Toward Model Synchronization Between Safety Analysis and System Architecture Design in Industrial Contexts

Anthony Legendre, Agnes Lanusse, Antoine Rauzy

Toward model synchronization between safety analysis and system architecture design in industrial contexts

Anthony Legendre[1], Agnes Lanusse[1], and Antoine Rauzy[2]

[1] CEA, LIST, Laboratory of Model Driven Engineering for Embedded Systems, F-91191 Gif-sur-Yvette, France, anthony.legendre@cea.fr,agnes.lanusse@cea.fr,
[2] NTNU, S. P. Andersens veg 5, 7491 Trondheim, Norway, antoine.rauzy@ntnu.no

Abstract. Classical organization in disciplinary silos in the industry reaches its limits to manage complexity: problems are discovered too late and the lack of communication between experts prevents the early emergence of solutions. This is why it is urgent to provide new collaborative methods and ways to integrate various engineering fields, early in and all along the development cycle. In this context, we are particularly interested in the possible exchanges between two engineering fields: system architecture design and safety analysis. The questions are: how can one ensure that the parties involved are speaking about the same system? And which concepts can synchronize several engineering fields? First we present a use case: a system embedded in a helicopter. Second we present the concepts that we define to implement synchronization of models. Finally we give our feedbacks, limits and related works.

Keywords: Model Based Safety Analysis, Models Synchronization, Integration in Interdisciplinary Processes, Model-Driven Engineering, System design, Safe Complex systems.

1 Introduction

Engineers solicitation to assess new proposal of critical systems (particularly new architectures) in terms of safety performances is increasingly important. They are more and more asked for a fast feedback on the design choices coming from upstream stages of the development cycle, without providing them reliable or accurate sources of information. In this context, we are particularly interested in system architecture design and safety analysis that play major roles in a critical system development.

Mathematical framework are the core of risk and safety assessments since the beginning of the discipline with dedicated artifacts such as fault tree, event tree, markov chain and the like. However, such models are still poorly connected with design models. Indeed, analysis started from paper documentation (issued by others disciplines). Information was captured manually into dedicated safety analysis tools and/or spreadsheets. This era of document based safety assessment is now leaving the way to MBSA (Model-Based Safety Assessment) where

information on the architectures and behaviors of the system come from various models and contexts, especially from system design models. This observation could be generalized on major disciplines of engineering. They virtualize their contents to a large extent, i.e. they are designing models. This is the era of model-based. Currently, the design/production/operation/decommissioning involves the design of dozens if not hundreds of models. Models are designed by different teams in different languages at different levels of abstraction, for different purposes. They are strongly linked to various activities implied by processes. Complexity impacts not only these products but also the processes involved in modeling tasks.

Today, relations between models and activities are not clearly formalized. Often done manually, interactions are time consuming and error prone. It may introduce mistakes by misinterpretation of models produced by different disciplines. This can bias the becoming system. This way of proceeding is risky and more and more difficult to deal with as complexity increases. No effective means are deployed to ensure consistency between these engineering fields. Indeed, although there is a great expertise in the Model Driven Engineering (MDE) community, its generalization to the whole industry is still a huge challenge.

The purpose of this paper is to present the results of a reflexion on the concepts, practices and recommendations that are useful to implement synchronization of models in an actual industrial context. This work is focused on Safety Analysis and Assessment issues and their synchronization with Architecture Design. In this paper, we present an industrial case study: a fire detection system embedded in a fighting helicopter in *section 3*. Then, we present the concepts that we use to support our methodology: environment of synchronization, the configuration and applications of models synchronization in *section 4*. In particular, we introduce the notion of need of exchange and point of synchronization that permit to identify and specify relations between models. Finally, we give our results, feedbacks and limits on this approach and quote some related works in *section 5*.

2 Related work

The context of this work is intended to support systems engineering where global views of systems is required and where interplay of different fields is important to capture and order requirements corresponding to multi-objective concerns. It also targets the elaboration of consistent solution in an incremental and cooperative way.

To carry MBSA approaches in accordance with MBSE, researchers explored several clues. Some are trying to incorporate safety properties on system architecture viewpoints [13]. Others attempt to add safety properties on the architecture models to drive safety analysis [22], [7]. Technologies are based on properties annotations (profile for SysML [15], Error annex for AADL [8] or EAST-ADL [3], [5]). These approaches may be criticized because they consider only oriented

relations from system architecture design to safety analysis. Most of the works are strongly tool oriented, and not enough cooperative.

Finally, some propose cooperative techniques (also called federative approaches) [9], that attempt to establish relationships between elements of models with different concerns. They conceptualize way to ensure consistency between heterogeneous viewpoints. They permit to build cross-concerns views, while maintaining traceability relations in order to ensure global consistency. In [21] a framework to implement synchronization links between model elements is proposed. They don't consider the needs of semantic synchronization between activities, but in the future their results could be used to support synchronization. Concerning semantic mappings we found that model weaving, as seen by [6], is an interesting approach to define dependencies between models. Many works related to ontology [2] could be profitable to support mappings and traceability as well as conflicts resolution. However, few contributions were found on both engineering fields.

In this paper, the position of the approach is a cooperative application that tries to manage models between MBSE and MBSA approaches. It supports dialog between engineering teams. It manages interfaces from several modeling contexts on different concerns to get a global cohesion. It is an iterative application that builds consistency of models used by different fields.

3 Case Study

The studied system is a fire detection system onboard a fighting helicopter. The system's mission is to detect fire events in three specific areas in the helicopter. The areas concerned are: the main engine, the secondary engine and the main rotor. This automatic fire alarm system is designed to detect the unwanted occurrence of fire by monitoring environmental changes associated with combustion. The system is intended to notify the helicopter crew and airport on ground.

This system is composed of four interconnected equipments, as shown in the Figure 1: a set of sensors, an alert device, a power supply and fire-fighting equipment.

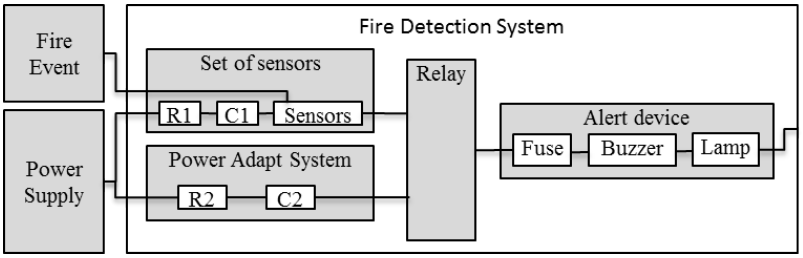

Fig. 1. Composite view of fire detection system

The case study has been considered in system architecture design and safety analysis concerns [11]. We consider the recommendation of following avionics standards: ARP 4761 [19] and ARP4754 [20] for the development and the safety assessment. In the article the fire detection system will permit to illustrate our argumentation by showing case studies focused on specific activities (illustrated by viewpoints) or sequence of activities.

4 Principles of synchronization

Models used during an application's life-cycle are multiple and quite heterogeneous. They are strongly connected to the processes and activities achieved all along the development. Their nature depends largely on levels of abstraction, and purposes. To be able to compare them, we have chosen to rely on architectural concepts reflecting their structure. The principles proposed for synchronization are organized according to three activities:

- defintion of the context and identification of the needs of exchanges,
- configuration of the synchronization,
- application of synchronization mechanisms based on: abstractions, comparison, concretization.

4.1 Definition of context

To build a model synchronization, we first try to define the contexts that characterize the models involved according to processes and activities used in the application domain (and generally required by standards). In this section, we define concepts of business contexts modeling and apply them on the case study.

One context definition is considered for each domain. It describes processes, activities, methods and viewpoints used. In a second step, it will allow to look for possible exchanges between models according to the activities concerned in the processes.

The figure 2 illustrates the contexts definition stage applied on the case study. It brings out concepts allowing the structuration of processes, activities and viewpoints. In this case it represents two engineering processes (system architecture and safety analysis) in a single formalism. It is focused here on operational analyses applied to the fire detection system. In a second step, it will allow to look for possible exchanges between the models.

The first context sets the definition of the environment and the operational system's interaction with the environment. The business process consists in three activties: definition of usecase, definition of scenario and definition of system's life cycle. Each activity applies a method and the method relies on the chosen viewpoints.

The second context considers the Safety Analysis according to ARP4761 at aircraft level. The purpose is to identify and prioritize unexpected events at aircraft level. The business process consists in three activties: Preliminary

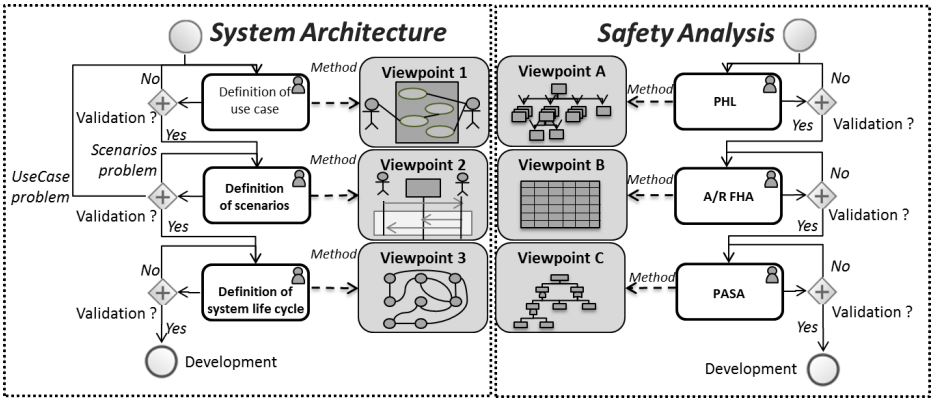

Fig. 2. Application of the definition of the context

Hazard List (PHL), Aircraft (FHA) and Preliminary Aircraft Safety Analysis (PASA). Each activity applies a method and the method relies on the chosen viewpoints.

Concepts. In this section we present definitions to describe the context of study or analysis of engineering fields. We define which engineering fields, purposes of analysis, models and elements will be able to interact with the synchronization and when ? We chose concepts as generic and as close as possible to industry terms. The definition of the context will allow, during the synchronization, to consider and manage multiple levels of abstraction, several interleaving between models used at different activities. The figure 3 spells out the link between each main concept and concepts from ISO42010.

Business context: It is an abstract notion that includes all the skills: knowledge, know-how and soft skills specific to a field of study or analysis. It has a purpose (or set of objectives), usually to deliver a service, product or specific result. The engineering field can be considered as a business context.

Business process: It is a collection of structured activities (also called tasks), that produce a service in a specific business context. Business processes are often represented by using flowcharts containing sequences of activities, interleaved decision points and fork/join. The sequence of activities is chaining activities from directional flow line. A flowchart is a mathematical framework that depicts a process behavior. It is widely used in multiple fields to study, plan, improve and communicate on complex processes in clear, easy-to-understand diagrams. Languages are defined: Business Process Modeling and Notation (BPMN) [14], or Process Flow Diagram (PFD) [10], Software Process Engineering Metamodel (SPEM). Some are related to other diagrams, such as Data Flow Diagrams (DFDs) [18] and Activity Diagrams of UML [16]. In the case study, the chosen representation was BPMN, it supports business context, business process, activities, method and viewpoint concepts. Other formalisms have not been tested.

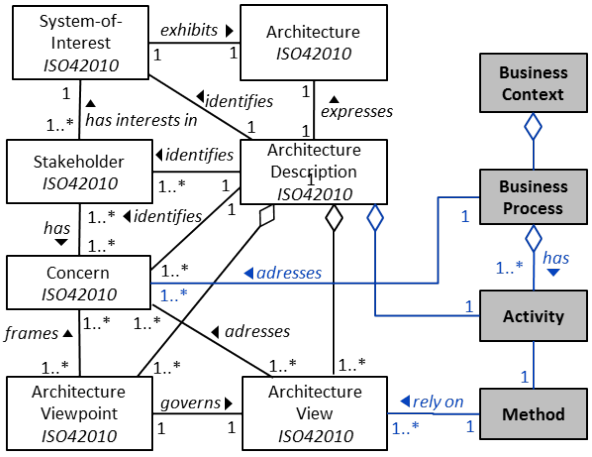

Fig. 3. Definition of the context

The decision point and fork-join allows to represent a division or to group several sequences of activities according to alternative or parallel sequences (temporally or logically).

Activity: It is the application of an accurate method at a given moment in a business process. It clearly defines input and output models whose added value are measurable. Interim models may exist to represent intermediate results. Thus an activity serves a clearly defined objective (sub-purpose of business context). It is possible to consider a condition (guard) on the feasibility of the activity according to the maturity of the input and/or the accomplishment of the upstream activities.

Method: It is a logically ordered set of principles, rules, steps, which is a means of achieving a desired result which reponds to the objective of its activity. A method usually relies on a viewpoint. Remark: A method can be used by different activities in several contexts.

Viewpoint: According to ISO 42010 [1], a viewpoint is a frame that shall spotlight a concern (or part of a concern) identified by the stakeholders. To do so, a viewpoint relies on models. To complete the definition, a viewpoint is an abstraction of data included in a model. This abstraction is built to spotlight specific concerns into a model. As "model", a viewpoint is defined by a metamodel, a formal definition or a language.

Model: Literature is extremely large about the definition and use of the term "model". ISO 42010 does not define the term but tries to give cases of use in the international standards. This is unsatisfactory for our work; therefore we propose the following definition: a model is an abstract capture of a practical or intended reality. Like the viewpoint, the model appeals to the cognitive faculty of the modeller. Each user builds its own (unique) interpretation of the reality and model in his mind.

In the context of system engineering, the model is an artefact. It could be characterized by two major criteria: the nature and the purpose.

The nature of a model can be defined by conceptual, mathematical, informatical (language, algorithm) or graphical (representations, structure, behavior) frameworks. The purpose of a model relies on the business context or a subset of this one. A model is defined for a specific purpose and highlights results or problems according to this purpose. We distinguish three kinds of model purposes: model to communicate, model to calculate, model to generate.

The defined concepts have been identified on the case study. It has been considered in system architecture design and safety analysis concerns. The following table lists activities conducted by the two engineering fields (Generic process for System architecture and ARP4761 for Safety program).

System Architecture		Safety Program Plan	
Operational analysis	System environment and use case definition	**Safety Analysis Level Aircraft / Helicopter**	Aircraft FHA
	Definition of scenarios		PASA
	System life cycle definition	**Safety Analysis System Level**	System FHA
Functional Architecture Design	Functional decomposition system		PSSA
	Internal functional architecture	**Safety Analysis Equipment Level**	System FTA
	Functional behavior		System FMEA
Physical Architecture Design	Physical decomposition system	**Commun causes analyses**	Aircraft CCA
	Internal Physical architecture		System CCA
	Physical behavior of components		PRA & CMA & ZSA

Fig. 4. Activities led by both expertises for the fire detection system

From figure 2 or 4, we can define relationships between points of view. We call these relationships: needs of exchange because they will involve discussions on the meaning of models.

Identification of the need of exchanges. The identification of relations between viewpoints open four questions: What are the needs of the engineering field vis-à-vis another field? Why do they need these exchanges? When, in the business processes, do we need exchanges? What do we want to exchange (what model elements, properties)?

We define a need of exchange as a clear selection of activities in both business processes where there is a need to establish a consistency between the models. This corresponds to a formalized need to share a model by two business contexte which handle elements of dependent models. The needs of exchange can refine studies through the decisions taken when pooling business context across viewpoints. The need of exchanges consists in three attributes:

- The main need and personal interest of each engineering field,
- The processes, activities, methods and viewpoints involved in the exchange,
- The elements and properties which depend on this exchange.

The identified need of exchanges shall necessarly comply with the definition of the context, i.e. to answer the previous questions, engineers should use defined business processes, activities, methods and viewpoints. This step needs an important maturation of "what it really needed to exchange ?".

In the case study, we tried to identify a need of exchange [12]. Around 50 needs of exchange have been identified.

4.2 Configuration of the synchronization

The configuration consists in identifying and formalizing possible exchanges between engineering fields. This configuration will define the implementation of the synchronization using interactive methods presented in *section 4.3*. The need of exchanges is subjective. However, it is interesting to formalize needs of exchanges into a more formal concept: the point of synchronization. This concept is consistent with the definition of context and the application of the synchronization. It considers three sub-concepts (cf. figure 5).

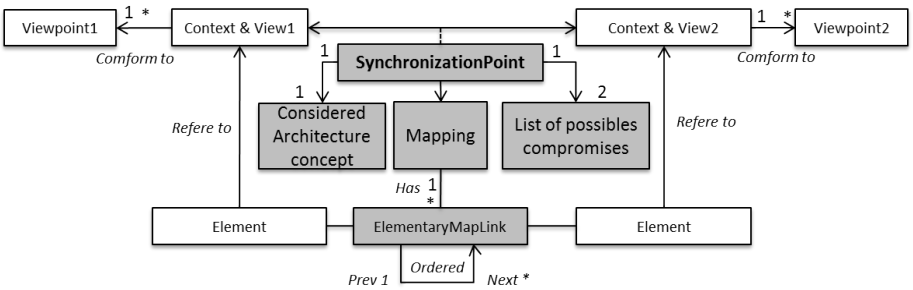

Fig. 5. Formalization of the point of synchronization

The considered architecture concepts capture part of system architecture that we want a consistency establishment. Indeed, we cannot build consistency of anything in only one try. That's why we promote the use of iterative cycles for synchronization. The points of synchronization shall respect the order of concepts in a structuring paradigm of each model. The considered architecture concepts will provide a pivot metamodel that encompasses structuring paradigms from models in each business context.

The mappings between metamodels describe the ways of transformation between each viewpoint and pivot metamodel. This spells out the dependency between elements, properties or relations of the models. We assume that the notion of "mapping" is highly bound to the principles of model to model transformation. There are as many mappings as viewpoint involved in this point of synchronization (minimum two).

The list of possible operations or compromises capitalizes a set of trade-offs that each business context could apply in case of inconsistency between elements. The operations (compromises) shall respect the purposes and rules of each business context. Generic operations can be considered as: add, modify, delete element, rename property, move element, etc). It can also be more developed as application of pattern for example. Redundancy is probably the most popular class of patterns. In case of an application of pattern, it should consider at least one concretization by business context and selected viewpoint.

The illustration, figure 6, is an application of the configuration of synchronization on functional architecture of the case study. The point of synchronization intends to establish a consistency of the composition of functions at system level. This considers a composition as: consideredArchitecture concept, a mapping and, a set of generic operations for the both engineering fields as lists of possible compromises.

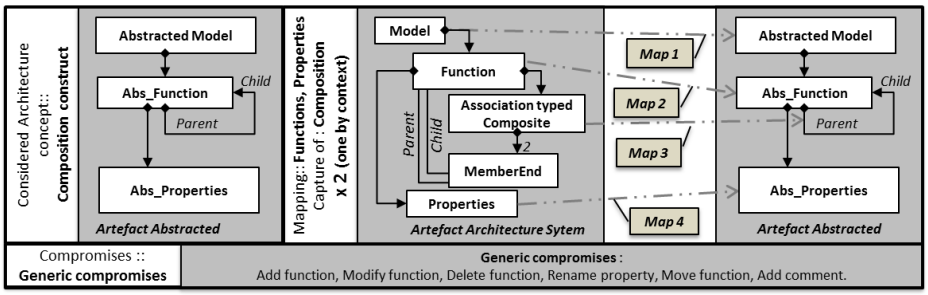

Fig. 6. Fire detection system application on point of synchronization of functional composition

4.3 Application of the synchronization

The target synchronization must satisfy constraints which will frame the methodology (to be applied). It must maintain a separation of concerns between system architecture and safety analysis. We consider synchronization as follows:

$$\textbf{Synchronization} = \textbf{Abstraction} + \textbf{Comparison} + \textbf{Concretization}$$

Abstraction. Here we consider a pragmatic definition of abstraction. It allows to read a source model, select the information carried by this viewpoint and rewrite the information in a target viewpoint (provided a target metamodel is defined). We assume that the abstraction applies to model-to-model transformation. The notion of abstraction highlights important concepts: the information encapsulated in the target viewpoint is a subpart of the information included in the source viewpoint. Abstraction could be generalized by formal definition.

Comparison. It identifies the differences between two abstract viewpoints defined by the same metamodel. Model objects are compared two by two. An algorithm must be developed to order comparisons according to dependencies of metamodel elements. Two types of results can be obtained as outputs of the comparison: a set of consistencies and a set of inconsistencies associated with chosen operation that users decide to apply or not in their own viewpoint.

Concretization. It allows, from an existing source model, to refine it by using a more abstract model. This latter has to provide consistent properties of metamodel used by source artifact.

The figure 7 resumes the relation of Abstraction, Comparison and Concretization with models, viewpoint:

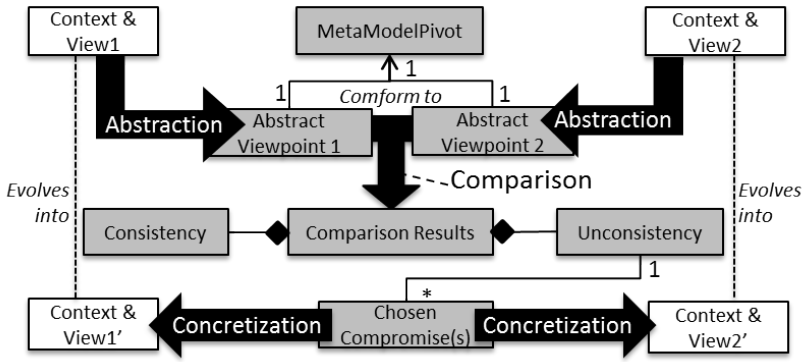

Fig. 7. Application of the synchronization

The application of the defined synchronization is an iterative and collaborative method. The method is a succession of 5 steps: verification of the consistencies from the previous synchronisation point, abstraction of the views, comparison of the abstract views, if one observes at least one inconsistency then concretization of the compromises and evolution of the views, or else validation of the consistencies of views [12].

We present in figure 8 an application of this synchronization method on the case study. The method is applied on two structural descriptions using dedicated modeling formalism for both business contexts.

The source views in figure 8 concerns the assembly of components. We consider that a previous point of synchronization has established consistency in the (hierarchical) description of the system architecture composition. The viewpoints are represented by SysML, Internal block diagram for the system architect and by S2ML [17], [4] for the safety engineer. Both viewpoints capture the internal interconnection of the fire detection system but the level of refinement is different and declaration data have different level of abstraction. The application in

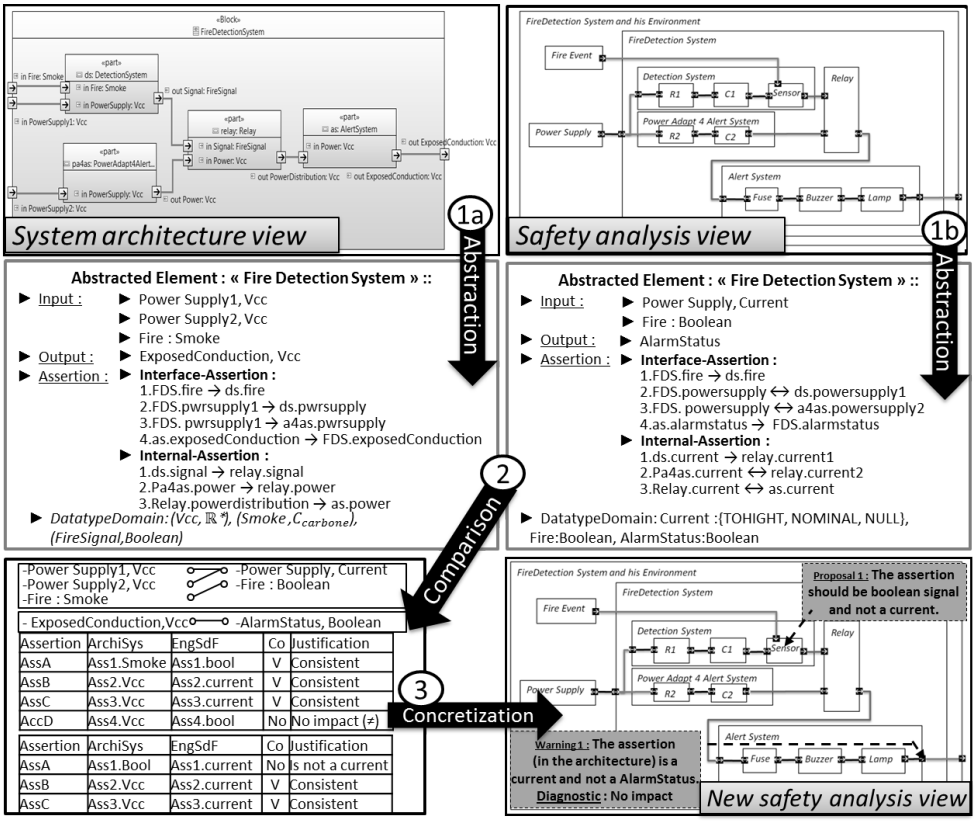

Fig. 8. Fire detection system application of synchronization on physical architecture considering assertion.

figure 8 shows the viewpoints of each engineering field, the abstraction of each viewpoint, the result of comparison and the concretization of operations (compromises) on the safety viewpoint. The result of the concretization step results in proposing two possible operations ("The assertion should be boolean signal and not a current" and "The assertion (in the architecture) is a current and not an AlarmStatus") to Safety engineers.

5 Lessons learned

We present our results from the application of these concepts and propose approach on the fire detection case study. Frequent asked questions are: What is the gain in applying this approach?? How are points of synchronization bound? What are the dependencies between configurations of synchronizations? Which structural concepts should bear the pivot language?

General lessons. A main benefit of the approach is to formalize exchanges. Indeed, a clear formalization permit, in second time, to addresses precise questions during comparison to the engineers. The formalization all along the approach permits to set from the most generic concepts to the most specific, i.e. contexts, processes, activities, etc. It opens dialogues between the activities to identify and resolve possible inconsistencies.

We were interested in the question: "When do you conduct the three activities of synchronization in a system's life cycle?" The specification of contexts were defined upstream of the project. Concepts are generic and are needed for larger reasons than synchronization concerns. The configuration of exchanges was set up at preliminary stages of the project. The later you formalize exchanges the more difficult (and costly) it will be. Synchronizations are applied at the intersection of activities during deployments of the processes.

Focus on the definition of the context. The definitions are close to standards and from some engineering guidelines apply by enterprises. It provides comprehensive understanding of the engineering purposes and interest of neighboring fields. The depth of the environment description depends on the degree of freedom we want to be let to the engineer, e.g. on case study, we define high level description of activities and method for operational analyses by three activities: system environment definition, usecase definition and operation behavior of the system.

Focus on the configuration of exchanges. This is the keystone of the collaboration between MBSE and MBSA approaches. If it is correctly configured, it will enable the generation and management of applications in iterations ways. We observe two main benefits of configuration: synchronization has the best benefits when points of synchronization are very specific with appropriate scheduling.

Difficulties of collaboration rely on exchanges dependencies, according to the scheduling of iterations and architecture considerations. Constraints shall be formalized and respected to avoid missing inconsistencies. We had tried to identify content of libraries mappings, considered Architectures and lists of possible compromises to assist configuration and avoid forget concepts.

The impact of applying such synchronization at early stages brings several advantages. On the fire detection system, we noticed larger need of exchanges on precise architecture concepts at the beginning of the process. Because of their impacts on solutions, it provides huge benefits in terms of project costs and time. The point of synchronization allows the generation of the iterative applications. Relations between configuration and application are shown in figure 9.

The application of this stage on the case study helped us consolidate structuring constructs and operators: composition of the system, assembly of components seems to be quite appropriate to operate on system architecture description. It seems that the scheduling of exchanges is directly linked to dependencies between architecture concepts.

Application / Configuration	Abstraction	Pivot Metamodel	Comparison	Concretization
Considered architecture concept	Determine the target model (pivot metamodel) of transformations	Define metamodel that capture considered architecture concept	Ordering the comparison	
Mapping	Establish abstraction transformations			Establish concretization transformation
List of possible operation			Set operations for identified inconsistency	Apply proposal in viewpoint

Fig. 9. Relation between Configuration concepts and Application concepts

This opens the following question: Which candidate languages are appropriate to represent pivot models? We have not tested a global pivot metamodel (or language) but only local and simple models that capture a part of the structure. This perspective shall take care of model semantics and engineering practices. It also needs more experience and feedback from industry.

Focus on the applications of exchanges. On fire detection system, the application steps show abilities of interaction between several concerns at adapted abstraction levels. We were able to provide the definition and internal descriptions of fire detection system from system architect to safety engineer by iterate three points of synchronization. Step by step, safety engineer has selected then enriched elements and properties. The Probabilistic safety assessment [12] has allowed to identify weakness of architecture. The applications of the synchronization had permit addressing to architect the gap and propose redundancy on specific branch of the system.

6 Conclusions

The methodology has been defined, formalized then applied manually to an industrial case. An experimental framework is under construction. It already contains possibilities of abstraction and concretization, a profile dedicated to context definitions and identifications of needs of exchange. We partially implement a first point of synchronization at operational level. This has allowed to test the feasibility and the efforts required to support the approach. A second implementation of a point of synchronization at architecture design level is underway. It encompasses system modeling with block diagrams (by SysML) and safety analyzes using AltaRica 3.0 [17] within Sophia framework [22] as an experimental test bench.

We address questions on model synchronization: "How can one ensure that the parties involved are speaking about the same system?". We propose a case study to support our argumentation. Three stages have been presented to implement synchronization: context definition, configuration and application of synchronization. This work has allowed to give feedbacks on contributions. Finally

we quickly introduce the state of implement of synchronization and relate it with other contributions.

Acknowledgments. This work is part of a PhD thesis contribution funded by CEA LIST and the DGA (the French Defense Procurement Agency). This thesis is co-supervised by Agnes LANUSSE at CEA LIST (Laboratory of Model Driven Engineering for Embedded systems), and Antoine RAUZY (Supervisor). I would also like to thank APSYS for allowing the dissemination of case studies.

References

1. ISO-42010 Systems and software engineering – Architecture description (Dec 2011)
2. Arnold, P., Rahm, E.: Semantic Enrichment of Ontology Mappings: A Linguistic-Based Approach, pp. 42–55. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
3. ATESST, project: EAST-ADL Domain Model Specification (juin 2010)
4. Batteux, M., Prosvirnova, T., Rauzy, A.: System Structure Modeling Language (S2ML) (Dec 2015), hal-01234903
5. Bozzano, M., Cimatti, A., Griggio, A., Mattarei, C.: Efficient Anytime Techniques for Model-Based Safety Analysis, pp. 603–621. Springer International Publishing (2015)
6. Didonet, D., Fabro, M., Bézivin, J., Jouault, F., Breton, E.: Amw: A generic model weaver. 1ère Journées sur l'Ingènierie Dirigée par les Modèles (IDM05) pp. 105–114 (2005), hal-00448112
7. Fada, M., Nga, N., Choley, J.Y.: Safesyse: A safety analysis integration in systems engineering approach. IEEE Systems Journal pp. 1–12 (April 2016)
8. Feiler, P.H., Gluch, D.P., John, J.H.: The Architecture Analysis & Design Language (AADL). Software Engineering Institute (Fevrier 2006), `http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=7879`
9. Guychard, C., Guerin, S., Koudri, A., Beugnard, A., Dagnat, F.: Conceptual interoperability through models federation. In: Semantic Information Federation Community Workshop. Miami, United States (Oct 2013), hal-00905036
10. KLM: PFD Process flow diagrams (Project standards and specifications) (Feb 2011), `http://kolmetz.com/pdf/ess/PROJECT_STANDARDS_AND_SPECIFICATIONS_process_flow_diagram_Rev1.2.pdf`
11. Legendre, A., Lanusse, A., Rauzy, A.: Directions towards supporting synergies between design and probabilistic safety assessment activities: illustration on a fire detection system embedded in a helicopter. In: PSAM13. Korean Nuclear Society, Séoul, South Korea (Oct 2016), hal-01425309
12. Legendre, A., Lanusse, A., Rauzy, A.: Model synchronisation between architecture system and risk analysis: Which gain, how and why? In: CNRS (ed.) Conference: Congrès Lambda Mu 20 de Maîtrise des Risques et de Sûreté de Fonctionnement. LambdaMu20, IMdR, Saint Malo, France (Oct 2016), hal-01425284
13. Mauborgne, P., Deniaud, S., Levrat, E., Micaëlli, J.P., Bonjour, E., Lamothe, P., Loise, D.: Towards a safe systems engineering. INSIGHT 16, 21–23 (December 2013)
14. OMG: Business Process Model and Notation (BPMN) V2.0 (Janvier 2011)
15. OMG: Systems Modeling Language (OMG SysML) (Sep 2015)
16. OMG: Unified Modeling Language (OMG UML) (March 2015)

17. Prosvirnova, T.: AltaRica 3.0: a Model-Based approach for Safety Analyses. Theses, Ecole Polytechnique (Nov 2014), tel-01119730
18. Rosziati, I., Siow Yen, Y.: Formalization of the data flow diagram rules for consistency check. International Journal of Software Engineering & Applications (IJSEA) 1 (October 2010)
19. SAEAerospace: ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment (dec 1996)
20. SAEAerospace: ARP4754 Certification Considerations for Highly-Integrated Or Complex Aircraft Systems (decembre 2010)
21. Wouters, L., Kaeri, Y., Sugawara, K.: Multi-domain multi-lingual collaborative design. In: Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD). pp. 269–274 (June 2013)
22. Yakymets, N., Perin, M., Lanusse, A.: Model-driven multi-level safety analysis of critical systems. In: SysCon (ed.) Systems Conference, 2015 9th Annual IEEE International. pp. 570–577. IEEE (April 2015)