A RF HARDWARE ABSTRACTION-BASED METHODOLOGY FOR FRONT-END DESIGN IN SOFTWARE-DEFINED RADIOS

Sabeur Lafi; Ammar Kouki; Ahmed Elzayat and Jean Belzile
(École de technologie supérieure, Montréal, Québec, Canada;
{sabeur.lafi.1, ahmed.elzayat.1}@ens.etsmtl.ca, {ammar.kouki, jean.belzile}@etsmtl.ca)

ABSTRACT

In the recent years, Software-Defined Radio (SDR) has gained increased attention from the scientific and industrial communities. Software components, baseband architectures and a myriad of design approaches were proposed in the perspective of realizing reconfigurable and interoperable devices. Open standards such as the Software Communication Architecture (SCA) were proposed to enable waveform porting and SDR interoperability. They included various abstraction concepts such as abstraction layers, middleware frameworks and hardware drivers. However, this effort was mainly focused on the baseband side. Little effort was deployed in the SDR radiofrequency (RF) / analog side, especially on the design level. In this paper, we investigate the challenges in RF circuitry which make front-ends' design methodologies awkward for SDRs. We propose a new design scheme based on a hardware abstraction strategy as a response to this issue. Finally, we present a case study in which the proposed design scheme will be used to design a RF filter in a step-by-step fashion.

1. INTRODUCTION

The crucial need of the military, police and firefighters for professional mobile radios allowing real-time and unlimited interaction between these forces on the ground had emerged the concept of full interoperable and reconfigurable radios. Such a radio is intended to be a multi-standard, multi-channel and multi-service device providing services transparency, immediate reconfigurability and components exchangeability. This implies a proper unification of communication systems and new mechanisms enabling reconfigurability, interoperability and full porting of communication standards between the different hardware platforms. In addition, the current state of technology still hinders the design of fully reconfigurable hardware. For example, the resolution of the analog-to-digital (A/D) converters is limited [1, 2]. Radiofrequency front-ends offer little margins of reconfigurability which needs a broadband and linear frequency response as well as a wide dynamic range to enable multi-standard designs. In the digital side, most of handset designers use communication-dedicated hardware to implement baseband algorithms such modulation / demodulation, equalization and digital filtering. This design choice was somewhat mandatory to cope with antagonist factors such as (i) requirements in terms of processing delay, bit-error rate, etc. (ii) the size, weight and power (SWaP) constraints and (iii) the market pressure in terms of cost and time-to-market (TTM).

Historically, it was progressively shown that using dedicated hardware is not the best solution for reconfigurable radio design, especially in multi-standard context [3]. The advances in software design and digital processing suggested the implementation of baseband operations (e.g. signal processing) as software components which can theoretically run on different hardware platforms. This was the inception of the *software radio* (SR) concept.

In fact, various definitions are provided in literature for the term software radio. The most common one is the following: "*A software radio refers to a device fully reconfigurable using software at any level of radio protocol stack. It implicitly supposes that A/D conversion is carried out at the antenna*" [4]. However, the practical implementation of such a device is only possible at very low frequencies due to serious technology limitations. For this reason, another paradigm emerged namely *software-defined radio* which refers to a presently realizable version of a software radio. The idea is to limit the frequency range at the antenna using analog filtering to overcome the A/D conversion limitations while baseband processing is intensively carried out by software. Communication standards are implemented as waveforms including the modulation / demodulation, coding, access and duplex modes as well as the protocol structure of transmission / reception methods [5].

To make waveforms portable between different hardware platforms, initiatives such as the *Software Communications Architecture* (SCA) have been proposed to provide an abstraction layer allowing software components to run independently from the underlying hardware. Furthermore, SCA enables interoperability between SDRs using Common Object Request Broker Architecture (CORBA). The *Government Reference Architecture* (GRA) is another initiative to enable modular software design for above 2 GHz (A2G) SDRs [6].

While most of efforts in both academia and industry, involved in the promotion and development of the SDR paradigm had a particular interest in hardware abstraction to enable waveform porting and interoperability between SDRs as well as the reconfigurability using over-the-air downloadable software components, little effort was

deployed to review the design of the radiofrequency front-ends in charge of the wireless communications. In fact on the digital front, hardware architectures such as digital signal processors (DSP), Field-Programmable Gate Array (FPGA) and General-Purpose Processors (GPP), had been deeply investigated [7]. Middleware frameworks, drivers and abstraction layers were developed to enable the porting of software components such as waveforms, user applications and other services between these platforms [8]. However on the analog front, design of RF front-ends is still carried out using classical techniques and offer little reconfiguration and interoperability. No design methodologies supporting hardware abstraction are currently in use. Most design schemes are still too technology-dependent to enable neither easy technology insertion nor efficient component exchangeability.

In this context, we investigated the matter of hardware abstraction in RF design in order to elaborate effective proposals enhancing the design of RF front-ends for SDR. We finally proposed a new design scheme based on the hardware abstraction. In this paper, we first review the basic abstraction concepts already developed in SDR's digital side. Then, we review the design challenges in RF/analog domain that require an effective response in order to enhance front-ends' design and reconfigurability. In the fourth section, we present the new design scheme and its key abstraction concepts. Finally, section 5 presents a step-by-step case study demonstrating how a RF filter can be designed using the proposed design scheme.

2. ABSTRACTION CONCEPTS IN SOFTWARE-DEFINED RADIO DESIGN

The term Software-Defined Radio was invented by Joseph Mitola in 1991 [9] who was the first to propose the architecture of a radio whose baseband operations are entirely carried out by software [9]. It is intended to push the transition between hardware and software as close to the antenna as possible [2]. The benefits of a SDR are numerous: it offers an unprecedented level of flexibility. Waveforms are easy to change, fix or upgrade. They are also portable and cost-effective. Multi-mode radios become easy to implement because many channels and standards can be supported simultaneously via the execution of concurrent waveforms [2]. Additionally, the time required for the design, implementation, testing and upgrade of SDRs is reduced. All these benefits are the result of using hardware abstraction layers and middleware allowing the masking of underlying hardware and enhancing the portability of both waveforms and hardware. In this section, we study some examples of the abstraction mechanisms used in SDR design. It is worth noting that it is not an exhaustive list, but rather an illustrative one.

2.1. A Typical Software-Defined Radio Architecture

[10] presents a typical SDR architecture for a multimedia multi-standard mobile handset. As shown in Figure 1, this architecture is composed of several layers:

- *System Hardware:* This layer consists of the physical hardware to be used for signal processing and data handling. It can use DSPs, FPGAs and / or GPPs as well as memory blocks, etc.
- *Real-time Operating System (RTOS) and hardware drivers:* This layer serves as an intermediate between the upper layers and the underlying hardware. It provides controlled access to the system resources such as memory banks, timers and processors.
- *Software and hardware abstraction layers:* These layers offer uniform access functionalities and procedures to the system resources for the upper layers (especially applications).
- *Framework:* acts as a "presentation" layer. It ensures that the system has the required capabilities (e.g. data transfer capabilities, synchronization functionalities, etc.) before proceeding further in the processing of high-level requests.
- *Application and services layer:* It provides all the functionalities needed for the modulation / demodulation, equalization, digital filtering, etc. It may be an internally layered layer especially in multi-standard context.
- *Human Machine Interface:* It is the highest level layer and allows the interaction with the end user (presentation of the received data, reception of the user commands, etc.).

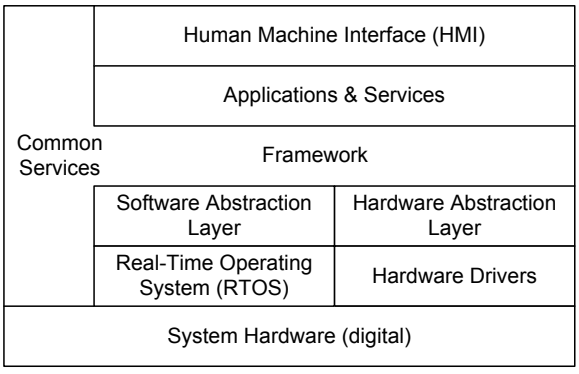

Figure 1. SDR block diagram of a multi-standard mobile handset [10]

This example of an SDR block dedicated to a multimedia multi-standard mobile handset shows the importance of abstraction layers either software or hardware. Three types of abstraction layers are presented: (i) a "presentation"

framework that controls certain parameters related to the data and performance and ensures that high-level requests are coherent with the current system capabilities, (ii) Hardware / software abstraction layers which mask the physical details of the underlying hardware and its management by the RTOS (access allowed via the OS primitives, APIs and interfaces) and (iii) Hardware drivers which provide the basic primitives to get access to specific peripherals.

As one may notice, the abstraction layer concept aims to enable the independence of the applications and services from the underlying hardware. However, the portability of software components may be limited only to some platforms. The interoperability is also not ensured because different SDRs may not be able to communicate. An obvious solution is the standardization of the abstraction layers in order to make software (respectively hardware) components from various origins executable on (respectively compatible with) different hardware platforms.

2.2. Software Communication Architecture

In order to make waveforms portable across platforms and SDRs interoperable, the Joint Tactical Radio System (JTRS) initiative was proposed for the purposes of radio communication systems unification, services' transparency and components' exchangeability. This initiative led to the elaboration of the software communication architecture. The SCA is an open framework instructing SDR engineers how the hardware and software blocks have to act together within the JTRS [5]. In particular, the SCA describes how a waveform has to be implemented on a given hardware platform. It defines also the software structure and especially defines the interfaces within an SDR.

The SCA core framework works within the JTRS operating environment (OE) which also includes a real-time operating system and an object request broker (ORB). This open framework establishes a coherent abstraction paradigm by the strict separation of software and hardware domains. It defines the interfaces between them which facilitate the development of either software or hardware by multiple vendors. It supports standardized APIs developed for the purpose of enabling applications porting and devices exchangeability. Interoperability between SCA-compliant SDRs comes from the use of object request brokers allowing the exchange of objects as well as the access to distributed software and hardware resources independently from the platform.

While the SCA offers a rich framework providing basic requirements for SDR development, it suffers from some lacks: it targeted systems ranging from low frequencies to approximately 2 GHz. It also faced a limited success in both industry and academia. In the technical side, the porting costs using the SCA are surprisingly not negligible. Moreover, it does not address the important architecture and design issues for waveform portability. For example, the SCA does not also specify any requirements or particular guidelines about the underlying hardware architecture such as physical interconnections, processing units, timers, etc. [11]. These SCA limitations pushed the US military to adopt a novel design framework for SDRs.

2.3. Government Reference Architecture

While the SCA provides several guidelines about how the software components should be designed and developed for an SCA-compliant SDR, it does not address the overall communication system. In particular, it has no specific instructions about the underlying hardware architecture. Consequently, the waveforms portability is expensive and devices exchangeability is not realized as anticipated. As a result, the Government Reference Architecture (GRA) was proposed.

The GRA is a software / firmware framework approach for above 2 GHz military satellite communications' terminal design and development. This framework aims to establish modular and open system architectures to maximize modules integration from various vendors as well as endorsing design flexibility, high portability and interoperability. It targets also the reduction of costs during the development, integration and upgrade phases [6].

While it is still a classified framework, public data shows that the GRA is a part of the Modular Open-Systems Approach (MOSA) which aims to modular system design, has well-defined interfaces and supported by common-used industry standards [12]. The GRA defines an Open Systems Interface (OSI) which is a software layer with an associated cooperative hardware architecture responsible for system control, data flow and traffic management [6]. It also defines a "middleware layer" that consists of a standardized terminal backbone and a Common Integration Layer (CIL) [6]. The CIL is simply a set of interfaces and APIs used to develop an open system architecture [6].

The GRA defined also the Modem Hardware Abstraction Layer which is an API specifically designed to enhance waveforms portability. It is intended to abstract the modem hardware and radio operating environment from the waveform applications to provide suitable host environment across platforms [13]. It facilitates the reuse of software / firmware modules in order to build across-platform physical radio system on which it can run [14].

In addition, the GRA defined a set of APIs and standards to handle both software and hardware design. The definition of suitable interfaces between components, strict partitioning of radio system and the abstraction of underlying hardware enhance the portability of waveforms, the reuse of modules and also facilitate the technology insertion.

3. CHALLENGES IN RF/ANALOG DESIGN

If salient efforts had been sustained for years in the SDR baseband side where communication standards and open APIs were implemented as software components in order to alleviate waveform portability, reuse and upgrade and bolster interoperability and reconfigurability, little efforts were deployed in the RF front-end side were circuitry design methodologies are still too technology-dependent to provide neither reconfigurable nor flexible front-ends.

Despite the advances in computer-aided design (CAD) tools and manufacturing platforms, manufacturers of RF systems and subsystems are experiencing an increasing pressure. The first reason is economic: Market trends are challenging fueled by new services and requirements claimed by both end users and network operators. Therefore, it has become necessary to update RF design, implementation and testing techniques and tools in order to address market and emerging applications' needs. The second reason is technical: Despite the fact that time-to-market and engineering costs are crucial factors, the current design and manufacturing procedures offer little to no design/product reuse possibilities, which limits efficiency.

A wireless radio system such as a SDR consists of two main building blocks: The first is digital or baseband part, where signal processing, internetworking and user applications are handled. The second is the RF front-end responsible for analog transmission and/or reception of radio signals carrying data using electromagnetic waves. Bringing emerging concepts such as end-to-end reconfigurability, efficient waveform portability, interoperability and scalability to wireless systems requires that both building blocks be able to meet the requirements of reconfigurability.

On the baseband side, one can argue that baseband subsystems (including both software and hardware) can meet such demanding requirements. This is because of the abundance of advanced CAD environments that include automatic synthesis tools and hardware abstraction strategies as well as the availability of modeling, hardware description languages and protocol layer standards. Hence, all of these instruments constitute a viable framework helping effectively the development of baseband systems that satisfy the requirements of next-generation networks and fit specifically with open wireless architectures and platforms. In fact, the effort of hardware abstraction had particularly remarkable effects in the evolution made in the baseband side. Indeed, today hardware abstraction is widely exploited in digital systems. It allows the designer to focus on the functionality and the related effects rather than the details of the underlying hardware. In doing so, it virtually masks the physical details of the intrinsic architecture [15], [16]. In fact, hardware abstraction has been boosting the industry of very large scale integration (VLSI) devices and especially general-purpose processors. It led to the birth of a hierarchical design approach in digital circuits that replaced the old techniques that were not amenable to design automation. It thus helped to reduce design complexity and created effective modularity. In the digital world, at each design level, the internal details of a complex module are replaced by a black box view. This black box is described by a model with known characteristics and contains all the information needed to deal with the block at the next level of hierarchy. This approach has led to the emergence of powerful CAD tools working with cell libraries. These libraries contain the technological information about the cells and atomic devices of which the module is composed. This concept finds its roots in the software design where libraries of software routines are used. Any software program can use the routines of these libraries without looking inside their structure or how they were implemented. It only cares about the intended result when the right input / output parameters are provided [17].

By contrast, RF front ends and subsystems had not seen the same rate of advances and changes. Despite notable advances in design tools, RF front ends' design procedures are still almost fully manual and are carried out according to old design schemes and flows. Actually, these conventional strategies are too costly, long and do not enable easy technology insertion. RF devices and components handle mainly analog signals. They have several particularities such as the sensitivity to the noise level, frequency and temperature drifts and may behave in non-linear mode where additional noise, interferences and signal distortion may occur. Moreover, the developed architectures are often non reusable. The changes and corrections of the design according to new specifications are very expensive and may take a lot of time. Final system integration is tedious, risky and slow particularly when different technologies are involved in the system architecture. The classic RF design scheme, either top-down- or bottom-up-like, lacks formal communication rules between the different developers involved in the project. Interpretation errors and forgotten specifications can easily happen and have negative effects on the designed system performance, cost and also the time-to-market factor [18]. In addition, RF design is not only non-adaptive but also very technology-dependent. Available RF technologies and even CAD tools often impose rigid constraints to achieve a given RF functionality. As a result, classical RF design is unable to ensure rapid design, prototyping and integration. It does not allow flexible and fast adaptation of CAD tools, technology libraries and product lines for new RF products especially SDRs and multi-standard radios. Consequently, it is not ready to meet the requirements of emerging wireless and mobile networks and represents a substantial bottleneck that must be overcome. To do so, it is necessary to elaborate a new RF design scheme that is able to tackle these issues. For this purpose, the research activities being conducted, target the elaboration of such a scheme supporting true adaptability,

flexibility and automation. This scheme also avoids specifications and communications errors from one design stage to another. Additionally, automation is an important goal and will be addressed in the proposed scheme, which we present in the next section as part of a complete framework.

4. HARDWARE ABTRACTION-BASED DESIGN FRAMEWORK FOR RF/ANALOG DESIGN

As stated above, the field of RF design needs a reliable approach which brings some degrees of freedom to the designer and provides certain levels of abstraction thus allowing the high-level design of RF devices and components and masking, at least partially, the physical details of implementation. At the following, we first present the proposed design scheme. Then, we show how the concepts of abstraction we adopted evolved and their impact on the design of RF front-ends.

4.1. The Proposed Design Scheme

The basic idea, initially proposed in [19], consists of considering that any RF system (or subsystem) is a black-box that is defined by inputs, outputs, and configuration parameters which act on a transfer function describing the system's behaviour. This consideration allows a high-level functional description of any RF device and brings a considerable abstraction of technology details related to that system. Starting from this point, we proposed an improved design scheme for RF components which is conceived in a way that separates the functional view of the system from the technology details of the underlying platform. It begins with a hardware-abstracted function description, which captures the desired functionality, and flows to synthesis and realisation through several sequential steps as illustrated in Figure 2. At the heart of this design scheme lays a new data structure, called Q-matrix, which is a generic multidimensional matrix that captures all of the RF device's electrical attributes through a minimum set of parameters. This structure makes the communication between the different stages easily feasible and enables the re-use of existing CAD tools through custom format conversions.

In a preliminary work, this proposed design approach was further elaborated through a case study [20]. We considered a typical RF transceiver and started by capturing its functional description using the Unified Modeling Language (UML). This step was then repeated using the Systems Modeling Languages (SysML) instead.

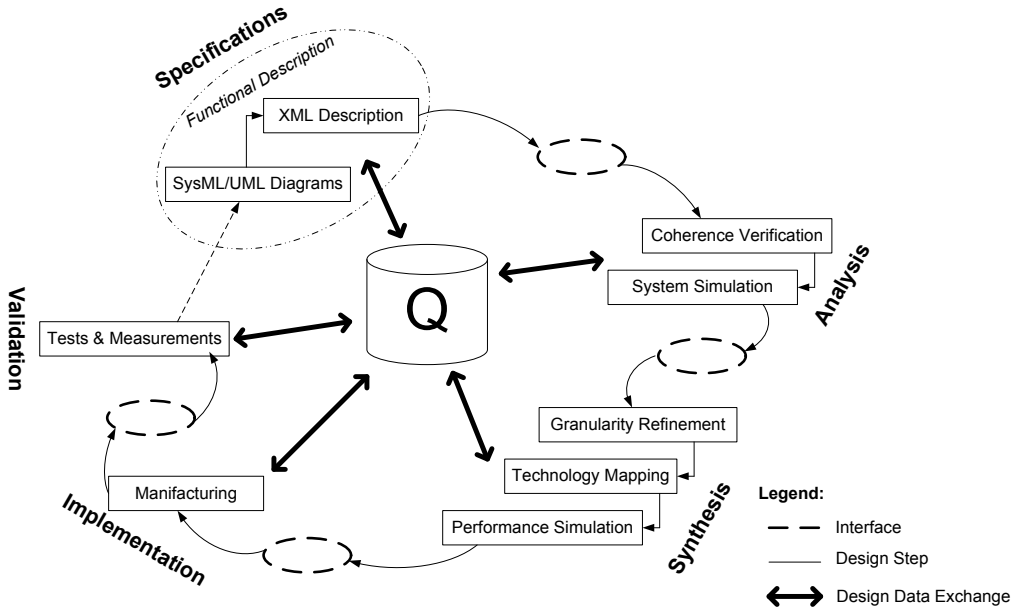

Figure 2. Improved RF design scheme [19]

4.2. Evolution of the Abstraction Concepts inside the Design Scheme

Modeling practice is a suitable solution to build functional descriptions independent from technology platform for complex RF front-ends. However, the question of how a complex RF system will be synthesized from its SysML model according to the initial specifications was an outstanding question to answer. In fact, we were inspired from the software engineering domain where intensive modeling activities are elaborated for complex software designs. The Object Management Group (OMG) proposed the Model-Driven Engineering (MDE) which provides a standardized approach to bolster complex software systems'

design using intensive modeling activities [21]. In practice, the MDA defines three main levels of abstraction (or subsequent models) that are reused by MDE (see Figure 3). As [21] states, the first level of modeling is the Platform-Independent Model (PIM). It is a "*model with a high level of abstraction that is independent of any implementation technology"* [21]. The PIM describes the system (or a part of it) from functional viewpoint without defining how this function is practically realized. The second level defined by this approach is the Platform-Specific Model (PSM). This model is tailored to specify the target system considering the specific details of a given platform technology. The PSM is generated using a particular transformation from the PIM. For each platform technology, a PSM is derived from the corresponding PIM using a particular transformation. The third and final level is the implementation (e.g. source code for software systems) which is also derived from the PSM using a predefined transformation. It is worthwhile to note that, according to [21], a transformation is the process of "*generation of a target model from a source model according to a given transformation definition"* (or rules).

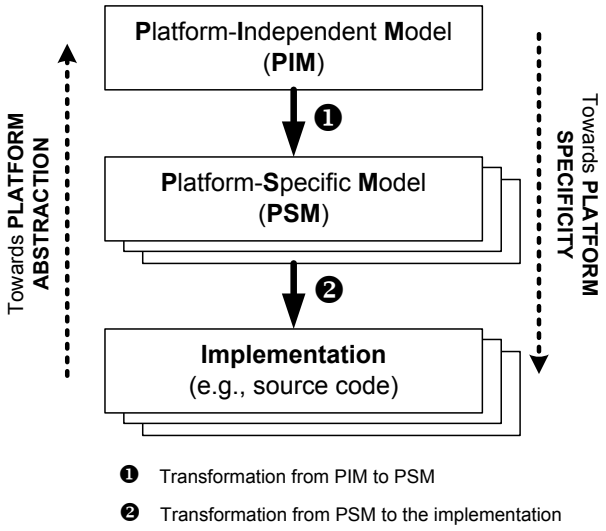

Figure 3. Model-Driven Engineering abstraction levels

For the proposed scheme to be streamlined along key MDE concepts, the first task is to delimit its various stages in accordance with MDE's three level approach, as shown in Figure 3. To this end we continue to consider the Q-matrix in a central position accessible at various steps be they in the PIM, the PSM or Implementation phases. The resulting mapping of our design scheme to the MDE's PIM/PSM/Implementation framework is captured in Figure 4. Under this scheme, the PIM covers the functional description of the system, the coherence verification and possibly system level simulation. In this domain, the system is presented as a level that is totally independent from any technology details or platform. At this level, the abstraction is very high in a way that even an unrealistic system may be functionally described but rejected through coherence verification. Next, the PSM domain may include system simulation and covers the steps of the synthesis process, which is composed of three sub-steps, namely, granularity refinement, technology mapping and performance simulation. In this domain, the system model is enriched with technology details and the abstraction level is lowered in order to take in consideration the physical constraints. At this level, technology limitations, if any, that may prevent the realization of the stated specifications are generally discovered and feedback to the previous stages can be given so the design process may be restarted or re-iterated. On the other hand if no technology limitations are met, then the design will be feasible and can be moved on to the Implementation domain which encompasses the manufacturing and testing steps.

Having established the proper mapping between the MDE framework and our RF design cycle, the next step consists of defining the required transformations to transit between domains. To be streamlined with the MDE approach, these transformations must allow the translation of the PIM (e.g., SysML model offering a relevant functional description) of an RF device to a PSM (e.g., mathematical or technical model considering a target technology). This PSM should be transformable to an implementation realizing the RF functionality.

This said, the transformation from PIM to PSM is dependent on the nature of the system. However, the PIM describes a set of parameters and attributes that are transformable to PSM elements. For example, the transformation shall take into account the power level and the bandwidth in which the device is operating. If the system to develop is a power amplifier, then the transformation may analyze and then generate mathematically the amplifier model that can operate at those conditions bearing in mind some considerations such as stability, noise, linearity, and so on.

5. CASE STUDY: DESIGNING A RF FILTER USING THE PROPOSED APPROACH

To illustrate the principles announced in the previous section, we present in this one, how an RF filter can be designed using the proposed approach. The example we present here, is intentionally simple and realistic in order to ease the illustration. At the beginning, one should note that an RF filter is a passive device widely used in communications in order to select a given bandwidth useful for a specific application. Technically speaking, an RF filter is a two-port symmetrical network. The design of such devices has been the subject of research activities for decades. A large amount of literature exists about the art of filter design. For in-depth readings, the reader may refer to [22], [23], [24] and [25].

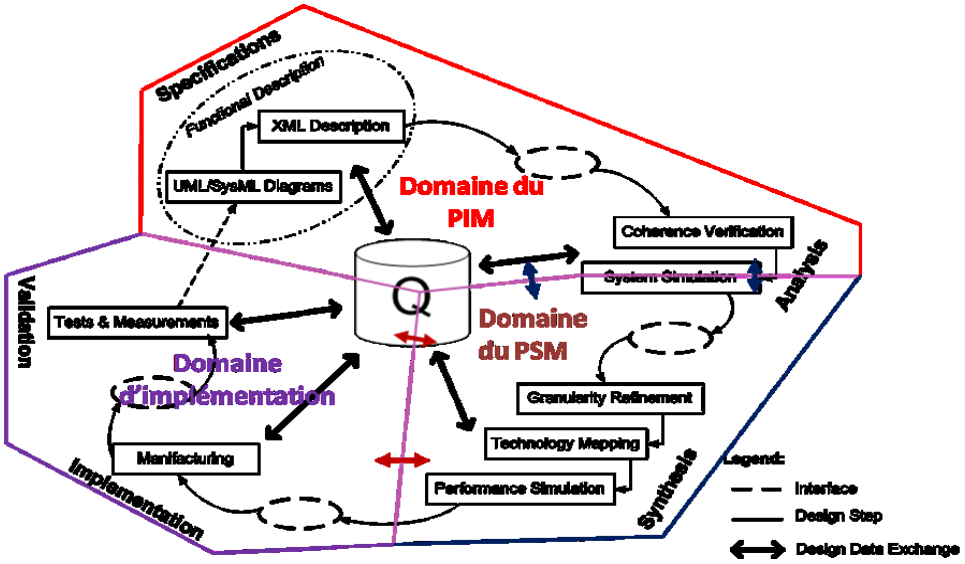

Figure 4. The design scheme with the different MDE domains

For the present case study, we have chosen to realize a lowpass RF filter having the following specifications (see Table 1):

Table 1. Specifications of a typical lowpass RF filter

Items	Specifications	Unit
Passband Edge Frequency (f_0)	0.9	GHz
Cutoff Frequency (f_c)	0.5	GHz
Insertion Loss in passband	≤ 6.0	dB
Ripple in passband	≤ 0.2	dB
Attenuation	≥ 53 @ f_0	dB
Input / Output Impedance	50	Ω

The steps of applying the MDE approach to RF filter design are the following:

5.1. PIM Domain Steps

The RF filter specifications can be captured using a SysML model [20]. One can capture both the properties and the requirements of the filter using the diagrams that SysML offers such as the block definition diagram (bdd) and requirements diagrams. The internal structure of the filter and its behavior (frequency response form) can also be captured by SysML if they are specified. The bdd of the filter captures the filter parameters and the constraints to which these parameters should obey. Figure 5 presents a typical bdd of an RF filter. The property "Type" stands for the filter type (e.g. lowpass, bandpass, highpass, etc.). One should notice that this bdd is a part of an entire SysML model modeling a passive RF filter. It contains the definition of used units, the bdd, the requirements, etc. Due to a matter of space, we present here only the bdd.

The frequency response of an RF filter can be mathematically modeled by a function. This mathematical model is totally independent from the technology. Many known models are described in literature such as Butterworth (also called maximally-flat), Chebyshev, Bessel, etc. [26]. Depending on the values of the RF filter parameters already specified by the designer, the PIM may correspond or not to a particular mathematical model of filters that characterizes its general frequency response. For example, if the specified ripple is equal to zero then the corresponding mathematical model which can present a frequency response meeting this constraint is the maximally flat one. If the designer implies a sharp frequency response at the edge of the passband, Chebyshev model may be the best to meet this requirement. Hence, different PIMs can be defined to an RF filter.

Looking at the specified parameters of the filter, one can notice that the ripple is greater than zero and the bandpass attenuation is slightly large. These observations suggest that the Chebyshev mathematical model is well suited to meet the filter's specifications.

5.1.1. Coherence Verification of the filter Specifications

The main goal of coherence verification is to carry out a preliminary check to verify the coherence of the values specified by the designer. This check is carried out according to a set of rules specified in the SysML model of the filter (see constraints subsection in Figure 5). Such rules take their legitimacy from both the filter theory and also the available implementation technology. For example, it is not

possible to obtain a stopband attenuation greater than 100 dB at a normalized frequency of 1.05 with conventional design methods. Such a choice judged as "unfeasible" is then rejected.

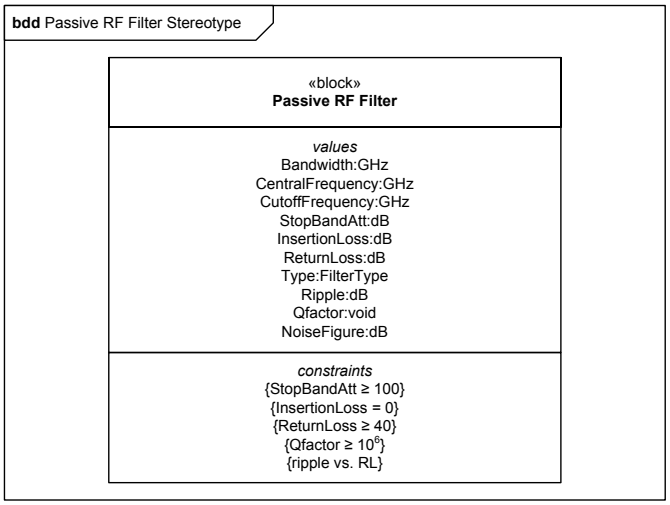

Figure 5. The block definition diagram modeling the RF filter characteristics

The equations (E.1) present a set of possible rules already defined in the bdd of Figure 5. Obviously, the specifications given in Table 1, successfully pass the test.

$$\begin{cases} StopBandAtt \geq 100 \text{ dB} \\ InsertionLoss = 0 \text{ dB} \\ \text{Re}turnLoss \geq 40 \text{ dB} \\ Qfactor \geq 10^6 \\ ripple = -10\log\left(1-10^{\frac{returnLoss}{10}}\right) \end{cases} \quad \text{(E.1)}$$

5.1.2. Q-matrix Creation

Another objective of the coherence verification is to prepare the creation and the use of the Q-matrix. This data structure is based on a mathematical formalism that does not accept any incoherencies [19]. Then to allow the fill of the Q-matrix with the electrical parameters initially deduced from the specifications and use them afterwards in the different stages of design, particularly within the transformation from the PIM to PSM, errors and incoherencies are not allowed and quickly revealed using this process. Theoretically, if the set of rules used to perform the coherence verification is complete, inconsistent specifications cannot be accepted from the beginning.

5.2. Transformation from PIM to PSM

Several methods exist in literature about how a filter can be designed. Two main methods are predominant: (i) image parameter and (ii) insertion-loss methods [22], [26]. The transformation from the filter PIM to PSM that we propose here is inspired from the second method which is frequently used. This transformation is composed of the following steps:

4.2.1. Order Calculation and Prototype Generation

Traditionally, the filtering functionality is represented by a network of lumped components (resistors, inductors and capacitors) [24]. The order of the filter determines the number of lumped elements that will constitute the network performing the required filtering response over the required bandwidth.

The order of a filter Chebyshev is calculated according to (E.2) [23, p.357]:

$$N = \frac{\cosh^{-1}\left(\sqrt{\left(10^{\frac{insertionLoss}{10}}-1\right)\Big/\left(10^{\frac{ripple}{10}}-1\right)}\right)}{\cosh^{-1}\left(\frac{f_0}{f_c}\right)} \quad \text{(E.2)}$$

Carrying out the calculations, we get $N = 6.9766$. Due to the fact that N must be an integer, we choose $N = 7$. Hence, the corresponding lowpass prototype is as shown in Figure 6.

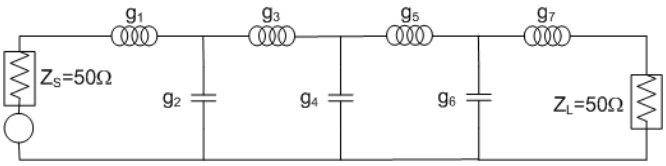

Figure 6. Lumped components prototype of the target filter

After frequency normalization, the values of g_i where $i = 1..7,$ are computed following the equations of Chebyshev lowpass filter synthesis [23, p.359]. After frequency and impedance scaling, these values correspond to the values of the inductors L_i and capacitors C_j which compose the filter network. Finally, we obtain the results in Table 2.

After generating the filter prototype, we obtained the values of its different lumped components. To check the validity of these values and figure out the frequency response of the resulting network, we carried out the simulation of this network using Advanced Design System (ADS) [27]. The frequency response of the designed filter is as shown in Figure 7(a) (for the insertion loss) and 7(b) (for the return loss).

Table 2. Values of the filter's lumped components

Lumped Element	Value (Unit)
L_1 (g_1)	21.84 nH
C_2 (g_2)	8.772 pF
L_3 (g_3)	34.22 nH
C_4 (g_4)	9.552 pF
L_5 (g_5)	34.22 nH
C_6 (g_6)	8.772 pF
L_7 (g_7)	21.84 nH

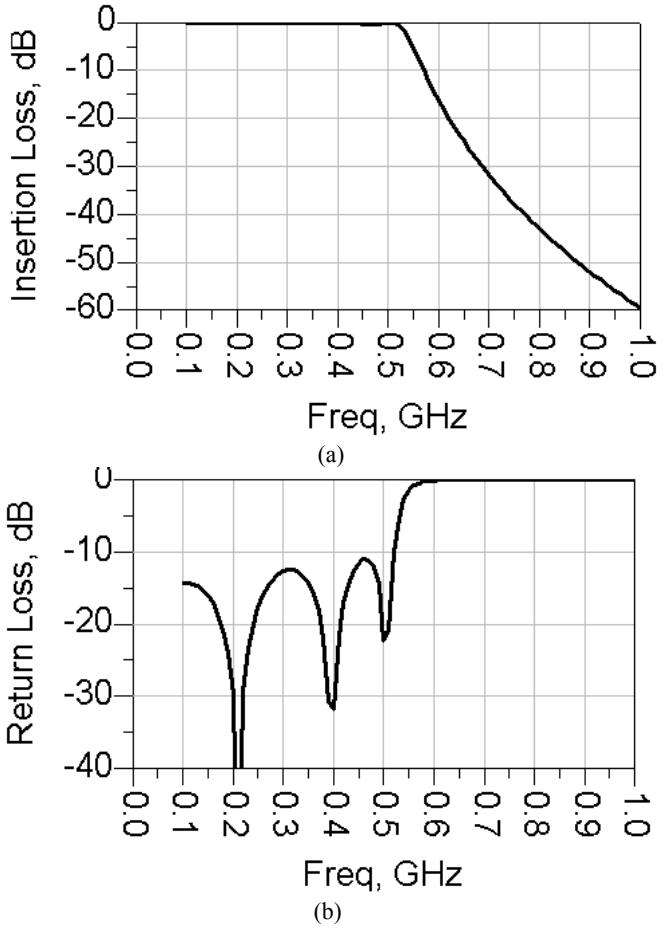

Figure 7. Frequency response of the lowpass filter: (a) insertion loss, (b) return loss

5.2.2. Technology Considerations

After generating the prototype of the filter and the calculation of the lumped components values, the details of the technology must be taken into consideration. For this case study, we consider using lumped components, which typically have low parasitics at frequencies below 500 MHz, and a standard printed circuit board (PCB) substrate (Rogers RO3010 Substrate) on which the filter will be fabricated. With the aid of the ADS package and using its built-in library of L and C lumped components provided by Dale Technologies, we carry out final filter simulation and produce a circuit layout for the completed design. This resulting layout, shown in Figure 8, constitutes the output of our PIM to PSM transformation. It is typically given in a standard format (*i.e.*, Gerber Format), and can then be used by various fabrication processes (in the Implementation domain) where it generally goes through a transformation that makes it compatible with the particular fabrication equipment/process being used.

The lowpass filter characteristics and the lumped element realization were chosen for illustration purposes and for simplicity reasons. However, bandpass and highpass filters can also be designed by using well established transformation identities known as Kuroda's identities [22, p.406]. Similarly, distributed elements may be used instead of lumped component by relying on simple transformations that use high characteristic impedance lines, Z_o^h, to produce the equivalent inductances (L) and low characteristic impedance lines, Z_o^l, to produce the equivalent capacitances (C). These transformations are given by the equations (E.3) and (E.4) [23, p.382]:

$$L = \frac{Z_o^h \beta d}{\omega} \qquad \text{(E.3)}$$

$$C = \frac{\beta d}{Z_o^l \omega} \qquad \text{(E.4)}$$

where β is the phase constant, d is the transmission line physical length and $\omega = 2\pi f$ is the angular frequency.

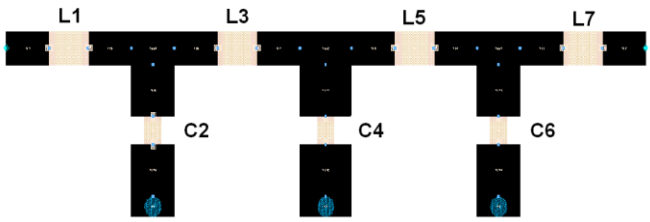

Figure 8. Final layout of the lowpass filter

6. CONCLUSION

In both academia and industry, the SDR paradigm is seen as the technology of the future. The hopes that one day end-to-end reconfigurability, effective interoperability and high flexibility will be achieved by the implementation of the ultimate software radio are real. Current efforts are deployed to define open standards allowing the development of fully portable software components across platforms. Various techniques of both hardware and software abstraction are used to enable cost-effective waveforms porting and

interoperability. However in the RF front-end side, little research effort targeted to adapting the design methodologies of RF front-ends to the pace of advances occurring in the SDRs' baseband design. RF design methodologies lack flexibility and are very technology-dependent. To find out a suitable response to this issue, we proposed a new design methodology based on hardware abstraction which aims to uncouple the RF device's functionality from technology.

The proposed design scheme includes a functional description stage in which the structure, requirements and constraints of complex RF systems are captured. We used modeling languages such as SysML to elaborate the platform-independent model of the system under design. This model is synthesized in the next stage of design using a model-to-model transformation (i.e. PIM to PSM). This operation results in a platform-specific model which is in practice a circuit implemented in a given technology. The design data are centralized in the Q-matrix all over the design scheme enabling thus cooperative design.

The abstraction and automation concepts we used and are inspired from the MDE approach offer a novel view for system design where functional and operational levels are separated. The technology input becomes an input to the design cycle which eases the insertion of various technologies and even the benchmark of their resulting performance. While our design scheme still needs refinement and experimentation, the case study we presented in this paper provides an overview of how a RF device such as a filter can be designed using the proposed design methodology.

7. REFERENCES

[1] M. Löhning, G. Fettweis, "The Effects of Aperture Jitter and Clock Jitter in Wideband ADCs," *International Workshop on ADC Modeling and Testing*, 2003.

[2] M.J. Leferman, *Rapid Prototyping Interface for Software-Defined Radio Experimentation*, Worcester Polytechnic Institute, Massachusetts, 2010.

[3] V. Rodriguez, C. Moy, and J. Palicot, "An Optimal Architecture for a Multi-standard Reconfigurable Radio," *IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1-5, 2006.

[4] H.W.W. Tuttlebee, "Software-Defined Radio: Facets of Developing Technology," *IEEE Personal Communications*, Vol. 06, pp. 38-44, 2002.

[5] K.F. Jondral, "Software-Defined Radio – Basics and Evolution to Cognitive Radio," *EURASIP Journal on Wireless Communications and Networking*, Vol. 2005, pp. 275-283, 2005.

[6] T. Rittenbach, "High Capacity Communication Capability (H3C) GRA: Alternatives for Third Party A2G Waveform Porting," *IEEE Military Communications Conference*, pp. 1-5, 2007.

[7] J. DeGroat, G. Reehal, and S. Nagarjuna, "Synthesizing Digital Modules for Software-Defined Radio," *IEEE National Aerospace and Electronics Conference*, pp. 358-363, 2009.

[8] V. Giddings, T. Kacpura, and V. Kovarik, "Methods and Approaches for Abstraction of Hardware Dependencies in Software-Defined Radio," *SDR Forum Technical Conference*, pp. 1-6, 2007.

[9] J. Mitola, "The Software Radio Architecture," *IEEE Communications Magazine*, Vol. 33, No. 5, pp. 26-38, 1995.

[10] L. Sabel, "Software-Defined Radio – The solution for multi-standard multimedia in the mobile environment," *EBU Technical Review*, No. 309, pp. 1-8, 2007.

[11] V.J. Kovariak, "Integrated Terminal System Development: The HC3 Reference Architecture," *IEEE Military Communications Conference*, 2007.

[12] P. Bhaskar, Z. Jianxin, and T. Rittenbach, "Software Development of Satcom Terminals," *IEEE Military Communications Conference*, 2008.

[13] M. Kling, et al., "An Implementation of the Government Reference Architecture Waveform Developer and System Integrator Roles," *IEEE Military Communications Conference*, pp. 1-5, 2008.

[14] L. Pucker, "Component-based Development of Radio Systems and Subsystems: Are we there yet? [Trends in DSP]," *IEEE Communications Magazine*, Vol. 45, No. 6, pp. 44-46, 2007.

[15] S. Yoo, and A.A. Jerreya, "Introduction to hardware abstraction layers for SoC," Design, Automation and Test Conference, pp. 336-337, 2003.

[16] S.M. Lee, D.G. Kim, and D.R. Shin, "General Purpose Hardware Abstraction Layer for Multiple Virtual Machines in Mobile Devices," *11th International Conference on Advanced Communication Technology*, Vol. 01, pp. 362-364, 2009.

[17] J.M. Rabaey, *Digital Integrated Circuit – A Design Perspective*, Prentice Hall, Second Edition, pp. 8 – 12, 2002.

[18] R. Frevert, et al., *Modeling and Simulation of RF System Design*, Springer, 2005.

[19] S. Lafi, A.B. Kouki, J. Belzile, and A. Ghazel, "Towards a Coherent Framework for Automated RF Front-Ends Design Using Hardware Abstraction," *IEEE International Summit of Telecommunications Conference*, pp. 1-5, 2007.

[20] S. Lafi, A.B. Kouki, and J. Belzile, "Modeling Radiofrequency Front-Ends Using SysML: a Case Study of a UMTS Transceiver," *1st International Workshop on Model-Based Architecting and Construction of Embedded Systems*, pp. 115-128, 2008.

[21] A. Kleppe, J. Warmer and W. Bast, *MDA Explained – The Model-Driven Architecture: Practice and Promise*, Addison Wesley, 2003.

[22] D.M. Pozar, *Microwave Engineering*, John Wiley & Sons Inc., Third Edition, 2005.

[23] D.K. Misra, *Radio-Frequency and Microwave Communication Circuits: Analysis and Design*, Wiley-Interscience, Second Edition, 2004.

[24] I. Hunter, et al., "Microwave Filter Design from System's Perspective," *IEEE Microwave Magazine*, Vol. 08, pp. 71-77, 2007.

[25] I. Hunter, et al., "Microwave Filters – Applications and Technology," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 50, No. 03, pp. 794-805, 2002.

[26] J.S. Hong, and M.J. Lancaster, *Microstrip Filters for RF/Microwave Application*, John Wiley & Sons, 2001.

[27] Agilent Technologies, *Advanced Design System Overview*, http://www.home.agilent.com/agilent/product.jspx?nid=-34346.0.00&cc=US&lc=eng, 2011.