Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-10 (2015) 075–080

Modelling and Simulation for the Integrated Design of Mechatronic Systems

Giacomo Barbieri * **Gabriele Goldoni** ** **Roberto Borsari** ** **Cesare Fantuzzi** *

* *University of Modena and Reggio Emilia, Via Amendola 2, 42122 Reggio Emilia (Italy) {giacomo.barbieri, cesare.fantuzzi}@unimore.it*
** *Tetra Pak Packaging Solutions SpA, Via Delfini 1, 41123 Modena (Italy) {gabriele.goldoni, roberto.borsari}@tetrapak.com*

Abstract: The development of mechatronic systems involves the use of multiple disciplines, from mechanical engineering to electronics engineering and computer science. Traditionally, every discipline was developed independently and then integrated to generate the final system. However, high quality designs cannot be achieved without simultaneously considering all the engineering disciplines. This mechatronic approach carries intrinsic complexity into system design process and numerous researches are on-going in order to find out optimal methods.
This article refines a SysML-based design process for the high level development of mechatronic systems, focusing on the integration of modelling and simulation as fundamental aspect for an integrated approach. How conceptual SysML diagrams may support the generation of simulation models is shown, along with a chain of simulations for an integrated design. The proposed approach is applied to a prototype case study designed from scratch, in order to be validated and to demonstrate its potentiality.

Keywords: SysML, Simulation, Modelling, MBSE, Mechatronic Systems.

1. INTRODUCTION

Mechatronic systems (MTSs) consist in the synergetic integration of mechanical engineering with electronic and intelligent computer control in the design and manufacturing of industrial products and processes, Harashima et al. (1996). The design of these systems requires a multidisciplinary approach through the integration of the single disciplines. Each discipline focuses on a particular aspect of the system and exploits different Domain Specific Models (DSMs) that should be merged and integrated. In order to cope with this situation, several tools implementing different capabilities are commercially available and start to have quite a large diffusion among industrial development communities (e.g. SysML [1], AutomationML [2], Lifecycle Modelling Language [3] etc.). However, the implementation of tools is not enough to develop performing systems, and optimal industrial processes are necessary.

The System Modelling Language (SysML) is a general-purpose graphical modelling language which is meant to become the "lingua franca" for MTSs modelling. Up to now, SysML has been mostly used as tool for reverse engineering processes; for example, Karban et al. (2011). Some tentatives were carried out in order to utilize it as supporting tool for the design process (e.g. Rosenberg and Mancarella (2010) for the development of an audio player, Weilkiens (2008) for an on-board computer for rental cars), but mostly for re-designing the behaviour of already existing systems concerning their interactions with the external world. In a previous work, Barbieri et al. (2014) introduce a SysML-based design methodology for MTSs. The integration of the involved domains is performed at two levels:

- Modelling (information) level; e.g. relating a requirement with the components which fulfil it.
- Simulation level; e.g. hardware in the loop simulation (HIL), in which the real controller is connected to a simulated physical model of the machine, Maclay (1997).

Purpose of this article is to link the two levels defined above. This link is pursued through the definition of a workflow. Then, the previous defined methodology is refined through the identification of a chain of simulations. These simulations can be performed in the different phases of the design process through the proposed workflow.

The paper is organized as follows: related works are described in section 2. Section 3 explains how to integrate modelling and simulation, and the chain of simulations inserted in the methodology. Section 4 resumes the design process and applies the proposed approach to a case study. Conclusions and future works are reported in section 5.

2. RELATED WORK

2.1 Model-Based Design

Model-Based Systems Engineering is the formalized application of modelling to support system requirements,

[1] SysML: http://www.omg.org/spec/SysML
[2] AutomationML: http://www.automationml.org/o.red.c/home.html
[3] Lifecycle Modelling Language: http://www.lifecyclemodeling.org/

10.1016/j.ifacol.2015.08.111

design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases, INCOSE (2007). In the last years, this concept has been recognized as fundamental for a mechatronic design. In fact, there is the need to develop the designed system through an unique core model which works as the repository and dependency trace of the information of the different mechatronic domains. This is the idea of Model Based Design (MBD), El-Khoury et al. (2005). AutomationML and SysML may be used for this purpose.

AutomationML, introduced by Drath et al. (2008), reuses mature data formats to store information about topology (through CAEX), geometry and kinematics (through COLLADA), and logic (through PLCopen XML). The language is proper for connecting information contained in DSMs; e.g. the dimensions of components in CAD drawings. Whereas, SysML is used at higher level: for tracing and (above all) relating DSMs, but not necessarily the information contained into them. We think that the central model must be the most general as possible for containing information of all the involved domains. For this reason, we decided to utilize SysML for the application of the MBD approach. However, MBD is just a tool for realizing the integration among the different domains of MTSs and must be supported by an integrated design methodology.

2.2 Link conceptual and domain specific models

MBD implies the exchange of information among the central SysML model and all the tools necessary for the design. There are two different ways for reaching this task:

- model transformation for generating DSMs from SysML models (either "manually" or automatically); for example SysML to Modelica, Paredis et al. (2010);
- connection with existing DSMs: utilization of the SysML model as a system representation and a repository of the system parameters and the DSMs. Different plugins are being born for implementing this connection (e.g. ParaMagic [4]).

Current tendency is to have a high level SysML model which contains an overview and traces dependencies among DSMs, and detailed SysML views which allow the one to one translation in DSMs, Shah et al. (2010). Every SysML view is modelled through a profile (Domain Specific Language, DSL) for mapping the semantic of the DSMs into SysML. Model Driven Engineering is also included in this category for the software viewpoint. Here, a higher layer of abstraction (e.g. SysML model, Chiron and Kouiss (2007)) is created and deployed in control code (e.g. Vogel-Heuser et al. (2005)).

However, these approaches lead to detailed SysML models with specialist views difficult to manage and to understand for people of other engineering domains. Moreover, unlimited extensions of the language would be generated, "destroying" the aim of a general purpose modelling language. For these reasons, we only create high level SysML models and we hyperlink them to DSMs.

[4] ParaMagic: http://www.intercax.com/products/paramagic

2.3 Conclusion

In order to integrate mechatronic domains during design process, there is the need to identify a method for integrating SysML modelling and simulations performed in DSMs. This should be reached without extending the SysML language with DSLs. A proposal is illustrated in Sec. 3.

3. INTEGRATION OF MODELLING AND SIMULATION

3.1 The proposed workflow

Following the criteria defined in Sec. 2, we propose the workflow shown in Fig. 1, which consists of:

- Conceptual model: SysML diagrams for rationalizing what will be implemented in the simulation tool. Which diagrams to utilize depends on the performed simulation. For example, for dynamics simulations, an internal block diagram (ibd) can be used for representing the structure, and a state machine diagram (stm) for the behaviour.
- DSM implementation: implementation (and not one to one translation) of the conceptual SysML diagrams in a simulation environment and model execution. Then, SysML model is hyperlinked to the generated DSM in order to trace its (file) path. A possible implementation at the example of a dynamics simulation performed in MATLAB [5] is described next:
  (1) Each block of the ibd becomes a sub-system in the Simulink environment. Each block implements the dynamics equations of the represented object.
  (2) Parameters are assigned to the equations (e.g. dimensions, friction coefficients etc.).
  (3) All the sub-systems are connected as indicated in the ibd for generating the system model.
  (4) The behaviour represented in the stm is implemented in Stateflow.
  (5) An integrated simulation is performed linking the Simulink and Stateflow models.

  Comparing the simulation implementation described above and a standard modelling process, it can be noticed that the validation of the model is missing. Validation is generally performed through the comparison of simulation results with experiments. However, a physical prototype is usually not available in all the phases of the design process. In fact, different alternatives for fulfilling a functionality can be developed and the generation of a physical prototype for each alternative might result in a long and expensive activity. For this reason, simulation is performed for comparing alternatives modelled assigning same values to the common parameters. In this way, alternatives are evaluated on the basis of their plausible behaviour. When the real physical behaviour is identified, one can perform again the comparison in order to validate the results.
- Failure-modes analysis: evaluation of all the possible failures and determination of new functions for making the system work correctly; e.g. definition of functions for fault detection and fault management.

[5] MATLAB: MathWorks www.mathworks.com