.

C. L3_CMU

The processing datapath architecture supporting the PBUs depends on the reconfigurable computing resources of the hardware architecture. The complete processing path could be formed by different types of reconfigurable resources such as digital signal processors, FGPA, and inside FPGA: configurable accelerators, array of DSP blocks or a fine grain reconfigurable datapath. The main task of the L3_CMU in the configuration path is to find available processing resources and configure them to enable the execution of the functional context created at the superior level. Clock management of the resources is also included as an important feature of the configuration manager to reduce power consumption and to provide a efficient management of the multi-rate applications.

IV. Cognitive Radio Architecture

The Cognitive Radio architecture is an extension of the configuration management architecture towards a Cognitive Radio global management.

A. Distributed Hierarchical CR architecture

The distributed hierarchical management of part III enables to reconfigure the system with adequate features in answer to the constraints imposed by the scenarios of use foreseen for such systems [4]. But it supposed that the reconfiguration orders only come from outside the system. There is a need to include smartness in the system itself. This is all about CR. As stated in II.B, this smartness has to be distributed inside the CR system, as well as the configuration management. Consequently, an identical approach has been used for the CR architecture. Fig. 3 shows the direct correspondence set between the reconfiguration management and the cognitive management:

- L1_CM → L1_CR (level 1 – *Cognitive Radio*)
- L2_CMU → L2_CRU (level 2 – *CR Units*)
- L3_CMU → L3_CRU (level 3 – *CR Units*)

The CR architecture should be capable of taking decisions at any level of the system, under a global supervision. It also should be aware about the reconfiguration capabilities of all its different elements at both HW and SW levels. The advantage of this hierarchical structure is to permit to take into consideration many parameters of different kind at any level of the system on the one hand, and to impact the system in terms of reconfiguration either locally or at a large scale on the other hand.

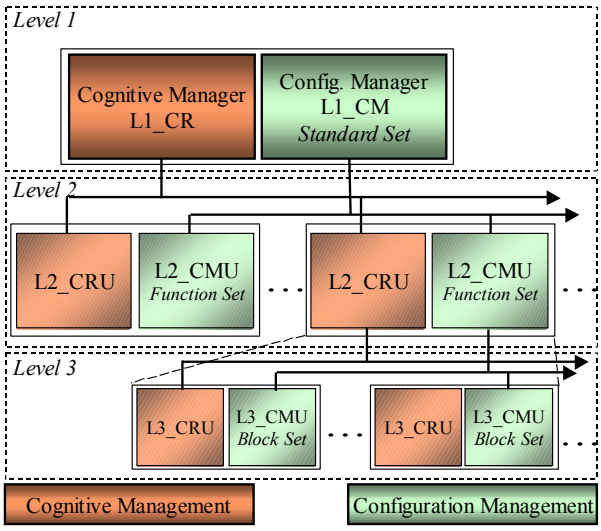

Fig. 3 – Hierarchical distributed Cognitive Radio management functional architecture

This hierarchical distributed architecture enables a wide range of possible scenarios. From global Quality-of-Service (QoS) considerations at the system level, to very local changes for bug-fixing or optimization purposes as instance. Let us consider the example of a large scale scenario: battery life extension. The distributed smartness reports to the L1_CR, through each L3_CRU, the status of power consumption of each of the PBUs of the system. Combining this information with others (concerning the QoS and the battery life duration) obtained through other elements of the CR architecture, the L1_CR takes appropriate decisions in terms of reconfiguration, in order to increase battery life duration for instance, while maintaining a targeted QoS. L1_CR gives this order to L1_CM that makes the necessary repercussion through the configuration management architecture of part III.

Smartness may vary from very simple reactions such as a threshold detection and associated response (referenced in a pre-defined table) up to very complex scenarios. The complexity depends on the number of metrics to take into consideration and to fuse, as well as the decision engine implemented. The following paragraphs detail the behavior of the elements of the cognitive architecture that make these scenarios feasible. We opt in the following for a bottom-up description.

B. L3_CRU

L3_CRU role is firstly to capture sensing information and translate it into metrics. Metrics can either be computed by the L3_CRU itself or directly be provided by the PBU it is associated to. In this last context, the L3_CRU only receives the result form the PBU. Otherwise the L3_CRU receives the necessary information from the PBU (if not data themselves) which permits to obtain the metrics. L3_CRU operation may vary from a very simple data capture to a very complex signal processing procedure. The implementation complexity of the L3_CRU is consequently very variable.

Thanks to the distribution of the smartness through the entire CR system, the L3_CRU may react in two different ways, which permits to deal with any situation the CR system has to face, as illustrated in Fig. 4.

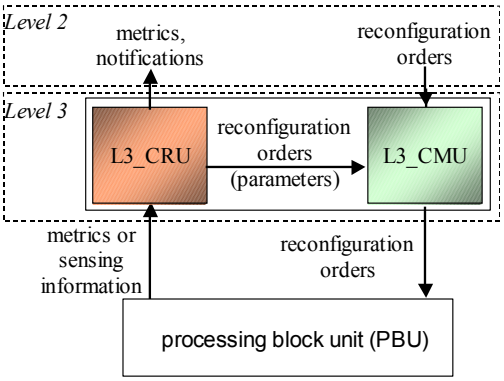

Fig. 4 – L3_CRU and L3_CMU message exchanges to manage a PBU

On the one hand, the sensing information captured by a L3_CRU is known to have impacts at the only level of the PBU it manages. It is no use to propagate the information through the entire system. L3_CRU acts locally, takes reconfiguration decision, interprets the metrics and converts them into parameters it sends to L3_CMU, that makes the reconfiguration as in III.C. L3_CRU notifies to higher layers of the CR structure the changes that were done. Additionally, it is the role of the L3_CRU to verify the PBU behavior and inform higher CR layers of its status (error, stand-by, run, etc.). If the impact is larger, on the other hand, L3_CRU sends the metrics to L2_CRU. If all the PBUconcerned by the reconfiguration are under the same L2_CRU responsibility, L2_CRU plays the roles of L3_CRU in this specific context (takes reconfiguration decision and notifies to L1_CR). Paragraph IV.C explains in detail how L2_CRU acts.

Note that the impact of a sensing information is *a priori* known. It is clear that depending on its nature, each *stimulus* has one or several well-identified causes and consequences (maybe combined with other *stimuli*). So that the reconfiguration procedures that could be provoked, as well as their range in the system are pre-defined. Another important point to stress is that the information sent from the lower CR layers to the higher layers have to be saved in databases that content the state of the system. Most of them will be located at the L1_CR level, but it can be considered to have databases accessible at L2_CRU level.

As previously mentioned, the L3_CRU must be able to take decisions for a local impact of the sensing information. As stated for the metrics computation, the decision procedure may range from very complex to very simple processing. The following examples may be given, but the list is not exhaustive, in increment of complexity:

- a simple threshold,
- a state machine,
- neural networks,
- optimization heuristics (genetic, etc.).

This range of decision procedures is also met at the higher levels L2_CRU and L1_CR. The higher the level, the more complex algorithms maybe expected. The reason is that more metrics are to be taken into account and combined, but also the processing devices are more suited to that kind of processing. Note that training techniques may also be implemented.

C. L2_CRU

L2_CRU gathers the metrics coming from the L3_CRU it supervises, as described in IV.B. Two situations may occur. Either it transmits the metrics to the L1_CR if these metrics may contribute to a decision of higher level than those it can take by itself. It may by the way update L2_databases if necessary. And/or it converts the metrics coming from the bottom in abstract reconfiguration orders (abstract in the sense that not HW-dependent) for a delivery to the L2_CMU. L2_CRU notifies the L1_CR of the reconfiguration it ordered.

Then L2_CRU supervises the correctness of the L2_CMU reconfiguration orders thanks to all the L3_CRU reporting the behavior of the PBU that depend from this L2_CRU/L2_CMU couple.

D. L1_CR

L1_CR is the supervisor of the cognitive sub-part of the system. It is the intelligence that can take initiatives depending on the environment *stimuli*. These *stimuli* are translated into metrics. They can come from the bottom hierarchy of the CR system (L2_CRU and L3_CRU), namely from the processing part of all layers of the OSI model. The PBU composing the processing datapath may indeed process physical layer, as well as protocol, or even application layer operations. But this is also the L1_CR role to receive the metrics from the network side. There is a heavy debate on the repartition of the smartness between the network and the access point or terminal CR system. Our proposal can cope with all possible repartition of the smartness. Either L1_CR processes itself raw data and extracts the corresponding reconfiguration information to be delivered to the L1_CM, or it directly transmits to the L1_CM the pre-processed information it receives. Nevertheless, user-related information are under the responsibility of L1_CR, as well as other *stimuli*, that are not directly extracted from the signal processing of the radio protocol stack: battery level, time-of-the-day, air temperature,

etc.

The actions taken by the L1_CR can be of very different range and complexity, depending on the nature of the system (terminal, PDA or even bigger), and its degree of smartness. L1_CR is in charge of the multi-standard reconfiguration, and of any reconfiguration whose range concerns several L2_CMU. It also supervises the correctness of the reconfiguration it initiated. Such large scale reconfigurations (at the standard level) must respect extremely hard constraints in terms of synchronization between the PBU, time of reconfiguration, etc. It still seems an impossible challenge to really perform a real inter-standard soft-handover. But if it will be possible one day, a complete scenario must be played in the L1_CR, going through steps with the two standards running at the same time during a small portion of time at least. So that the two standards could catch the application at the very same time in order to maintain a continuity of the service.

The decision at L1_CR may be triggered by various kinds of events, depending on the scenarios. All can not be anticipated. At least arrays of different operating mode (very good, good, energy saving mode, etc.) must be considered, so that the system can reconfigure itself according to its environment. The trade-off is to be done between the two following extremes. Either be very directive with all the supported scenarios in pre-defined databases. It is possible to consider they may be updated. Main advantage is that each scenario may have been pre-tested. Or a certain autonomy is let to the system. The degree of supervision of the user or the manufacturer/network/operator has to be defined.

E. Li_CR(U) and Li_CM(U) interactions

Fig. 5 aims generalizing the interactions described separately for the configuration management and the cognitive management in the previous sections. The Li_CM(U) may be activated either from the superior level(s) of the configuration management, or from the Li_CR(U) of the same level. Li_CM(U) manages reconfiguration and propagates reconfiguration orders down to the lower level, in order to do the reconfiguration of the PBUs.

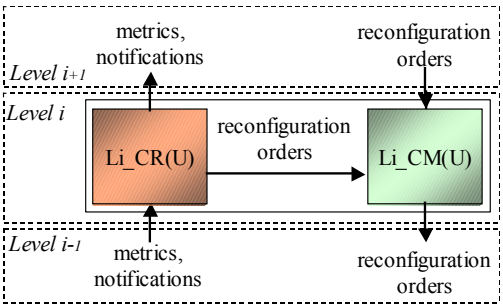

Fig. 5 – Generalized view of the interactions between the CR and CM mangement.

The Li_CR(U) receives or compute metrics extracted from *stimuli* of different kinds or from the inferior level of cognitive management, and decides to send the information either to the superior level Li+1_CR(U) or to the Li_CM(U) depending on pre-defined rules, that maybe static or computed on-the-fly. The Li_CR(U) also send notifications (of behavior accuracy for instance) to the higher level of CR management. So, we observe a bottom-up metrics and notifications flow in the cognitive management and a top-down configuration flow in the configuration management.

V. Conclusion

The functional Cognitive Radio architecture proposed in this paper relies on the configuration management architecture we already defined for SDR systems, and which is currently under implementation. This prototyping at the configuration management level strongly orientates our investigations towards cognitive management, which is a guarantee of the viability of our solution for the implementation the CR concepts. The originality of this proposal is due to the effective separation of the configuration and the cognitive management of a SDR system, while keeping a strong relationship between them. Respective roles and duties are well identified, as well as their interactions and interfaces with the other entities.

Future papers will more precisely detail the intimacy of the architecture, and illustrate its mechanisms through examples of cognitive radio scenarios. Many points still remain to be investigated or at least better defined. Let us just stress the repartition and the content of the configuration and knowledge databases. Configuration databases are the repository of the necessary pieces of binary code (for processors) or bitstream (for FPGA) for the instantiation of the standards supported by the system. Knowledge databases contains the metrics from which the cognitive management takes its decision. The management of the decisions itself is a wide area we plan to address in the future.

References

[1] Joe Mitola, "The software Radio Architecture", *IEEE Communications Magazine*, May 95, pp. 26-38.

[2] Walter Tuttlebee, "Evolution of radio systems into the 21st century", *Proc. IEE Int. Conf. on 'Radio receivers and associated systems'*, 1995.

[3] Apostolos Kountouris, Christophe Moy, "Reconfiguration in Software Radio Systems", *2nd Karlsruhe Workshop on Software Radios*, Mar. 2002.

[4] Jean-Philippe Delahaye, Jacques Palicot and Pierre Leray, "A Hierarchical Modeling Approach in Software Defined Radio System Design", *IEEE SIPS*, Athens, Greece, 2-4 Nov. 2005

[5] Joe Mitola, "Cognitive Radio – An Integrated Agent Architecture for Software Defined Radio", thesis dissertation, 8 May 2000.

[6] Christophe Moy, Alexis Bisiaux, Stéphane Paquelet, "*An Ultra-Wide Band Umbilical Cord for Cognitive Radio Systems*", PIMRC'05, Berlin Germany, September 2005.

[7] Markus Dillinger, Nancy Alonistioti, Nikolas Olaziregi, Thomas Wiebke, "Network Functions Supporting Reconfiguration in B3G Environments", *Frequenz 58*, May 2004, ISSN 0016-1136, pp. 121-125.

[8] Jean-Philippe Delahaye, Christophe Moy, Pierre Leray, Jacques Palicot, "Managing Dynamic Partial Reconfiguration on Heterogeneous SDR Platforms", *SDR Forum Technical Conference*, Anaheim, USA, 15-17 Nov. 2005