A Study on SysML and Altarica Model Synchronization

Aroua Berriche, Faïda Mhenni, Abdelfattah Mlika[3], Jean-Yves Choley

▶ **To cite this version:**

Aroua Berriche, Faïda Mhenni, Abdelfattah Mlika[3], Jean-Yves Choley. A Study on SysML and Altarica Model Synchronization. IC3M, Dec 2019, Sousse, Tunisia. hal-02546655

HAL Id: hal-02546655
https://hal.science/hal-02546655

Submitted on 18 Apr 2020

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.

A Study on SysML and Altarica Model Synchronization

Aroua Berriche[1], Faïda Mhenni[2], Abdelfattah Mlika[3], Jean Yves Choley[4]

[1] LMS Laboratory & Quartz Laboratory, Supmeca, 93400 Saint-Ouen, France,
aroua.berriche@supmeca.fr

[2] Quartz Laboratory, Supmeca, 93400 Saint-Ouen, France,
Faida.mhenni@supmeca.fr

[3] LMS Laboratory, ENISo, 4054 Sousse, Tunisie,
Abdelfattah.mlika@gmail.com

[4] Quartz Laboratory, Supmeca, 93400 Saint-Ouen, France,
jean.yves.choley@supmeca.fr

Abstract. The development of a mechatronic system involves the use of multiple models from a variety of domains. These models are created by different actors using a variety of modeling languages, formalisms, and tools for addressing specific concerns and are used for representing different views on the same system. The separation of concerns is considered as a good practice to develop every model independently from other disciplines. However, a complete separation is impossible. The integration process can produce communication issues and poor understanding between various actors. The main consequence is a high risk of inconsistency between the different models of the same system. In this paper, we apply a model synchronization methodology to detect inconsistencies between the different models of a mechatronic system. The proposed method is composed of three phases: first, the entry models are abstracted into a common representation; second, a comparison between the abstracted models is carried out to identify potential inconsistencies among various models, and finally, the concretization phase acts to solve the detected inconsistencies.

Keywords: System engineering, Safety analysis, SysML, Altarica, Consistency management.

1 Introduction

Because of its multi-disciplinary nature, the development of a mechatronic system requires the consolidation of models from a variety of disciplines such as mechanics, electronics, and software engineering among others. Various designers having different perspectives on the overall system usually create these models using diverse formalisms. As different actors perform these models, they may present some inconsistencies.

The contribution of this paper is to provide a first step towards consistency management of heterogeneous models involved in the development of mechatronic systems, using a cooperative process to support the exchange between different designers. The methodology allows multi-disciplinary interactions between multiple designers at an early stage of the development process that improves competitiveness and reduces development time and cost.

The remainder of the paper is organized as follows. Section 2 depicts similar works that deal with assuring consistency between models. Section 3 gives a presentation of the proposed methodology. In section 4, the methodology is illustrated in a case study from the aircraft industry. Finally, the conclusion is given in the last section.

2 Related work

To improve the collaboration between various designers integrated into the design process of a complex system, four types of approaches are typically used:

2.1 Integration approach

This approach proposed to incorporate the different disciplines-specific views in a single model. For example, to integrate safety analysis in the system engineering process, Mauborgne et al. (Mauborgne et al., 2015) proposed to incorporate safety properties on system architectures' viewpoints. Moreover, CATIA V6 (Kleiner and Kramer, 2013) is presented as a commercial application, which proposes a single tool with multiple views. This approach allows different disciplines to be integrated and be managed in a collaborative manner.

2.2 Model transformation approach

This approach provides a mapping from one discipline to another. Two technologies are used to apply this approach:

Firstly, the use of profiles or SysML (System Modeling Language) extensions to enrich SysML models, transforming system models into another language as

Modelica Language, the two well-known profiles are SysML4Modelica (Paredis et al., 2010) and ModelicaML (Schamai et al., 2009) linking SysML to Modelica.

Secondly, based on language transformation such as Triple Graph Grammar, Adourian et al. (Adourian and Vangheluwe, 2007) had built a meta-model of the relation between geometric (CAD) models of a mechanical system and the corresponding dynamic simulation models to assure consistency between the two views.

2.3 Federative approach

Guychard et al. (Guychard et al., 2013) proposed the federative approach that aims to build a unique database to store all the model data and then execute a partial projection to generate the models for each tool.
Thramboulidis (Thramboulidis, 2013) proposed a framework to implement the federative approach using the powerful and rich semantics of the SysML language.

2.4 Inconsistency management approach

This approach proposed to focus exclusively on managing inconsistencies.
Gausemeier (Gausemeier, 2009) synchronized domain-specific models with a cross-domain system specification based on model transformation. By that, domain-specific models could be derived initially, and changes in one model could be propagated via the cross-domain specification.

While these approaches allow consistency management between domain-specific models, they have several practical limitations. For one, the integration approach requires that designers must be adaptive in order to design their model with a single tool. Since the model transformation approach considers oriented relations encoded in the transformation rules. Therefore, certain model transformation does not guarantee consistency between each model developed. Moreover, the federative approach and the inconsistency management approach can be criticized by the fact that the development of such technologies requires huge effort of encoding large amounts of knowledge and information to manage consistency between models.

Therefore, the contribution of this paper is to provide a first step towards effective managing inconsistencies in heterogeneous models, using a cooperative method to support dialog between various actors. In this paper, we have selected two particular but representative modeling languages for illustrating our cooperative approach: SysML for systems engineering and Altarica for safety engineering.

3 Methodology

To provide a first step towards model synchronization of mechatronic systems, a conceptual approach is proposed in this section.
The suggested approach consists of identifying, detecting and managing differences and inconsistencies during the process of design mechatronic systems. This approach, illustrated in Fig. 1, is based on three phases: abstraction, comparison, and concretization. These three phases will be described in the following.

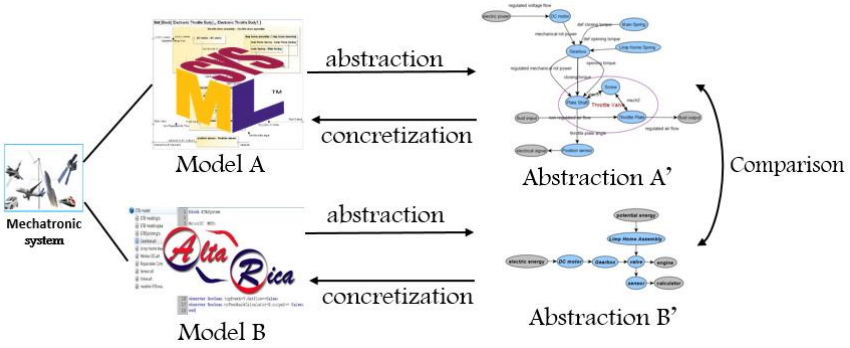


Fig. 1 Model synchronization approach

3.1 Abstraction

The first phase includes the representation of entry models (SysML, Altarica) in a common formalism using graph theory (Ruohonen, 2013). We assume that the abstraction applies to model-to-model transformation (Mens and Van Gorp, 2006). Table 1. presents an overview of structural constructs of the modeling languages SysML and Altarica and shows how these constructs can be transformed into a directed graph.

Table 1. Mapping between SysML, Altarica and directed graph constructs

SysML (IBD)	Altarica	Directed graphs
Part	Block/Class	Vertex
Port	Flow variable	Vertex
Connector	Assertion	Edge

Each part in an IBD model is transformed into a block or a class in Altarica model and into a vertex in a directed graph. In addition, each port in an IBD model can be transformed respectively into a flow variable and vertex in Altarica and directed graph models. Connectors in an IBD model represent connections between parts via its ports through which energy or information is exchanged. They can be mapped to assertions in Altarica model and to edges in a directed graph.

3.2 Comparison

In order to identify differences and inconsistencies between abstracted models, a subgraph isomorphism algorithm is developed. This algorithm executes three principal activities:

- Search for possible isomorphism between graphs.
- Search common subgraphs between abstracted models.
- Detect missing nodes and edges in a graph compared to another.

An "interface" expert generates a report that classifies the differences and inconsistencies between the abstracted models according to the comparison algorithm results. The differences are authorized because they represent the specification of different modeling languages, while the inconsistencies should be analyzed and solved by the "interface" expert and then validated by designers.

3.3 Concretization

The last phase allows updating the source models using abstracted models. This latter will be annotated with the necessary corrections proposed by the "interface" expert according to the detected inconsistencies from the previous phase. This phase will be implemented using the model-to-model transformation technique. As a result, we obtain consistent information between the different views of a global system.

4 Case study

The studied system is an Electro-Mechanical Actuator (EMA) onboard an aircraft. An EMA is composed of three interconnected equipment, as shown in Fig. 2. An electric motor, a mechanical transmission and an electronic and software part composed of a calculator that controls the system.

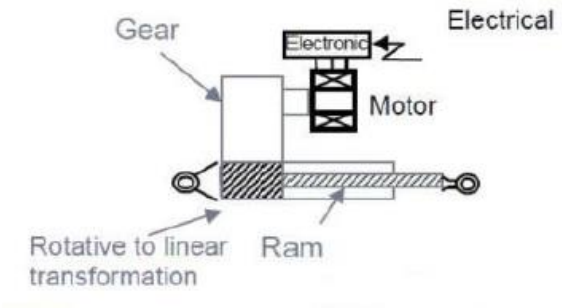


Fig. 2 Electro-Mechanical Actuator architecture

In our scenario, we consider two different domain-specific models of the EMA system during the engineering process.
A SysML model is developed for outlining the physical architecture design of our case study using an IBD diagram. An Altarica model that assesses the dysfunctional behavior of the system.
In the following, these models are represented in more detail.

4.1 System Engineering Perspective (SysML)

SysML (OMG, 2012) is a general-purpose graphical modeling language specified by OMG that supports the analysis, specification, design, verification, and validation of complex systems including hardware, software, data, procedure among others. The SysML model is used for the purpose of formally capturing requirements, specifying the physical decomposition and describing the behavior of the system. An IBD diagram represents the internal architecture of a system and models the interconnections between components.

Fig. 3 gives an overview of the internal structure of the EMA system using an IBD diagram. The main function of the EMA is to control the aileron angle of an aircraft. This function is achieved by three components: the MCU that controls the system, the geared motor that provides the electric power to the system and the mechanical transmission that transforms the electric energy into mechanical power to actuate the aileron of the aircraft.

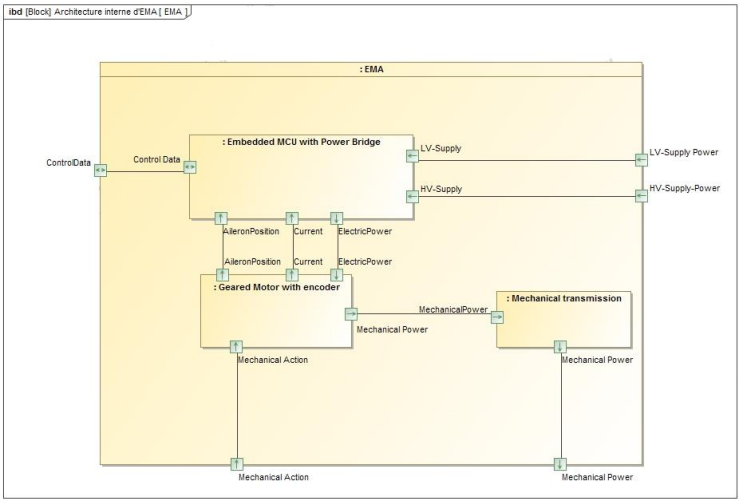


Fig. 3 Decomposition of the EMA system (IBD diagram)

4.2 System Safety Engineering (Altarica)

Altarica (Altarica Association, 2017) is a formal modeling language dedicated to safety analysis. Altarica semantic is defined using Guarded Transition Systems (GTS) and as a structural paradigm, it is based on a System Structure Modeling Language (S2ML) as discussed by Prosvirnova (Prosvirnova, 2014). Models, specified using the Altarica language, can be compiled into a lower-level formalism such as fault trees and other safety assessments can be performed.

In Fig. 4, we represent the global model of the EMA system in Altarica. It is composed of a gear motor, a screw-nut assembly, and a carter. The assertion represents the connections between components, and observers are defined to calculate the reliability indicators.

RepairableComponent.alt | gearmotor.alt | screw_nut.alt

```
include "ComponentLibrary/carter.alt"
include "ComponentLibrary/gearmotor.alt"
include "ComponentLibrary/screw_nut.alt"

block ElectricalMechanicalActuator

    // external flows declaration
    Boolean electrical_power,mech_m (reset=true);

    // components declaration
    carter c;
    gearmotor gm;
    screw_nut sn;

    // GTS structure
    Boolean vfOut(reset=false);
    parameter Real pLambda= 0.1;
    event efailure (delay= exponential(pLambda));
    transition
    efailure:gm.working ->gm.working:=false;

    // interactions between components
    assertion
    gm.mech1 :=: c.mech1;
    gm.mech2 :=: sn.mech2;
    sn.mech3 :=: c.mech3;

    gm.alim:= electrical_power;
    c.mechanic_motion:=:mech_m;

    electrical_power:= true;
    mech_m:=true;

    // top event declaration
    vfOut:=gm.mech2;
    observer Boolean oMovement=vfOut;
    observer Boolean isNotMovement=not vfOut;
    end
```

Fig. 4 Safety analysis of the EMA (Altarica)

4.3 Methodology application

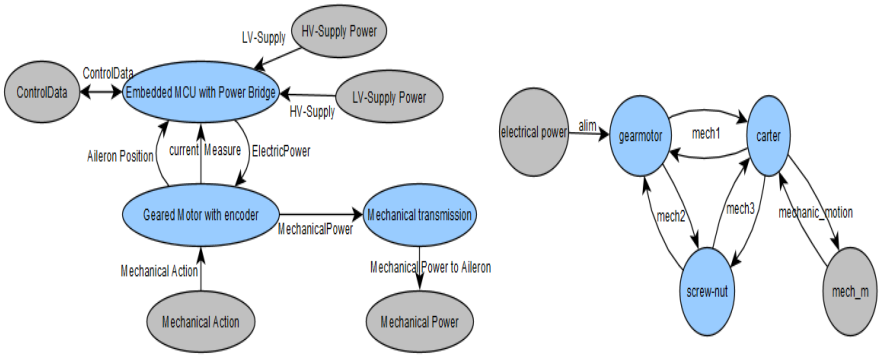
We illustrate in the following, the different phases of our methodology based on the EMA case study.

- **Abstraction**

The source viewpoints of system engineering and safety engineering of the EMA system are subject to a model transformation to obtain two abstracted views, retracing the structure of different models. As discussed in the previous section, we propose to transform our entry models into directed graph representations, a

graphical representation of SysML model and Altarica model are represented in Fig. 5.

We define by light blue vertices components belonging to the studied system and by gray vertices representing the external elements of the system.



(a) SysML-IBD directed graph model (b) Altarica directed graph model

Fig. 5 Abstraction phase

- **Comparison**

We compare our entry models using their directed graph representations. The comparison is a semi-automatic and an iterative process.

We represent in Table 2. the results of the comparison process between SysML and Altarica abstracted models.

Table 2. Results of the comparison between SysML and Altarica models

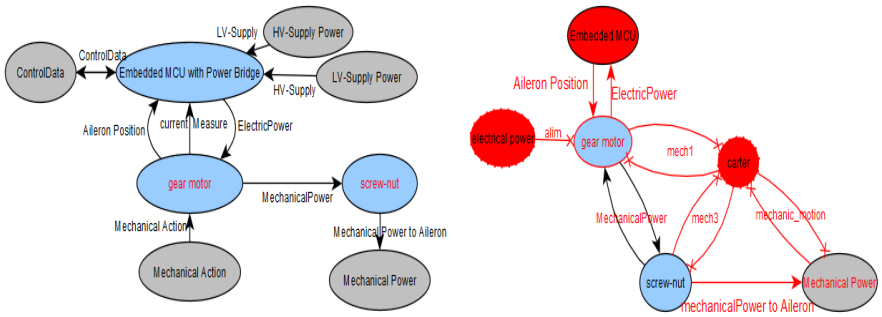
SysML	Altarica	Differences	Inconsistencies
Geared Motor	gearmotor	Labeling	-
Mechanical transmission	Screw-nut	Labeling	-
Mechanical Power	mech-m	Labeling	-
ElectricPower	electrical power	Labeling	connector corresponding to block
Embedded MCU	-	-	Concern
-	carter	-	adding an element
Mechanical power to Aileron	-	-	deleting an element

This case study allows detecting architectural differences between the different abstracted models.

- **Concretization**

An “interface” expert analyzes the differences detected in the comparison phase in order to propose a list of corrections that will be annotated in abstracted models. The Fig. 6 represents the different corrections proposed by the “interface”

expert in the SysML and Altarica directed graph models. The annotations are represented in red vertices and (or) edges.



(a) SysML-IBD directed graph model (b) Altarica directed graph model

Fig. 6 The annotation of abstracted models

After verification and validation of corrective actions proposed by the "interface" expert, a model transformation will be executed on the annotated abstracted models to update entry models. The concretization phase allows updating entry models with chosen compromises between the different designers and the "interface" expert.

As a result, we ensure that the different models include consistent information to design the EMA system. Meanwhile, the specificities of each modeling language are guaranteed.

5 Conclusion

Traditionally, system engineering and safety engineering aspects are described in different modeling languages (e.g. SysML and Altarica). As a result, the system-engineering model is decoupled from the safety engineering models. Therefore, the risk of inconsistencies is high. A complex system can fail due to miscommunication among designers and resulting in wrong decisions taken during the design process. In this paper, we propose a methodology to evaluate the consistency of mechatronic systems through the multi-view modeling process. We showed how the proposed approach covers all phases of early detection of inconsistency problems in mechatronic systems design between different views such as system engineering view and safety engineering view for safety assessment. As future works, we will investigate the possibility of applying our conceptual approach to evaluate the behavior consistency of a complex system.

References

Adourian, C., and Vangheluwe, H. (2007). Consistency between geometric and dynamic views of a mechanical system.

Altarica Association (2017). AltaRica 3.0 Language Specification Version 1.1.

Gausemeier, J. (2009). Management of cross-domain model consistency during the development of advanced mechatronic systems.

Guychard, C., Guerin, S., Koudri, A., Beugnard, A., and Dagnat, F. (2013). Conceptual interoperability through Models Federation. 23.

Kleiner, S., and Kramer, C. (2013). Model Based Design with Systems Engineering Based on RFLP Using V6. In Smart Product Engineering, M. Abramovici, and R. Stark, eds. (Berlin, Heidelberg: Springer Berlin Heidelberg), pp. 93–102.

Mauborgne, P., Deniaud, S., Levrat, E., Bonjour, E., Micaëlli, J.-P., and Loise, D. (2015). Preliminary Hazard Analysis Generation Integrated with Operational Architecture - Application to Automobile. In Complex Systems Design & Management, F. Boulanger, D. Krob, G. Morel, and J.-C. Roussel, eds. (Cham: Springer International Publishing), pp. 297–309.

Mens, T., and Van Gorp, P. (2006). A Taxonomy of Model Transformation. Electron. Notes Theor. Comput. Sci. *152*, 125–142.

OMG (2012). OMG Systems Modeling Language (OMG SysML™).

Paredis, C.J., Bernard, Y., Burkhart, R.M., Koning, H.-P., Friedenthal, S., Fritzson, P., Rouquette, N.F., and Schamai, W. (2010). 5.5. 1 An Overview of the SysML-Modelica Transformation Specification. In INCOSE International Symposium, (Wiley Online Library), pp. 709–722.

Prosvirnova, T. (2014). AltaRica 3.0: a Model-Based approach for Safety Analyses.

Ruohonen, K. (2013). Graph Theory.

Schamai, W., Fritzson, P., Paredis, C., and Pop, A. (2009). Towards Unified System Modeling and Simulation with ModelicaML: Modeling of Executable Behavior Using Graphical Notations. pp. 612–621.

Thramboulidis, K. (2013). Overcoming mechatronic design challenges: the 3+ 1 SysML-view model. Comput. Sci. Technol. Int. J. *1*, 6–14.