HAL
open science

UNIFIED MODELLING TECHNIQUE USING VHDL-AMS AND SOFTWARE COMPONENTS

Abir Rezgui, Laurent Gerbaud, Benoît Delinchant

To cite this version:

Abir Rezgui, Laurent Gerbaud, Benoît Delinchant. UNIFIED MODELLING TECHNIQUE USING VHDL-AMS AND SOFTWARE COMPONENTS. Electrimacs 2011, Jun 2011, Cergy-Pontoise, France. hal-00600311

HAL Id: hal-00600311

https://hal.science/hal-00600311v1

Submitted on 24 Jan 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Unified modeling technique using VHDL-AMS and software components

A. Rezgui, L. Gerbaud, B. Delinchant

Grenoble Electrical Engineering Lab (G2ELAB),
UMR CNRS 5269, Grenoble-University
ENSE3, rue des Mathematiques, BP46, 38402 Grenoble, France.
Tel: +33(0)476826208 ; Fax: +33(0)476826300
Abir.REZGUI@g2elab.grenoble-inp.fr

Abstract

The paper deals with the dynamic modeling of mechatronic devices, which usually need detailed modeling to be described and to take into account the physical properties of the system. VHDL-AMS[1] , which is a powerful unified modeling language for mixed system, allows to describe a large range of physical systems, for their dynamic simulation. It allows to describe models of physical components and then to connect them to obtain the model of a system.. However, this language cannot support the description of some physical phenomena, such local ones, defined by numerical methods (e.g.: finite element method, special numerical integrals). When an aspect of a model cannot be described in VHDL-AMS, the paper proposes to use software components. So, the aim of the paper is to propose a generic way to extend the computation capability of VHDL-AMS, by coupling the models described in VHDL-AMS with external ones specified as software components (where VHDL-AMS fails). The approach has been applied on several applications, among them the time simulation of an electrical plunger

Keywords: VHDL-AMS, software components, coupling models, complex multi physics systems

1. Introduction

Mechatronic systems can be defined as complex and heterogeneous systems. Their models deal with different physical aspects: mechanical, electrical, thermal, and magnetic, etc. It is therefore essential to achieve a successful simulation of theses systems; which is a delicate task facing several problems related to models and simulation tools: (1) wide choice of models of different natures and described in different formalisms; and (2) multitude of time simulation tools,

[1] Very High Speed Integrated Circuit Hardware Description Language - Analog and Mixed-Signal

specific for its domain, which don't have a direct compatibility between them. The modelling level of the components of such systems depends on the aim of the study carried out for them. For example, a electrical generator may be represented by an ideal source for its study in a large power grid simulation, a park model for it study in a wind generator simulation, a finite element model in the analyse of its magnetic behaviour. So, the choice of the abstraction level is a compromise between computations speed, accuracy, model description easiness, or its availability in a model library. However, there is no tool allowing a complete simulation of mechatronic system and its components and be able to meet all the user's needs. To address these problems, specially for modeling electromagnetic devices as a kind of mechatronic system, it is important to adopt interoperability solutions ensuring compatibility with other models and with simulation tools. The IEEE Glossary defines interoperability as: "'the ability of two or more systems or components to exchange information and to use this information that was exchanged". Various approaches can be used:

- The first approach focuses on the possibility to link applications at runtime in order to co-operatively exchange information. This co-simulation strategy allows a more flexibility modeling and simulation than others because the models are simulated separately on their own dedicated software.

- - The second approach is achieved on the level of the models, which describes mechanical, electrical, magnetic and other physical processes. It consists on extracting models developed in a specialized tool for externally reuse in industrial mechatronic simulators (e.g. Portunus, Simplorer, Smash, etc) in order to benefit from the expertise behind. This is done by: (1) describing models with a standardized multi-physics modeling languages like VHDL-AMS [1], Modelica [www.modelica.org], system-C [www.systemc.org] and Verilog-AMS [www.designers-guide.org]; or (2) exchanging and reuse of existing components models at the source code level or through a dynamic link library.

The paper focuses on the system modeling level for dynamic simulation, applied to electromagnetic devices and for interoperability needs in order to develop a unified architecture for the multilevel modeling of theses devices, based on a standard for model description. The most difficult task of the proposed approach is to simulate accurately the magnetic component in VHDL-AMS. However, what to do for modelling aspects that are not explicitly supported by VHDL-AMS (ex: partial differential equations) or difficult to describe in such a language (e.g. due to numerous symbolic treatment, like for the force computation in the application presented in the last section)? To avoid the limits of the language, the paper proposes for that an approach based on the transfer of models from dedicated magnetic tools of these components like RelucTool software [2], developed at G2ELAB, for designing reluctant networks [3]; to a general mechatronics simulation environment like Smash, thus allowing the simulation of the component coupled to its electro-mechanical environment. RelucTool

software generates automatically a black box model as a software component from a graphical description of the reluctant network. This standardized software component has been defined in our laboratory and they are named ICAr (for: Interface for Component Architecture). So, the proposed approach have to export this specific software components into a VHDL-AMS models, ensuring then the interoperability between models and offer to designers the possibility to use VHDL-AMS as a unified description language for many modeling aspects.

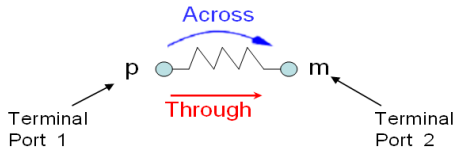
In this way, section 2 deals with VHDL-AMS aspects and the software component paradigms, focusing on the ICAr standard developed in our lab. Section 3 presents the need of the proposed approach (of exchanging models) for designing an electromagnetic actuator. Then, section 4 discusses the way to carry out the coupling between VHDL-AMS and ICAr software. In this section, a generic implementation of this coupling will be presented discussing the data exchange between the two formalisms. Finally, in section 5, this approach is applied on the linear actuator. It shows how to mix such formalisms to perform the dynamic simulation of the actuator.

2. Modeling techniques and tools

2.1. VHDL-AMS Language

The VHDL-AMS, an IEEE standard 1076.1, is an extension of VHDL hardware description language, which is used for the description and simulation of event- driven systems [1]. VHDL-AMS propose to support the description and the simulation of analog and mixed signal circuits and systems. It allows to define and to deal with the modeling of the behavior of multi-physical and multi-technological systems [4]. Its purpose is to provide formalism for the hierarchical description and simulation of physical systems. Designers can create and use models that integrate high-level behavioral descriptions, as well as structural and physical descriptions. A VHDL-AMS model consists of an entity and one or more architectures [5]. The entity specifies the extern view of the model. It includes the description of its physical ports and the definition of its generic parameters. The architecture contains the implementation of the model. It may be coded using a structural or behavioral description, or both. The structural description specify the connectivity between the physical components; which can be carried out with the Architectural Description Language (ADL[6]),and the behavioral description is made by concurrent statements to describe event-driven behaviors and simultaneous statements for continuous ones. Fig.1 is a VHDL-AMS description of an electrical resistor.

This language specifies the possibility to describe the behavior of a complex continuous system by formulating its corresponding differential algebraic equations (DAE). Thanks to its standard well-defined in terms of syntax and semantics, it ensures portable descriptions among different industrial simulators (e.g. Portunus, Simplorer, Smash, etc.). Due to their complexity, partial differential equations (PDE) were left out in VHDL-AMS [7]. This limits the accuracy of system block modeling which includes local physical effects. Because of the



Extern View of Resistor	Intern View of Resistor

entity resistor is
generic(R:real:=1.0E2);
port(terminal m,p: electrical);
end resistor;
	architecture behav of resistor is
	begin
	quantity V across I through m to p;
	begin behav
	V=I*R ← Simultaneous statement
	end behav ;

Figure 1: VHDL-AMS model of a resistor

importance of PDE in electromagnetism, different approaches filling this lack in the design flow using VHDL-AMS are proposed. A reduction of FEM models is presented in [8], which is an export of reduced order modeling (ROM) to VHDL-AMS with an interface matching the system structure. In paper [7], two modelling approaches were used: the first consist in using a spatial discretization and the second consists in using an equivalent circuit approach. Then, an extension of VHDL-AMS has been also proposed in order to take into account this aspects [7], but it is not yet integrated on the standardized language.

2.2. Software components

The "software component" modeling approach has been extensively studied to design models on several applications and domains, as electromagnetism, electronics, microsystems and so on [2] [9] [10] [11] [12]. A software component standard is defined as an autonomous deployment entity, encapsulating a software code and described by its interfaces. A specific pattern has been defined in our laboratory for the standardization of such software components [9]. They are named ICAr (for: Interface for Component Architecture). The ICAr component is an executable Java code defined by its input/output variables and the services that it can bring. It is characterized by an evolutionary architecture and can be seen as a multi-services component. Services are available via different facets that are accessible via the interface "component" (Fig. 2). Each facet may represent behavioral model, sensitivity computation, time simulation, model accuracy. Facets mayalso represent a part of the system to be simulated (electromagnetical, mechanical, thermal, economical).

The "component" interface presents the way to instantiate the component, defines its inputs and outputs, and how to interact with all its features, as shown in Fig. 3

This paradigm of software component in general and specifically the ICar component assures sharing, exchanging, capitalizing and reusing models. It can

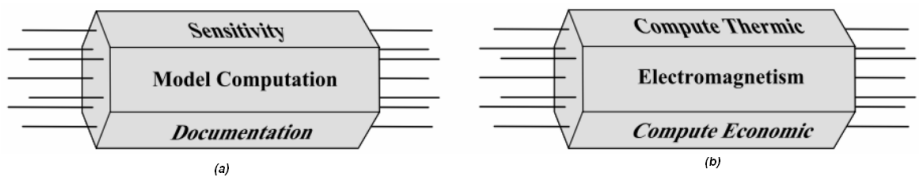


Figure 2: Multi-facets component: (a)multi information, (b) multi-physics.

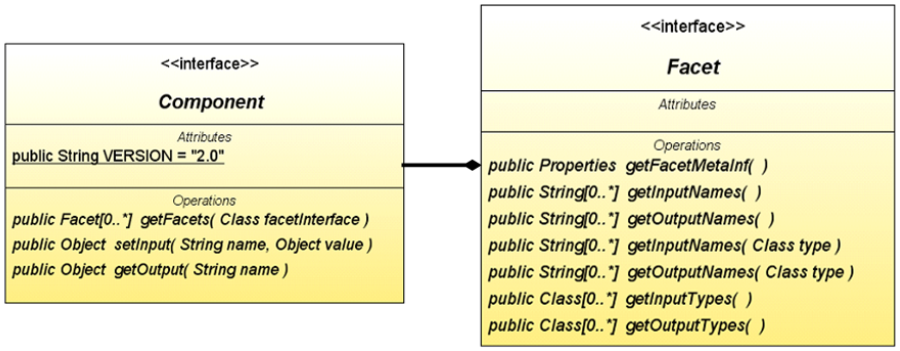


Figure 3: ICAr interfaces: component and facet (Java implementation)

also support complex models and their solvers whatever their physical nature is.

3. The need of an exchange model approach on electromagnetic design

Fig. 4 show the studied device, which is a dynamic moving core actuator with a coil bounded to its central leg [13]. The dynamic motion of a core is driven by a magnetic system, and it is dealing with electric, magnetic and mechanical modeling. It is very challenging to design such a system accurately because its operating is based on several physics and complexity levels. Indeed, it is a multi-domain device including the magnetic circuit, interfacing the electrical domain and the mechanical loads. The electrical driver circuit controls the current in the coil, and the mechanical subsystem computes the influence of damping sources on the displacement of the moving part damping behaviour. Also, many physical effects in this system can be mathematically formulated by a coupling between PDE and DAE. In this way, how to design this system by invoking the multi physical modeling using VHDL-AMS as unified language modeling?

The objective of this work is to be able to design and simulate this system with VHDL AMS language to solve interoperability problems. In this electromagnetic actuator:

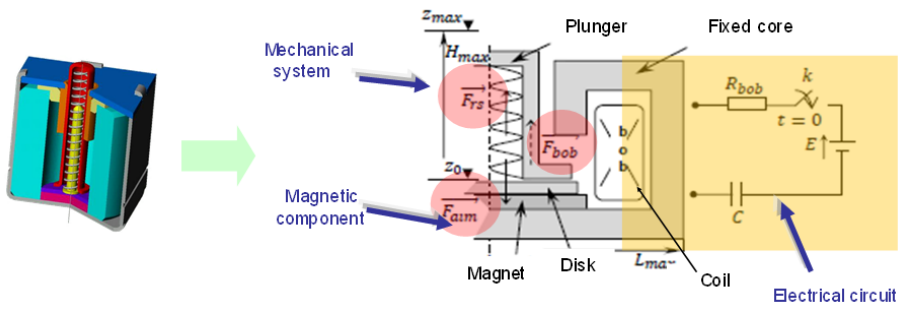


Figure 4: Simplified moving core actuator description

- the state system modeling of the electrical circuit feeding the coil was defined by two differential equations; the inductance value is provided by the magnetic model.
- the state variable of the mechanical part of this system are the core position z, and its velocity v; the state system is defined by two differential equations depending on the value of applied residual forces.
- the magnetic part of the actuator which is represented by reluctant networks (RNM[3]).

The complete device model can be carried out in several ways, among them:

- a VHDL-AMS modelling of the complete device using the model of paper [13]; however, the magnetic force calculation is very approximated. This difficulty is more presented in paper [14] proposing a Modelica description of reluctance networks.
- an accurate model by finite element for the magnetic part and a VHDL-AMS modelling for the electrical and mechanical parts; however, the numerical method is computing time consuming [15]
- a reluctance network model for the magnetic part with the magnetic force obtained from the derivative of the co-energy according to the movement variable (z in figure 4) [16], and a VHDL-AMS modelling for the electrical and mechanical parts.

The third is more suitable for system level modeling of the dynamic actuator. However, how to render easy the modelling of magnetic part, in particular as said previously VHDL-AMS language don't support non temporal derivative? The solution proposed is then to apply the approach based on exchanging model from a dedicated tool for designing magnetic circuit and coupling the VHDL-AMS model to the electro-mechanical one on Smash simulator.

ReluctTool is proposed as a dedicated tool for the fast simulation and presizing of electromechanical actuators [12]. In which, the reluctance network

is described graphically. The magnetic force expression is obtained automatically from the derivative of the co-energy in the reluctances according to the movement variable [16]. The model is automatically generated in a software component as an ICAr component.

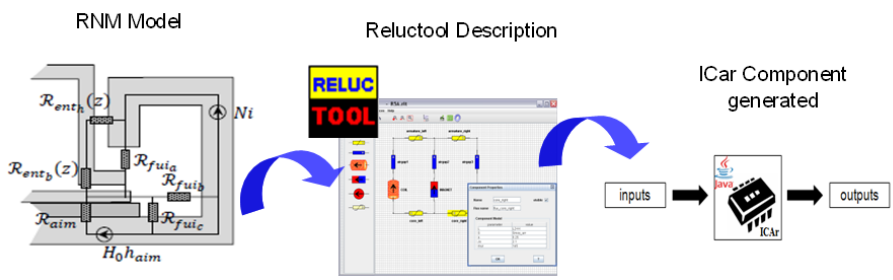


Figure 5: The reluctance representation of the magnetic circuit

As a result, for the approach, the behavior model of the magnetic part of the actuator is a RNM model, implemented in RelucTool tool (Fig. 5). It generates, then, an ICAr component encapsulating this model. It is the software component, that it be included into a system simulation of the whole dynamic actuator on Smash simulator using VHDL-AMS language. How can the coupling between VHDL-AMS and ICAr software can be achieved?

4. Model interchange using VHDL-AMS language

4.1. Interchange porcess

As it has been indicated, the approach followed for model interchange between the ReluTool software et SMASH tools consists in transferring the models using VHDL-AMS for model description. Fig. 6 shows the overall process that starts by creating the model of the magnetic component using the proprietary languages of ReluTool software. Then this tool exports its models in VHDL-AMS format, using the language subset described previously. This architecture allows the transfer of models from RelucTool to other VHDL-AMS native simulators. This independent entity can be imported in SMASH as a VHDL-AMS model via external functions. For that, the VHDL-AMS export function is added to Reluctool software have to deal with the coupling between VHDL-AMS models and others described in software components.

4.2. Export an ICAr component in VHDL-AMS

The export functionality consists in defining an interface or overlapping layer between the VHDL-AMS models and an external code (written in C or encapsulated software component), for data exchanges. The final model obtained is a VHDL-AMS model with external functions, as one single ENTITY of VHDL-AMS, with its dynamic behaviour represented by ARCHITECTURE. It requires

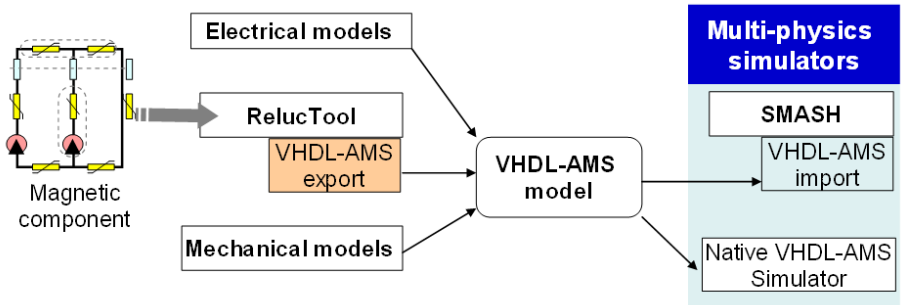


Figure 6: Process and Model interchange using VHDL-AMS

as a first step to encapsulate the whole magnetic model as a black box that behaves externally as any dynamic system. VHDL-AMS offers this opportunity to extend his functionalities by including external functions written in another programming language or and foreign subprograms in its models. That is possible thanks to its pre-defined attribute called "Foreign" [1], which allows the user to transfer additional information to the simulator or other external entities.

The "FOREIGN" attribute is used then to complement architectures, functions or subprograms for VHDL-AMS models. Although this attribute is defined in the VHDL-AMS its use is not specified, so it depends on its implementation in simulation software. A foreign subprogram is a program in other formalism than VHDL-AMS, but call in VHDL-AMS thanks to FOREIGN' attribute, as shown in Fig. 7. This call is defined by a string characterizing the external program as a specific function. This string value depends on the implementation of both the "Foreign" attribute in the time simulator and the external program implementation.

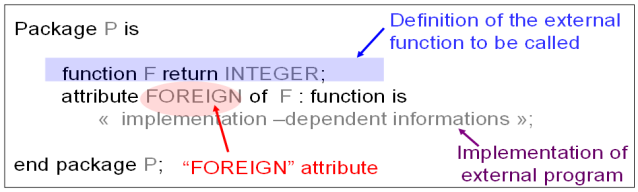


Figure 7: Package defining the 'Foreign' attribute of VHDL-AMS

However, the VHDL-AMS standard does not specify the mechanism for the implementation the FOREIGN attribute. Consequently, the exact interpretation of this attribute depends on its implementation in a particular simulator. At the present time, the only simulator which implements this attribute is Smash from Dolphin integration [www.dolphin.fr]. This implementation requires only the integrating of C functions. So, the export functionality of ReluctTool relies basically on the use of native code (C/C++) to perform this coupling [17].

As a result, to import of the external models on various formats into VHDL-

AMS systems or to make it able to communicate with external environements/solvers, the authors have developped a generic C-communication interface for this coupling. This interface makes able easily (Fig. 8) the VHDL-AMS model to include C/C++ models or dynamic linked libraries (DLL) and allows exchanging data between them.

However, ICAr components are developed in Java language, because it is an independent platform language and so it offers more portability for component. That is why, to be able to load and manipulate these software components from a program or a software tool developed primarily in C/C++, Java data-structures have to be converted into programs compatible with C data-structures. These conversion routines are based on the Java Native Interface (JNI), i.e. the standardized C-interface of the Java Virtual Machine (JVM).

So, to import an ICAr component from a C program, this last should be able to understand the component services via its various facets. For that, the paper proposes an "Interface Adapter" to implement this functionality proposing generic services for introspection of theses components (Fig. 8). A C/C++ dll project was built gathering the different C functions for JVM launching, JNI adaptation routines, ICAr loading, facets and interfaces introspection and methods calling. Once the adaptation between the ICAr component and the C code is made, each C function compiled in a library can be after that called in VHDL-AMS using Foreign attribute defined by C/++ external functions.

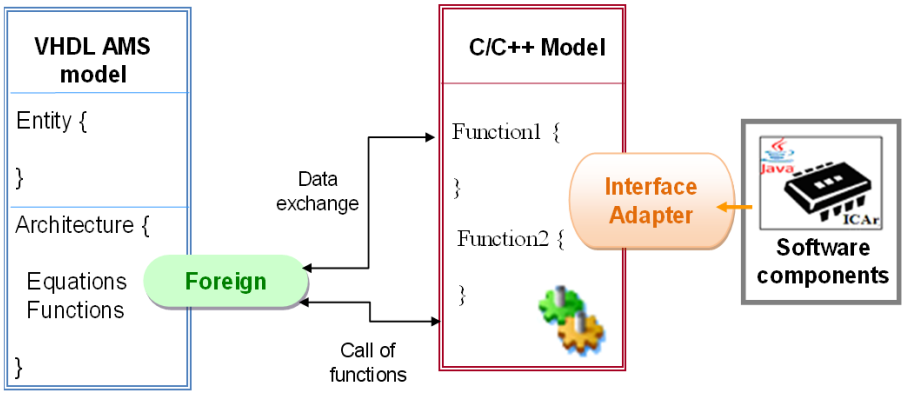


Figure 8: Coupling between the AMS and VHDL model component ICar

Finally, a VHDL-AMS -skeleton, corresponding to the architecture of the ICAr model, is generated automatically by a Java program. This program introspects the component in order to determine its facets, different inputs and outputs and generates the corresponding VHDL-AMS block description. This block defines its inputs/outputs and parameters. It also describes relations between them, in equation section, defining the model. These relations are defined through a well defined sequence of C external functions.

5. Example of coupling Strategy on electromagnetic actuator

n order to validate the methodology of "data coupling" between VHDL-AMS model and ICAr component, in order to validate the export functionality added to Reluctool, an electromechanical actuator is designed in VHDL-AMS language and simulated (Fig.5). As said in section 3, the dynamic motion of a core is driven by a magnetic system, and the model consists of electric, magnetic and mechanical modeling. The dynamic model of the device represents a coupling between the electrical circuit feeding the coil and the mechanical part of the moving part. Only the linear vertical movement of the moving part is possible, which is subject to different damping sources due to translational linear spring.

This electromagnetic device is designed and simulated using the approach previously presented:

- the electrical circuit was described by its two differential equations (ODE) on VHDL-AMS with the inductance value as an external inputs;

$$\begin{cases} \dot{i} = \dfrac{di}{dt} \\ \ddot{i} = -\dfrac{R}{L(z)}.\dfrac{di}{dt} - \dfrac{i}{L(z).C} \end{cases}$$

- the mechanical part was also described by its two differential equations (ODE) on VHDL-AMS to compute position z and velocity v, depending on the value of applied residual forces;

$$\begin{cases} \dot{z} = \begin{cases} 0, \text{si } F < 0 \\ v, \text{si } F \geq 0 \end{cases} \\ \dot{v} = \begin{cases} 0, \text{si } F < 0 \\ \dfrac{F}{m}, \text{si } F \geq 0 \end{cases} \end{cases}$$

- - the magnetic circuit of the actuator is developed on ReluctTool as RNM model (Fig. 7), providing an ICar component exported into VHDL-AMS model. This component compute the magnetic force from the derivative of the co-energy according to the movement variable (z) [16];

The ICAr component of the RNM model was generated with the Reluctool tool as a static component (no time differential expressions). Once the component is generated, the "interface adapter" based on the JNI is implemented. The last program must respect the "FOREIGN" mechanism as implemented on SMASH software, as shown in Fig.8. Then, the VHDL-AMS package is programmed to be able to call the "foreign subprogram" on VHDL AMS model

by specifying the name of the called function, its path and its arguments. This "interface adapter" is an automatic interface which allows to load an ICar component, and makes it able to communicate with its exterior environment.

This component is then inserted into the global dynamic actuator with its environment circuit model described in VHDL-AMS language. The full VHDL-AMS model is shown in Fig. 9.

Fig. 10 shows the exchange protocol between a VHDL-AMS model and an ICAr software component in a time simulation process. The protocol is implemented into SMASH software, considering the actuator as a macro VHDL-AMS model calling external functions which exchange inputs/outputs through a software component. So, when the simulator starts, all the modeling objects are loaded and connected, and initial values are set. During this stage, all foreign shared libraries are loaded and initialization functions of all foreign architectures are called.

A first analysis was performed with the moving core actuator system. The main dynamic behaviour is analysed. Fig. 11 shows simulation results for the evolution of all state variables of the actuator happening during the simulation:

- the position of the moving core (z) and the speed of the moving core (v).
- the residual forces (Ftot) applied to core and the magnetic force (Fmag) generated by magnetic component
- the current generated by the RLC circuit (I) and the variable inductance.

The aim of the paper is the coupling of VHDL-AMS and component software for the dynamic modelling of a electromechanical actuator. So, results are not discussed. It can be say that they are similar to the result given in [13] where the force is more approximated, and where a finite element simulation is used as a reference.

6. Conclusions

The approach proposed in the paper aims to couple models described in VHDL-AMS models and in ICAr software component, allowing the addition of functions which are not in VHDL-AMS standard like numerical algorithm used in electromagnetism. This approach provides also capabilities to import structural models implemented in ICAr components that describe physical properties of the system (e.g. finite element models), in order to be integrated into the global system described in VHDL-AMS. This strategy offers the possibility to deal with complex

This approach is valid for any time simulator which provides mechanisms for the coupling of VHDL-AMS models with any C/C++ interfaces (or other language) thanks to its 'Foreign' attribute, which is not yet implemented in all VHDL-AMS compatible simulators, and is often specific to each simulator. So, some adjustments must be made from a 'Foreign' attribute implementation to another. Since both ICAr components and VHDL-AMS have been defined

for purposes of portability and reusability, models developed under different formalisms must be interconnected to build global system models.It an be also used for any systems can be designed as an ICar software component such magnetic microsystem, and others.

Acknowledgements

The authors thank the French National Research Agency (ANR) for their support throughout the project Mocosymec, Yannick Herve from the INESS for its advices, and Dolphin Integration for their specific development to improve the FOREIGN implementation in Smash.

References

[1] VHDL Analog and Mixed-Signal Extensions, IEEE Standards 1076.1-1999, Mar. 1999.

[2] B. Du Peloux, L. Gerbaud, F. Wurtz, E. Morin, A method and a tool for fast transient simulation of electromechanical devices: application to linear actuators, MOMAG , Vila Vehla, Brazil, 2010.

[3] J. Turowski, Reluctance networks, Computational Magnetics, J. K.Sykulski, Ed. London New York, Chapman and Hall, ch. 4, First Edition, 1995.

[4] E. Christen, K. Bakalar, VHDL-AMSA Hardware Description Language for Analog and Mixed-Signal Applications, IEEE transactions on circuits and systems: Analog and digital signal processing, vol. 46, no. 10, october 1999.

[5] P.R Wilson, J.N Ross, A.D Brown, A. Rushton, Multiple domain behavioral modeling using VHDL-AMS, Circuits and Systems, ISCAS '04, 2004.

[6] N. Medvidovic, R.N. Taylor, Member, A Classification and Comparison Framework for Software Architecture Description Languages, IEEE trans. software engineering, vol. 26, no. 1, January 2000

[7] P.V. Nikitin, C.R. Shi, B.Wan, Modeling partial differential equations in VHDL-AMS, Systems-on-Chip Conference, 2003.

[8] M. Schlegel, F. Bennini, J. Mehner, G. Herrmann, D. Mller, W. Dtzel, Analyzing and Simulation of MEMS in VHDL-AMS Based on Reduced Order FE-Models, IEEE SENSORS JOURNAL, VOL. 5, NO. 5, OCTOBER 2005.

[9] B. Delinchant, D. Duret, L Estrabaut, L. Gerbaud, H. Nguyen Huu, B. du Peloux, H.L. Rakotoarison, F. Verdiere, F. Wurtz, An Optimizer using the Software Component Paradigm for the Optimization of Engineering Systems, COMPEL 2007.

[10] B.Delinchant, F.Wurtz, D.Magot, L.Gerbaud,A component-based framework for the composition of simulation software modeling electrical systems, Simulation, Transactions of the Society for Modeling and Simulation International, 2004.

[11] P. Enciu, F. Wurtz, L. Gerbaud, B. Delinchant, AD for Optimization in Electromagnetism Applied to Semi-Analytical Models Combining Composed Functions, COMPEL 2009.

[12] B. Delinchant, F.Wurtz, J.Fandino, Mixing of FEM and Analytical Modeling for the Preliminary Design of a Transformer, Optimization and Inverse Problems in Electromagnetism OIPE'02, 2002.

[13] E. Atienza, M. Perrault , F. Wurtz, V. Mazauric, J. Bigeon, A methodology for the sizing and the optimization of an electromagnetic release, IEEE Transactions on Magnetics, Jul 2000, Volume: 36 , Issue: 4, pp 1659 1663

[14] T. Bodrich,T. Roschke, A Magnetic Library for Modelica, International Modelica Conference, 2005

[15] A. Rezgui, B. Delinchant, L. Gerbaud, VHDL-AMS to Support DAE-PDE Coupling and Multilevel Modeling, IEEE TRANSACTIONS ON MAGNETICS, 2011

[16] V. Mazauric, From thermostatistics to Maxwells equations: A variational approach of electromagnetism, IEEE Trans. Magn., vol. 40, no. 2, pp. 945948, Mar. 2004

[17] SMASH, Foreign C functions in VHDL-AMS and SPICE models, Release 5.13 June 2009

[18] Y. HERV, P. DESGREYS, Functional virtual prototyping design flow and VHDL-AMS, Forum on specification and Design Languages (FDL'06), Darmstadt (Germany), 2006.

```
Entity linearactuator is
                r1 : real := 13.55e-3;
   generic (    h1 : real := 14.55e-3;
                ...

                C  : real := 100.0e-6;
                R  : real := 5.46063083;
                E  : real := 18.0);

 port (  quantity z    : out real;
         quantity v    : out real;
         quantity i    : out real  );

end entity ;
```

Actuator parameters

Electrical Parameters

States variables

```
Architecture behav of linearactuator is
Begin
        -- Initial conditions:
           break i  => 0.0; break z  => zmin;  break v  => 0.0 ;

        -- computation of Magnetic outputs from the ICar Component
        [Actuator_outputs] == FOREIGN_FUNC_CPP (Actuator_parameters);
        F_mag== Actuator_outputs[1];
        L_tot==Actuator_outputs[2];

        -- computation of residuals forces
        F_res==(-1)*1.24*(z+27.5)+59.1 ;
        F_tot==F_res - F_mag ;

        -- Electrical and Mechanical Equations
        di == i'dot;
        di'dot == -(R*di+i/C)/(L_tot);
        z'dot == dv;
        v'dot == acc;

        -- computation of velocity and acceleration values
        if ( F_tot>0.0 ) use
                if ( z<zmax) use
                acc == F_tot/masse;
                dv  == v;
                else
                acc == 0.0 ;
                dv  == 0.0 ;
                end use;
        else
                acc == 0.0 ;
                dv == 0.0 ;
        END USE;
end;
```

Link between VHDL-AMS and Software Component

Figure 9: The VHDL AMS model of the moving core actuator with a component ICar

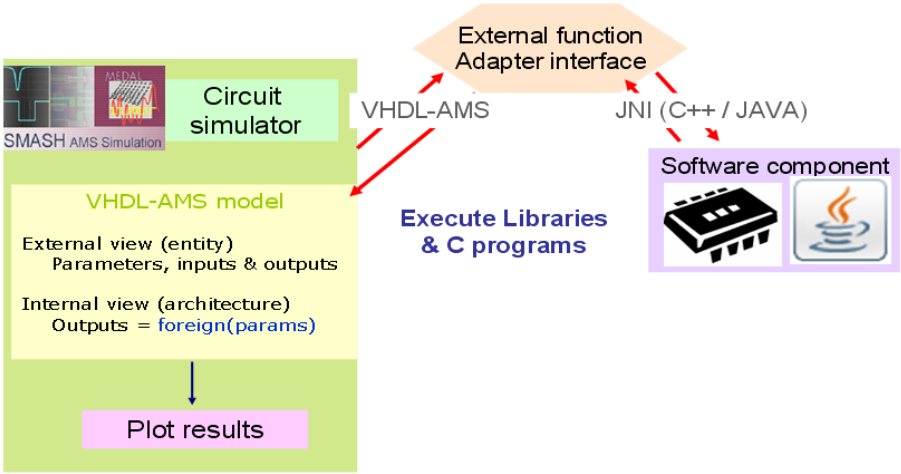


Figure 10: Exchanging of data between a VHDL-AMS model and an ICAr software component controlled by SMASH

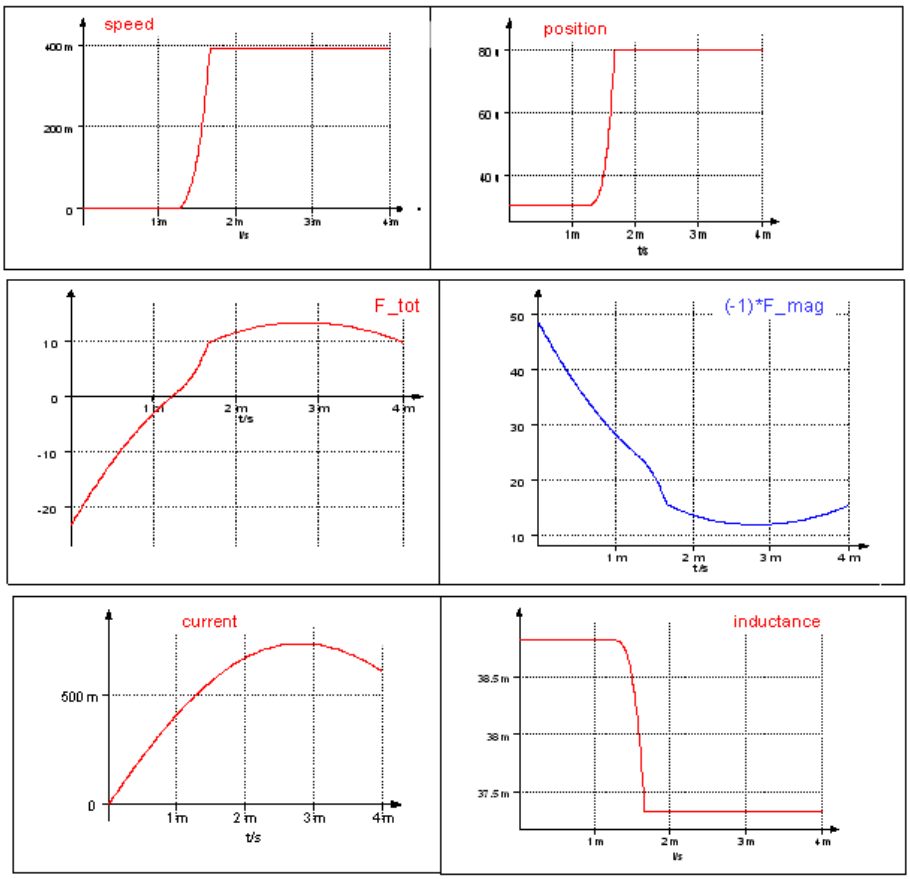


Figure 11: The behaviour of the actuator during simulation