

Handling Consistency Between Safety and System Models

Tatiana Prosvirnova¹(✉), Estelle Saez¹, Christel Seguin²,
and Pierre Virelizier¹

¹ IRT Saint-Exupéry, 118 Route de Narbonne, 31432 Toulouse, France
tatiana.prosvirnova@irt-saintexupery.com,
estelle.saez@liebherr.com,
pierre.virelizier@safrangroup.com

² ONERA, 2 avenue Edouard Belin, 31055 Toulouse, France
christel.seguin@onera.fr

Abstract. Safety analyses are of paramount importance for the development of embedded systems. In order to perform these analyses, safety engineers use different modeling techniques, such as, for instance, Fault Trees or Reliability Block Diagrams. One of the industrial development process challenges today is to ensure the consistency between safety models and system architectures.

Model Based Safety Analysis (MBSA) is one of the newest modeling methods, which promises to ease the exchange of information between safety engineers and system designers. The aim of this article is to discuss an approach to manage the consistency between MBSA models and system architectures.

Our study is based on the experimentation of the co-design of an RPAS (Remotely Piloted Aircraft System) involving system design and safety teams during the early conception phases of an industrial development process. We simulate the process of exchange between the system design and the safety assessment with the constraint of creating safety models close to system architecture. We identify significant exchange points between these two activities. We also discuss the encountered problems and perspectives on the possibility to ensure the consistency between safety and system models.

Keywords: MBSA · AltaRica · FHA · Development process · System architecture · Safety assessment · RPAS · ARP4761 · ARP4754A

1 Introduction

One of the industrial challenges to improve the current development process is to ensure consistency between safety analyses and system design, which currently requires a very costly effort.

In the industrial practice, the use of models is often limited to a particular engineering domain, e.g. safety, mechanics, thermic. There is no pivot model shared by the different engineering teams and information is most of the time available in textual or informal graphical form. Safety engineers use models (e.g. Fault Trees) to support the validation of system architectures. These models are created from system architecture descriptions provided by system design teams and validated by reviews involving

system and safety engineers. Classical safety models (e.g. Fault Trees) are structurally far from system architecture descriptions. This makes them difficult to maintain, to update and to share with other stakeholders.

We are convinced that the generalization of models use for system architecture description on one hand and the adaptation of Model-Based Safety Analysis approach on the other hand, will bring a solution to this problem.

In this article we study the co-design of the architecture of AIDA (Aircraft Inspection by Drone Assistant) system during early conception phases and following the aeronautical development process standards. We simulate the process of exchange between system designers and safety engineers with the constraint of creating safety models as close as possible to system architecture in order to ease the common understanding of models and speeds up their validation.

The contribution of this article is twofold. First, we identify significant exchange points between the system design and safety assessment activities and show how the use of models contributes to coordinate them. Second, our experiment highlights the possibility of automating the review and exchange processes through the synchronization of architecture and safety models in order to ensure their consistency.

The remainder of this article is organized as follows: Sect. 2 presents the related work, Sect. 3 introduces the experimental framework of this study, Sect. 4 describes the activities of the architecture design team, Sect. 5 is dedicated to safety modeling, Sect. 6 discusses the coordination between safety and system design activities, Sect. 7 concludes this article and gives an overview of the foreseen perspectives.

2 Related Work

Model Based Safety Analysis (MBSA) is one of the newest modeling methods that promises to make easier the exchange of information between safety engineers and system designers. The idea is to write models in high level modeling formalism so to keep them close to the functional and physical architecture of the system under study. Several modeling languages and tools support the MBSA approach, for instance, AltaRica [3, 4, 10], Figaro [5], SAML [7], HiP-HOPS [6], AADL EMV2 [8], SOPHIA [9].

Basically, two approaches to integrate safety models with system architectures can be found in literature. The first one consists in creating a common single model, which describes the system architecture and also encompasses the safety data. To do this, the formalisms used in other system engineering domains are extended. The system model is expressed in a dedicated modeling language; and is annotated with safety data and converted into a low level formalism to perform safety analyses (e.g. Fault Tree). This approach is used in AADL, where the system architecture is described in AADL and is then extended with error model using the EMV2 annex [8]. A similar approach is proposed by the CEA-LIST, where SysML is used to define the system architecture, and SOPHIA for safety modeling and annotation [9].

The second approach consists in using two different domains specific modeling languages: the first one dedicated to the architecture description, the second one to safety analyses. The co-evolution of software architecture and Fault Tree models is studied in [17]. An interesting approach to synchronize system and safety models is proposed in [16].

In this study we focus on the synchronization of system development and safety assessment processes from an industrial perspective.

3 Experimental Framework

3.1 Studied System: The AIDA (Aircraft Inspection by Drone Assistant) System

The studied system is a Remotely Piloted Aircraft System (RPAS). The system is composed of a quadcopter drone, a control computer and a remote control. The mission of this system is to help the pilot to inspect the aircraft before flight. The quadcopter drone can be piloted in automated or manual mode. In manual mode, the pilot guides the inspection of the aircraft by the drone. In automated mode, the drone follows a flight plan and records the video of the inspected zone.

RPAS has been chosen for this study because it is representative of a safety critical embedded system.

3.2 Industrial Development Framework

Our work purpose is to provide methodologies that are applicable to industrial developments. To do so, we position our development activities within the development processes described in the ARP4754A and ARP4761 standards [1, 2] for transport airplanes. They are unquestionable guidelines to show compliance with regulations.

Our experiment implements some of the system development processes that occur during the concept main stage, defined by the ISO 15288 [14] and ISO 24748 [15], during the architecture framing stage. This stage is dedicated to the system preliminary definition. Several architectures can be defined, assessed and compared upon several criteria such as safety, availability, system performance, price, or risk level.

Given the numerous design iterations performed during this stage, the proposed safety assessment process has to remain lightweight.

3.3 Collaborative Experiment Plan

Our study focuses on the co-design activities involving system and safety teams during the early concept stage. In practice, industrial developments require bringing together a large set of skills which are brought to projects by dedicated specialists usually organized in separated teams. That is why, in addition to MBSA specialists, our analysis also involves a system architect and a safety specialist, both of them with an industrial background.

The experiment plan of the co-design (Fig. 1) includes the following activities: Architecture definition, presented in Sect. 4.2; Functional Hazard Assessment, discussed in Sect. 5.2; Safety modeling, described in Sect. 5.3; Safety model review, presented in Sect. 6.1; Safety Analysis, given in Sect. 6.2.

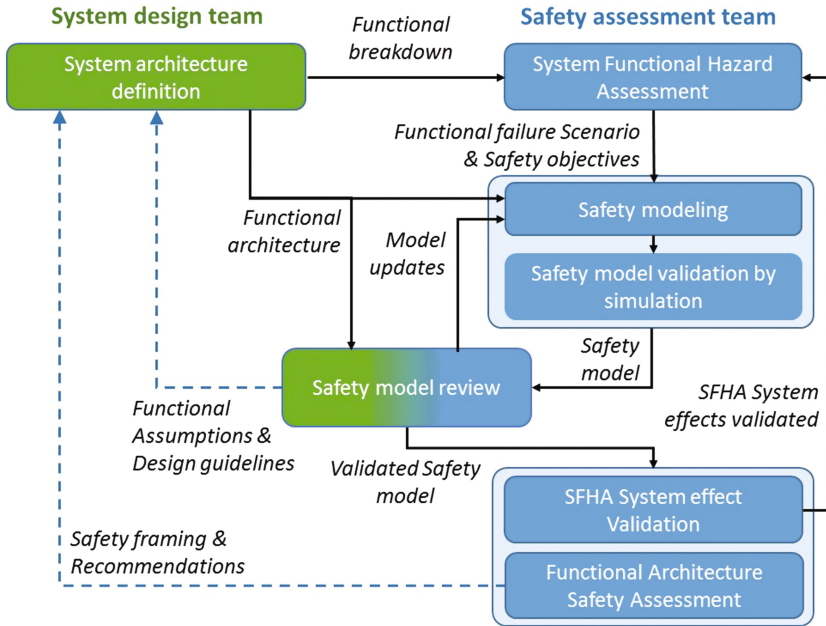


Fig. 1. Co-design of the AIDA system

Table 1. Link between AIDA activities and ARP4761 activities or guidelines

Co-design activities	System development activities from ARP 4754A
System architecture definition	Aircraft Function Development; Allocation of Aircraft Functions to systems; Development of System Architecture (Functional breakdown)
System Functional Hazard Assessment	System Functional Hazard Assessment (SFHA); Functional failure scenario and definition of safety objectives qualitative and quantitative
Co-design activities	Development process guidelines from ARP4761
Safety modeling	Provide a systematic examination of the proposed architecture to determine how failures could cause the Failure Conditions identified by the SFHA
Safety model validation by simulation, and Safety model review	Provide the necessary assurance that all relevant failure conditions have been identified and that all significant combinations of failures which could cause those failure conditions have been considered
SFHA System effects validation	Provide understanding of possible failure modes and effect of failure conditions on the aircraft, crew and occupants
Functional safety assessment	We focus on recommendation directly related with ARP guidelines: FDAL allocation constraints, potential Independence between functions, potential monitors, fail safe criteria and Common failure/faults cause failure

Fundamental interactions, including input and output synchronization between system early development processes and functional hazard safety assessments, described in ARP 4761 [2], are respected in our experiment (Fig. 1). In addition activities in the proposed experiment plan, can be directly related to ARP early development process and guidelines as shown Table 1.

4 Architecture Description

4.1 The Current Practices and Needs

System engineering challenges are numerous and very well detailed by INCOSE [12]. From an industrial perspective, System Engineering promises to solve the problems arising when developing complex systems. But, on the other hand, it is also a costly activity that requires rethinking well-established processes and organization.

Industrials cannot only rely on technical excellence in order to face the new challenges brought by the development of systems with high level of complexity and innovation. System engineering promises to assist and to coordinate development teams, to answer the customer needs and to continuously monitor and mitigate the technical risks.

To do so, system engineers and architects rely on high level abstract representations of the system of interest also named views. Those views are the material that improves the communication with all the stakeholders of the project and carries a common reference vision of the system for the development team. It is the system architect responsibility to insure the views consistency and to define the right piece of information to share with the stakeholders, including the safety analyst. This right amount of information is critical to avoid oversizing architecture description that is costly to manage.

The system team builds its architecture proposal progressively: view by view, and from abstract to detailed systemic levels. System engineers try to secure each incremental conceptual step, by validating the consistency of the whole proposal, by checking the compliance with system key requirements and by seeking feedback from other design teams. In particular, it is of main interest to assess the safety of proposed architectures as soon as possible. Safety assessment provides design safety requirements and recommendations. By taking into account this early feedback, system team expects to make wise design choices and to prevent further heavy reworks.

4.2 Architecture Modeling Activity

The proposed system model is produced by following an industrial method, which proposes to describe the system with views grouped into three main domains: operational, functional and physical.

Our experiment represents the architecture framing stage, and only a limited number of views are available: the system team has chosen to model approximately 80% of the operational architecture and 50% of the functional architecture. Detailed mission scenarios, functional behavior and functional control flow, as well as

functional scenarios are not modelled. The physical architecture is not addressed, which is a recommended practice in order to leave the field of investigation opened.

The main kinds of views used for the safety assessment are the functional breakdown and the functional interaction views. The functional breakdown lists all the system functions, in a three levels hierarchy. The AIDA system contains 6 functions of level 1, 23 functions of level 2 and 59 functions of level 3.

The functional interaction views describe all the functional flows between all the functions and the system stakeholders. Several views represent each systemic level. For instance, Fig. 2 describes the functional flows between the first level functions

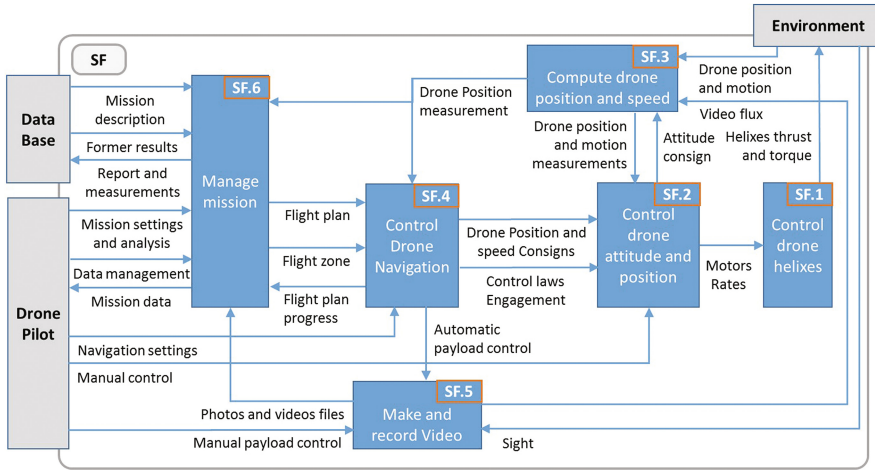


Fig. 2. AIDA upper level functional interactions view

5 Safety Modeling

5.1 The Current Practices and Needs

During early conception phases, safety analyses provide system design teams with a safety frame. This safety frame ensures that safety aspects in architecture design are identified and handled. Safety teams identify the safety critical functions and provide recommendations to ensure the architecture complies with safety objectives. This activity leads, for instance, to the addition of function segregation constraints, failure detection and isolation, recovery functions as well as safety requirements on functional behavior.

Safety assessments rely on a systematic examination of how functional failures cause the failure conditions identified in the System Functional Hazard Assessment (SFHA). The assessment quality requires a good understanding and representativeness of: the system definition, the interfaced systems, the environment, and the operations and operators actions with regards to their impact on safety. Note that handling efficiently design modifications and evolution tracking is a major challenge for large

industrial systems. Eventually, safety analysis validation is part of the development process. During this validation, consistency with system design is usually checked by reviews. That is why safety models need to facilitate these reviews and to contain sufficient level of details and information.

In current practices, Fault Tree Analysis, which provide a graphical representation, are widely used. They fulfill all the needs previously enumerated, but at a heavy cost. For instance, modeling rework workload is an issue, as well as design evolution tracking. The chosen MBSA approach proposes an alternative, closer to the functional architecture representation. Our expectations regarding MBSA is to respond to safety current practices and needs in an economical and modular way. Moreover, MBSA guidelines are about to be introduced and detailed in the next revision of ARP4761 standard. This will allow a larger application of this modeling method in industrial developments, in particular for certification.

5.2 Functional Hazard Assessment Activity

In our experiment the SFHA is performed based on the system functional breakdown, following ARP4761 safety guidelines: functional system failure modes, also called functional failure scenarios, are identified for each function of level 2 in order to determine and classify the effects of the failure conditions on users and environment. Each functional failure scenario details the functional failure conditions effects and repercussions, the operator actions, the detection means, the high level reconfiguration and the allocated criticality. In a second time, functional failure scenarios with the same repercussions are regrouped in order to identify higher level failure conditions and related safety objectives (see Table 2).

Table 2. SFHA Failure Conditions of the AIDA system

Function failure conditions	S/R repercussion immediate effect of failure on drone, operator, people around	Detection means	Classification	FDAL
FC01: Uncontrolled drone, crash in an unauthorized area	Potentially flight in unauthorized zone leading at worst to fatalities	Visually detected by operator	CAT	B fail safe criteria
FC02: Uncontrolled drone, crash in an authorized area	Loss of drone uncontrolled in authorized area. Potentially crash on inspected aircraft	Visually detected by operator	HAZ	C
FC03: Controlled loss of drone	Loss of the capability to control drone position. Increased workload with small reduction of safety margins. End of mission	Visually detected by operator	MAJ	D
FC04: Degradation of drone control	Aircraft inspection fails. No workload increase	None	Min	E

Note: Classification and corresponding FDAL allocation, provided in Table 2, are defined following CS25.1309 [11] transport aircraft approach adapted to RPAS. For instance, catastrophic classification is allocated in case of “Multiple serious or fatal injury”, “Material destruction with major impact on safety”, or “critical reduction of safety margins”.

5.3 Safety Modeling Activity

The entry points of safety modeling activity are system functional architecture provided by the system design team and functional failure scenarios and failure conditions provided by the SFHA activity. This activity consists in two steps: the safety model creation, and the safety model validation by simulation. The result of this activity is the safety model that will be validated by the system design team.

One of the constraints for creating the safety model is to be as close as possible to the functional architecture of the AIDA system, and to represent system reconfigurations. To satisfy these constraints, we use AltaRica Data-Flow modeling language [4] and Cecilia OCAS workbench in order to create convenient graphical models.

The AltaRica model of the AIDA system is made of three parts: the functional architecture view, the reconfigurations managed by the pilot, and the observers on failure conditions and high level functions.

The first part of the model (see Fig. 3) is made by connecting components such that the model topology reflects as much as possible the functional architecture of the AIDA system, given Fig. 2. It is structured in the same way: high level functions SF1, SF2, SF3, SF4, SF5 and SF6 defined in the functional architecture and connections between them are represented in the AltaRica model.

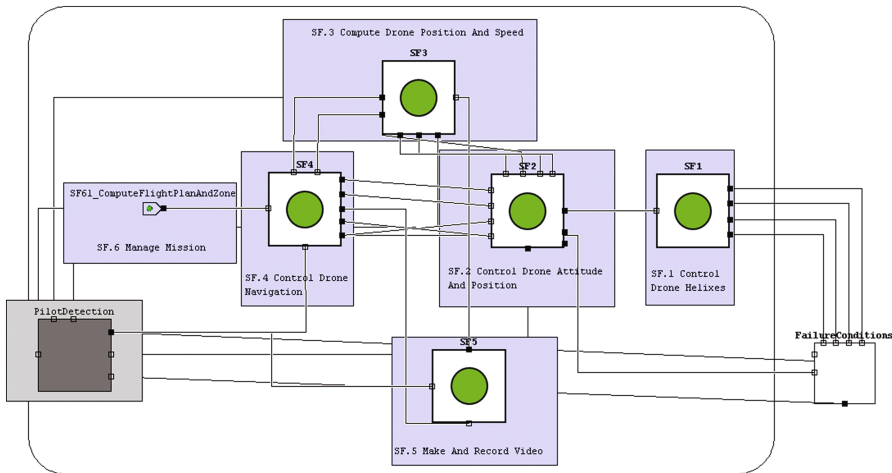


Fig. 3. AltaRica model: functional architecture view

Each basic component models a basic function with interfaces that can propagate or contain failures. The local function behavior is described by a state machine given Fig. 4. The state of the function is represented by a state variable which can take three values: OK, LOST or ERRONEOUS to account with the SFHA failure modes. The transitions between states are caused by stochastic events: `fail_loss` and `fail_error`.

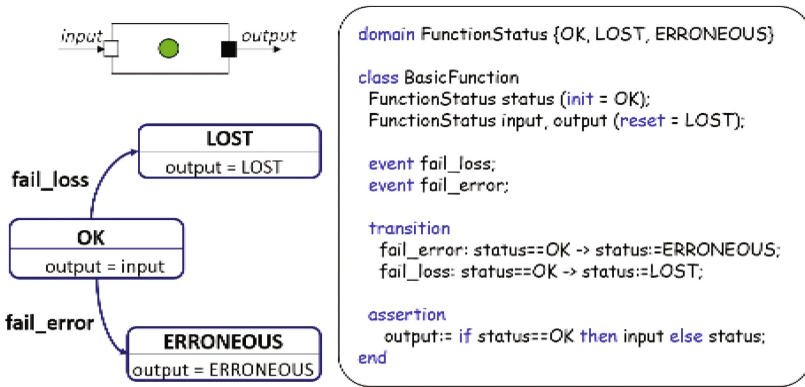


Fig. 4. AltaRica model of a basic function

The second part of the model describes the detection and reconfiguration functions performed by the pilot. For this analysis step human errors are not expected to be considered and consequently we do not introduced failure modes in the model of the pilot.

The third part makes the connection between the failure conditions, high level functions (RPAS functions) and the system functions described in the functional architecture of the AIDA system. This view, given Fig. 5, is closed to a Fault Tree. The top events of the Fault Tree represent the failure conditions FC01, FC02 and FC03 defined in the SFHA (see Table 2). The leaves of the Fault Tree are observers on variables from the functional architecture view. Intermediate events describe failures of high level functions.

The second step of the activity is the validation of the safety model by simulation. It works as follows: sequences of failure events are triggered, their propagations paths to the failure conditions are analyzed and compared by the safety engineer to the SFHA. Functional failure scenarios of the SFHA are represented by sequences of events in the AltaRica model. All of them are simulated. On one hand, it confirms that all the functional failure scenarios have been taken into account in the AltaRica model. On the other hand, it allows the verification of system effects and their severity.

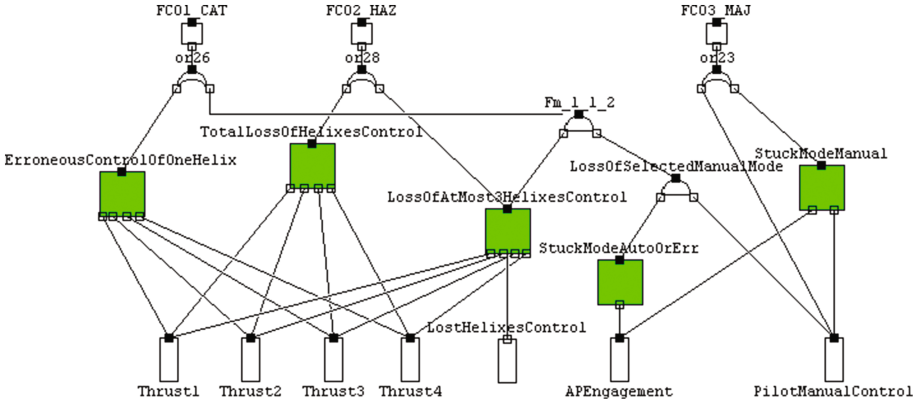


Fig. 5. AltaRica model: link between failure conditions and system functions

6 Model Contribution to the Coordination of System and Safety Activities

The goals of coordination of system and safety activities are to:

- Ensure common understanding and consistency of working hypotheses: validation of the safety model by a system designer and conversely;
- Support design with safety considerations: identification of acceptable design choices from the safety analysis.

6.1 Safety Model Review Activity

The safety model review activity is an important exchange point between system designer and safety assessment teams. The entry points of this activity are the system functional architecture and the corresponding safety model. This activity is the safety model review by the system designer. It consists in reviewing the safety model with the support of the safety engineer and in comparing it to the functional architecture. The results of this activity are multiple. First, it provides the safety model updates, i.e. remarks to take into account in order to better comply with the functional architecture of the system. The process is iterative (see Fig. 1): the safety model is updated according to the remarks of the design team, it is then validated by simulation and reviewed again by the system designer. Second, it provides missing elements and assumptions on functional and stakeholders behavior to be integrated into the next version of system architecture. Finally, at the end of iterations it provides a validated safety model to be assessed by the safety team.

AIDA system safety model review. In order to perform efficiently the safety model review, we establish a set of comparison criteria. For each function of the functional architecture the following criteria are verified:

- Representation of the function in the safety model,
- Representation of the function input and output ports in the safety model,
- The function internal failure events and their propagations to the outputs and to the other functions.

Our analysis justifies the differences found between models. The safety model has the same structure as the functional architecture. It contains: the 6 level 1 functions, 19 out of 23 level 2 functions and only 3 out of 59 level 3 functions. Only two levels of functional breakdown are considered (the same levels as in the SFHA). This abstraction level is sufficient to perform preliminary analysis of the proposed architecture. In addition, it is possible to refine the model in the next iterations. Few functions of level 2 are not represented in the safety model. Basically, functions with failure modes leading only to failure conditions of Minor severity and/or having no impact on other functions (e.g. functions used only on ground with no dependency on the flight). Three functions of level 3 are introduced to clarify dependencies between functions of level 2; all of them manage RPAS control mode.

However, to create the safety mode, assumptions are done about functional flow dependencies via connections (not provided by the functional architecture definition). The safety modeling of a function should characterize the quality of each function output depending on the quality of the functional input and the function state. Based on our experiment, functions with multiple inputs and outputs raises the following questions:

- *How to combine several inputs in function to calculate the output?* A certain number of assumptions are made by the safety assessment team to create the model. The minimal quality is propagated when several inputs are used to calculate the output.
- *How are calculated the function outputs when there are several?* Functions of level 3 are analyzed to find out the possible paths for failure propagation. In some cases it is possible to assemble together several outputs to simplify the model.
- *How are calculated the flows in case of loops between functions?* Temporal loops are not specified in the functional architecture. However, causality hypotheses are needed to analyze the failure propagation. They should be discussed with the system designer.

Compared to the system model, the safety model also contains some additional components and connections. Main information to build the pilot safety model is taken from the SFHA. For example, the safety model of the pilot represents the corrective actions following the visual detection of system failures. This detection requires some observations modeled by additional connections with system functions. The safety model also contains the view on the failure conditions and high level functions as explained earlier in Sect. 5.3.

The review activity produces a document which lists all the differences between the models, their justifications and actions to be done by safety and system design teams. The safety model may need to be updated in order to take into account the remarks of system designers. This process is iterative: the safety model is updated by the safety

team, then it is reviewed by the system design team and so on, until the validation of the safety model.

Benefits of using models for the safety model review. Our experiment shows that the use of new modeling techniques for both system architecture description and safety assessment makes the review activity easier. First, it improves a common understanding. As we have seen earlier, our models are structured in the same way, the functions have the same names. To some extent both disciplines are manipulating the same objects. Second, it allows the definition of the stopping criterion for the review activity. The review is considered finished when all the functions from the functional architecture have been analyzed. Finally, it enables to establish a clear set of criteria to compare and to validate the models.

The use of MBSA models also speeds up the safety model updates. Indeed, differences between models are localized to functions. So, the safety model updates are also very local. Also, the next model review only focuses on the modified part which increases the efficiency.

Benefits of the safety model review activity. Doing the review of the safety model at this stage of the development process brings the following gains:

- It enables to identify some missing elements in the system functional architecture. In the proposed exercise, we identify the list of functions used and not used in the automated control mode and in the manual control mode and the list of functions used and not used in flight or on ground, to be able to take them into account in the analysis of failure conditions. They will be integrated into the next version of the functional architecture.
- It highlights ways to improve the functional architecture. For example, the modeling of failure propagations from the payload to the control laws function, outlines a possible design drawback.
- Several target assumptions on functional and stakeholders behavior are produced, for example, formalization of pilot actions in case of visual detection of the AIDA system loss of control.
- The review activity also allows to identify the elements of the functional architecture having critical impact on system safety in order to take them into account efficiently in the next iteration.
- Finally, it enables the safety engineers to have a feedback and an early validation of the model very soon in the development process.

Towards automation of model review. On the basis of the established comparison criteria presented earlier, it is possible to define an algorithm to automatically compare the models structure. The main constraint is to use in the AltaRica model with the same names (or prefix) of functions and input/output ports as in the functional architecture.

Let denote by $FA = (F, C)$ one hierarchical level of system functional architecture, where F is a set of functions and C is a set of connections. A function $f \in F$ is defined by its name and two sets: the set of input ports $I(f)$ and the set of output ports $O(f)$. Each input or output port is defined by its name. A connection $c \in C$ between two functions f and g is a relation between two ports of functions f and g . Let denote it by $c(i(f), o(g))$.

Let denote by $ASM = (F', C')$ one hierarchical level of the abstracted safety model, where F' and C' are sets of functions and connections represented in the safety model.

In the AltaRica model, class instances (also called nodes) are abstracted to functions, flow variables are abstracted to input or output ports, and assertions representing the failure propagation paths between functions are abstracted to connections of the form $c'(i(f'), o(g')) \in C'$.

For each hierarchical level of system functional architecture the algorithm starts with the comparison of function sets F and F' . For each function f from F , it searches if the function with the same name (or prefix) exists in F' . If it does not exist, it reports that the function f is missing in the safety model. Otherwise, the algorithm compares the input and output ports of the functions f and f' and reports if there are missing or additional ports in the abstracted safety function f' . This comparison is done by port names. Additional functions of the safety model are also reported.

The next step of the algorithm compares the sets of connections C and C' . For each connection $c(i(f), o(g))$ from C , it searches if there is a corresponding connection $c'(i(f'), o(g'))$ in C' , i.e. a connection which makes a relation between two ports with the same names. If it does not exist it reports that the connection c is missing in the safety model. It also reports if there are additional connections in the safety model.

At the end, the algorithm produces a report on missing and additional elements which would help to perform the review activity.

In fact, to be able to compare structures of models efficiently, it is of interest, to first abstract them into a pivot language. The candidate of such a pivot modeling language could be S2ML (System Structure Modeling Language) [13], which provides a small but sufficient set of constructs to structure models.

6.2 Safety Analysis Activity

The entry point of the Safety Analysis activity is the safety model validated by the system designer. The activity is performed in two steps. First, functional failure scenarios described in the SFHA are simulated in order to check their resulting failure conditions. The safety model allows to validate SFHA system effects or correct them in case of discrepancy. It also allows to identify safety impacts of functional dependences. As an example consider the following functional failure scenario: “*Loss of the capability to make and record video*”, which is related to the function SF5: Make and Record Video. The system safety effect in the SFHA is minor (FC04 (MIN)), but in the chosen design, video flux is used to calculate the speed in automated pilot mode. The simulation allows to detect that the “*Loss of the capability to make and record video*” leads to a MAJOR effect because it forces the pilot to switch to manual mode with additional safety constraints.

The second step of the activity is the functional architecture safety assessment. It consists in generating Minimal Cut Sets (MCS) leading to the failure conditions of severity CAT and HAZ. The MCS of order 1 and 2 are analyzed in order to provide safety framework and recommendations to the design team. An example is given in Table 3 for an order one MCS.

Table 3. Example of safety recommendations

MCS leading to FC01 (CAT)	Regulatory safety requirements	Recommendations
SF1. ControlHelix4.fail_error <i>“Erroneous control of helix 4”</i>	Single functional failure leading to FC01 => FDALA	No hardware single failure
	Two independent functions leading to FC01 => FDAL A + FDAL C, or FDAL B + FDAL B	System robust to erroneous control of one helix, or new monitoring function

The safety model provides the expected feedback to the system designer. The safety recommendations will be integrated to the system design and its justification.

One of the main contributions of doing safety analysis at this stage of development process is also to avoid “over-design“. The idea is to start with a minimal architecture and then to add necessary monitorings, reconfigurations and redundancies, according to the recommendations of the safety assessment team. In that way, all the safety related mechanisms are clearly justified and “over-design” driven by previous experiences or habits, can be avoided. This methodology relies on the capacity of system design and safety teams to iterate quickly and efficiently through models.

7 Conclusion and Perspectives

According to the experiment plan given Fig. 1, we have identified four important exchange points between system design and safety assessment teams. The first one corresponds to the providing of system functional breakdown by the system design team to the safety assessment team in order to perform Functional Hazard Assessment. The second exchange point consists in providing the functional architecture by the system design team to the safety assessment team. The safety assessment team uses this information to create the corresponding safety model. It is an entry point of the safety modeling activity. The third exchange point is the safety model review activity. It produces useful feedback for both safety and system design teams: safety model updates and functional assumptions and design guidelines. The fourth exchange point is the safety recommendations provided to the system design team after the model analysis.

Our experiment has shown that the use of new modeling techniques from an industrial perspective for safety and system design supports efficiently exchange between safety and design teams in early design phases because the perimeter of the exchange data is clearly identified. It makes the safety model review easier, improves the common understanding and speeds up the safety model update. It is worth to note that used safety and functional models have both commonalities and differences. Commonalities speed up understanding exchange between safety and design teams that is a must in early design phases. Differences are also necessary to avoid the overload of system model, whilst keeping an efficient safety model that provides useful feedback.

Our experiment has proved that the functional architecture view provided by the system team is necessary to frame most of the safety model. It has also shown that a

view describing the system functional modes would greatly improve the system understanding by the safety engineer.

Moreover, our experiment highlights the possibility for automating the review and exchange process. Further work will consider the synchronization of architecture and safety models to ensure the consistency between them.

The experiment has been carried out for one architecture version. It is also interesting to study how the proposed process will support an incremental design.

Finally, we intend to continue our work and to propose another process for the safety assessment of a physical architecture based on the updated and extended architecture views, supported by a safety model. One of our main challenges will be to increase the efficiency of this new process by benefitting from the models and results that we have obtained at functional level.

This study is based on aerospace development and safety assessment processes. As a future work, it would be also interesting to study a mapping of the identified synchronization points to other engineering domains.

Acknowledgements. The authors thank all people and industrial partners involved in the MOISE project. This work is supported by the French Research Agency (ANR) and by the industrial partners of IRT Saint-Exupéry Scientific Cooperation Foundation (FCS).

References

1. SAE, ARP4754A: Guidelines for development of civil aircraft and systems (2010)
2. SAE, ARP4761: Guidelines and methods for conducting the safety assessment process on civil airborne system and equipment (1996)
3. Point, G., Rauzy, A.: AltaRica: constraint automata as a description language. *J. Européen des Systèmes Automatisés* **33**(8–9), 1033–1052 (1999)
4. Rauzy, A.: Modes automata and their compilation into fault trees. *Reliab. Eng. Syst. Saf.* **78**, 1–12 (2002)
5. Bouissou, M., Bouhadana, H., Bannelier, M., Villatte, N.: Knowledge modelling and reliability processing: presentation of the FIGARO language and associated tools. In: *Proceedings of SAFECOMP 1991, Trondheim, Norway* (1991)
6. Adachi, M., Papadopoulos, Y., Sharvia, S., Parker, D., Tohdo, T.: An approach to optimization of fault tolerant architectures using HiP-HOPS. *Softw. Pract. Exper.* **41**(11), 1303–1327 (2011)
7. Güdemann, M., Ortmeier, F.: A framework for qualitative and quantitative model-based safety analysis. In: *Proceedings of HASE* (2010)
8. Delange, J., Feiler, P., Gluch, D., Hudak, J.: *AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment*. Carnegie Mellon University, Pittsburgh (2014)
9. Cancila, D., Terrier, F., Belmonte, F., Dubois, H., Espinoza, H., Gerard, S., Cuccuru, A.: SOPHIA: a modeling language for model-based safety engineering. In: *MoDELS 2009 ACES-MB Workshop Proceedings, Denver, CO, USA* (2009)
10. Prosvirnova, T., Batteux, M., Brameret, P.-A., Cherfi, A., Friedlhuber, T., Roussel, J.-M., Rauzy, A.: The AltaRica 3.0 project for model-based safety assessment. In: *Proceedings of IFAC Workshop on Dependable Control of Discrete Systems, York (Great Britain)* (2013)

11. EASA, CS-25: Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes, Amendment 12 (2012)
12. INCOSE website. <http://www.incose.org/AboutSE/WhatIsSE>
13. Batteux, M., Prosvirnova, T., Rauzy, A.: System Structure Modeling Language (S2ML) specification (2015)
14. ISO 15288: Systems Engineering – System Life-Cycle Processes (2015)
15. ISO 24748-2: Systems and software engineering - Life cycle management - Part 2: Guide to the application of ISO/IEC 15288 (System life cycle processes) (2011)
16. Zeller, M., Höfig, K.: INSiDER: incorporation of system and safety analysis models using a dedicated reference mode. In: Proceedings of RAMS, Tucson, AZ, pp. 1–6 (2016)
17. Getir, S., Tichy, M., van Horn, A., Grunske, L.: Co-evolution of software architecture and fault tree models: an explorative case study on a pick and place factory automation system. In: Proceedings of the 5th International Workshop on Non-functional Properties in Modeling, Miami, USA, 29 September 2013

Model-Based Safety and Assessment

5th International Symposium, IMBSA 2017, Trento, Italy,

September 11-13, 2017, Proceedings

Bozzano, M.; Papadopoulos, Y. (Eds.)

2017, X, 273 p. 109 illus., Softcover

ISBN: 978-3-319-64118-8