Directions towards supporting synergies between Design and Probabilistic Safety Assessment Activities: illustration on a Fire detection system embedded in a helicopter

Anthony Legendre, Agnès Lanusse, Antoine Rauzy

Directions towards supporting synergies between Design and Probabilistic Safety Assessment Activities: illustration on a Fire detection system embedded in a helicopter

A. LEGENDRE [1], A. LANUSSE [1], A. RAUZY[2]

[1] *CEA, LIST, Model driven engineering Laboratory for Embedded Systems, Gif sur Yvette, 91191,* anthony.legendre@cea.fr, agnes.lanusse@cea.fr
[2] *NTNU, S. P. Andersens veg 5, 7491 Trondheim, Norvège*, antoine.rauzy@ntnu.no

The complexity of modern critical systems is growing rapidly while the industry is submitted to more and more pressure for reducing costs and time-to-market. Traditional development methods "in disciplinary silos" used to design and analyze such complex systems are reaching their limits. RAMS engineers face more and more difficulties to satisfy demands of reliability evaluation especially at early stages of system design. In this context we offer to take advantage of Model-Driven Engineering (MDE) approaches to reduce construction time of reliability models and improve their consistency with system models. Model-Driven Engineering is a promising approach used to develop and analyze complex systems from different domains. In this paper, we exploit MDE to support PSA analysis and illustrate the approach on a case study from avionics industry. This experimentation has enabled the suggestion of a seamless methodology to support iterative Probabilistic Safety Analysis, thus improving the cooperation with system designers.

I. INTRODUCTION

The solicitation of engineers for RAMS[1] assessment of new critical systems or functions is getting more and more important. The pressure to provide fast returns on estimations of system reliability at early design stages is getting more and more intense. At the same time the complexity of these systems is increasing drastically. To establish these estimations, RAMS engineers face, in addition, the difficulty of gathering reliable and accurate probabilistic data on system components and the context of their utilization. This paper presents parts of a PhD Work. It concerns suggestions for a methodology of synchronization and the associated framework, which contains several types of tooling to support exchanges between system architecture design and the evaluation of dependability processes. It also has the effect of enriching the manipulated models. The suggested approach aims at improving the consistency between models and makes their elaboration easier and faster. We are interested here in the practical application of possible exchanges between system engineers (in charge of designing and managing architecture models according to the viewpoints of the several domains at stakes) and RAMS engineers. The approach is illustrated by an avionics case study: an embedded fire detection and firefighting system onboard a combat helicopter. The study is focused on PSA analyses.

In the next section we present the engineering context of the study and precise the characteristics of the stakeholders involved (system designers and RAMS engineers), their skills, the roles' expectations and the work products. Then, in section III, we introduce the case study and present the related safety and reliability requirements. In section IV we describe the used methodology and illustrate it with the case study. The methodology is enforced iteratively to illustrate how recommendations from RAMS experts can be taken into account within architecture and reanalyzed in a short loop. The results are obtained using advanced RAMS tooling based on AltaRica formalism. Reliability Information is stored using OpenPSA[1] formalism, which makes interoperability with other tools possible. Finally, in a discussion section (V), we make a brief return of experience and present perspectives for going further.

II. SYSTEM ENGINEERING CONTEXT

System Engineering [2] promoted by INCOSE[2] is an interdisciplinary approach and means to enable the realization of successful systems. It allows one to elucidate major interplays between various concerns in complex systems. In particular by studying the interactions between different views of a system and the businesses of the stakeholders. It focuses on defining customers' needs and the functionality required early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation.
It integrates all the disciplines and specialty groups into a team effort, thus forming a structured development process that proceeds from concept through production to operation.

[1] RAMS : Reliability, Availability, Maintainability and Safety field.
[2] INCOSE The International Council on Systems Engineering

One major benefit expected from adopting a holistic system design is to enable an early tradeoff elicitation thanks to better dialogues between experts from different fields. However, in order to make it technically possible, there is a need for providing guidelines and, if possible, supporting tools to help. Our approach is based on this vision. In our context we focus on two expertise domains: system architecture design, and system RAMS analysis. We want to provide a support for dialogue and means for easier model exchange and/or synchronization between them. Before entering the presentation of our suggestions and for a better understanding of the problem, it is necessary to precise the roles of such experts in the industry.

II.A. The roles of a System Architect

System architecture is a subfield of system engineering. It issues the conceptual model that defines the structure, behavior and more views of a system. The architect is in charge of the design of the architecture part, he has to deal with the major part of expertise specialized in specific technology to provide consolidated versions of the architecture and refine it, according to the company's development process and feedbacks from experts of specific domains. There is a set of standards to define the general process (EIA-632 [3], IEEE-1220, ISO-15288 [4]), methods (ACCORD [5], ARCADIA [6] and functional analysis formalisms), and technical approaches (MDE, MBSE) dedicated to the practice of system architecture.

II.B. The roles of a RAMS Engineer

RAMS analysis is also a subfield of system engineering. It builds the conceptual model that allows one to define all performance concepts (Reliability, availability, maintainability and safety) and issues regarding evaluation data, safety requirements and solution recommendations. The RAMS analyst is in charge of assessing the system architecture, considering both functional and dysfunctional behaviors according to Safety concepts. The applications of Safety analysis are various and can be used during all of a system's life cycle. In this article we will focus on the applications of Reliability and Safety analysis at the early stages of the development cycle.

There are two major approaches to perform Safety analysis: the classical approach (which uses basic methods, tables and documents) and the MBSA (Model Based Safety Assessment).

In the context of this work, we consider several MBSA approaches:

- Method of safe architecture design by construction: offer ways to design system architecture by directly taking into account Safety performances within design activities.
- Safety Co-modeling: offer ways to model safety concepts and handle analysis by several experts together on the same model using different views.
- Safety analysis from Architecture model: manage Safety analysis by adding concepts and properties on the architecture model to drive Safety analysis. The technology used can differ according to different communities: use of profiles for SysML [7], Error annex for AADL [8] or EAST-ADL [9], [10]. Some researches related to this approach have resulted in supporting frameworks: MEDISIS method [11] by PRISME , SOPHIA tool [12], [13] by CEA LIST or SafeSysE [14] by the CentraleSupelec French school and others [15].
- Safety modeling: offer modeling languages dedicated to the representation of safety architecture and dysfunctional behaviors to assess the system in term of safety performance. Some researches are related to this approach: AltaRica 3.0 [16] [17] by AltaRica Association, Hip-Hops [18] by the University of Hull and others.

We have selected Sophia tools suite and AltaRica formalisms to support this experimentation. Sophia is integrated within Papyrus UML/SysML modeler. It uses the concept of profile to annotate the system architecture model with safety concepts,which enables one to lead several safety analyses. It provides various MBSA services, including several FTA, FMEA, hazard analysis, requirement engineering, and transformations facilities towards AltaRica formalism.

III. PRESENTATION OF THE CASE STUDY

In this section we introduce the case study and the different points of view considered (system design and RAMS analysis). Since we are considering a case study in avionics, the approach described here complies with the following standards: DO 178 C [19] standard for the system design and software development, and ARP 4761 [20], ARP4754 [21] for the safety assessment concerns.

III.A. Fire Detection System embedded in Helicopter

The studied system is an embedded fire detection system onboard a helicopter. This system consists in four interconnected devices: a set of sensors, an alert device, a power supply and a fire-fighting equipment.

The system's missions are to detect the fire event 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.

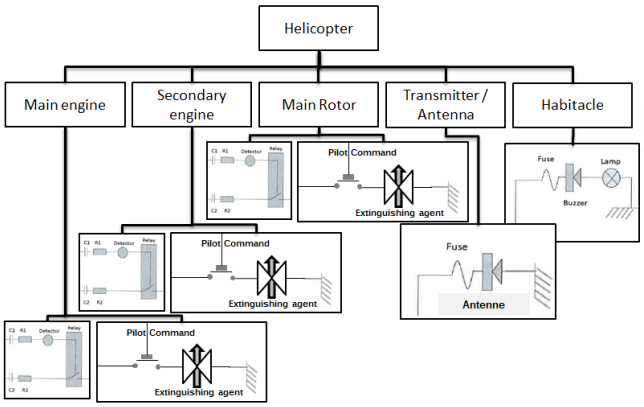

Figure 1 Decomposition of the fire detection system

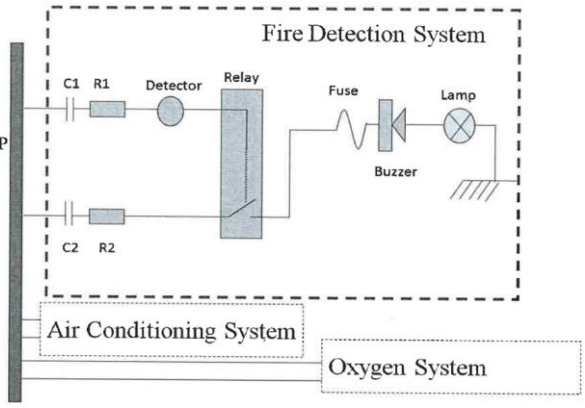

Figure 2 Assembly of the fire detection system

According to helicopter program, the fire detection system will be used in the following environmental conditions:

- Ambient temperature: 45°C
- Electric warning: 15°C
- Environment: helicopter

III.B. Description of the System Architecture

To be able to represent correctly a system architecture, we need to consider an appropriate language that provides a formalism expressive enough to include all representations needed by a system architect. The language has to provide concepts, generic elements (usecase, function, components, functional behavior …) and relationships between them (hierarchical relationship, dataflow relationship, allocation relation …) at several stages of the dedicated process and at several levels of abstraction.

Indeed, we define a set of assumptions to enable the choice of a modeling language dedicated to system architects. The chosen language and tools used have to respect these assumptions. In our context we have chosen SysML language to represent the system's architecture. Other languages could also have been appropriate as AADL, EAST-ADL, etc.

We should note here that SysML is not a method but a graphic modeling language dedicated to represent a system, its requirements and traceability information. For the study we have used the Papyrus SysML modeling tool to describe the system architecture. Papyrus is an open source tool developed in the laboratory CEA LIST and is an Eclipse project.

The system design is represented with several points of view (views): architecture hierarchy, structure and interactions between components, behaviors, and more …

The Figure 3 below presents two diagrams: a definition block diagram and an internal block diagram (provided by SysML). They describe the main system components of the "Fire detection System" and their possible interactions (exchanges between components).

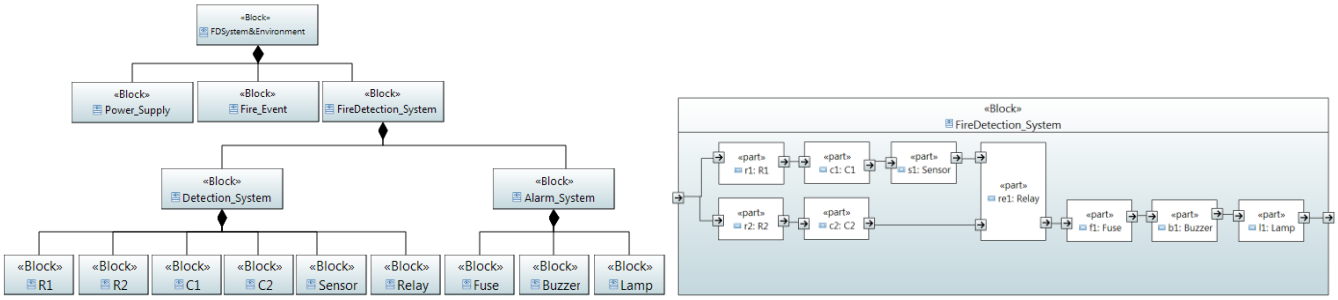

Figure 3 : Definition Block Diagram and Internal Block Diagram of the « Fire detection System » in SysML

The block "Fire detection System" consists in two interconnected devices and two system inputs:

- Detection system: A set of sensors to calculate the heat and the level of carbon monoxide saturation,
- Alert system: An alert device to inform on-board crew and ground rescue team
- Power supply: 24Vcc is an input of the system
- Fire Event: Event occurs from external actors or natural environments.

The system architecture provides an important set of data that will benefit the RAMS Engineer: description of the physical system's architecture and description of the components.

III.C. PSA requirements

The scope of the PSA is to assess an architecture according to a RAMS requirement list. The methodology to lead PSA must respect a Safety Program Plan.

In our context, the RAMS Engineer has to assess the fire detection system to qualify if the requirements specified by the helicopter's program are met with the current design. The system should be tested 1000 hours (MT0) and the mean flight time duration should be 1h (FT0). The concerned system is an unrepairable system; therefore we only consider the Reliability and Safety properties.
Remark: the tool or languages paradigms used here present a huge advantage for the calculation of availability or maintainability on systems because it becomes impossible to calculate them manually for complex systems.

To begin a PSA, a set of input data is required. We will make the assumption that the RAMS Engineer gets:

- The organic system architecture built by the System Architect using SysML modeling.
- The Helicopter FHA and System FHAs (Functional Hazard Analysis) done by RAMS Engineer during the early phases of the safety program plan and the resulting list of requirements.
- The description of the system's environment done by System Architect or RAMS Engineer (depending on the industrial business context).

When PSA starts, previous Safety analyses have already been carried out on the helicopter: they provide many results, including RAMS requirements useful to conduct the analysis:

- Requirements coming from the Aircraft/Helicopter FHA.
    - The Fire detection system is DAL B,
        - The system shall have an MTBF equal to or larger than 10 000 Failure.Hour.
- Functional Safety Requirements coming from the system FHA.
    - FC1: Uncontrolled fire 1.10^{-9} /FH
        - Safety requirement: Uncontrolled fire event shall occur less than 10^{-5} /FH
    - FC2: Loss of fire protection 1.10^{-5} /FH
        - Safety requirement: The Detected Loss of fire protection shall occur less than 10^{-6} /FH
          Remark: there is no requirement for "Undetected Loss of fire protection" but we will consider it later as unexpected event.
    - FC3: Uncommanded fire protection 1.10^{-4} /FH
        - Safety requirement: Uncommanded alarm event shall occur less than 10^{-4} /FH
- Description of the system's environment
    - External failures: EF_01 Loss of bar 109X 1.10^{-5} /h
    - External events: EE_01 Fire event 1.10^{-5} /h

The requirements specified by the helicopter's program are summarized in the following table:

Table 1: List of requirements concerned by the fire detection system

Requirement's sources	**Requirement's type**	**Title** (link to Fire detection system Req)	**Objective values** (/Failure.Hour)
Helicopter FHA, DAL B	Reliability	REQ_REL01 : Loss of the system - Lambda	10^{-4} /FH
System FHA, FC1	Safety	REQ_SAF01 : Uncontrolled fire	10^{-9} /FH
System FHA, FC2	Safety	REQ_SAF02 : Detected Loss of fire protection	10^{-6} /FH
System FHA, FC3	Safety	REQ_SAF03 : Uncommanded alarm	10^{-4} /FH

Requirements are allocated to each system area of the system. The fire detection system has to respect the four requirements in three areas. Therefore we have a total of 12 requirements. To simplify the case study, in this paper, we will only consider the system in the main engine (4 requirements).

IV. APPLYING INCREMENTAL PSA ON THE CASE STUDY

Probabilistic safety assessment is aimed at determining which undesired scenarios can occur, their likelihood and the severity of related consequences. In addition it can produce indirect information such as the importance of individual risk contributors.
To perform such analysis, RAMS engineers must collect a huge quantity of information and gather them into a RAMS model that can be submitted to specific RAMS tools. However collecting such data, putting them into databases and insuring their accuracy and consistency is very time-consuming and can be error-prone. If shorter response time is given to RAMS experts to issue recommendations on designs, while systems are getting more and more complex, it becomes an important issue to provide methodological and tooling support to assist them.
It is the motivation for the work presented in this paper, even if it copes with only a part of this big challenge. The proposal described here deals with two aspects of the problem:

- Provide guidance and tooling support to build safety analysis models in an incremental way;

- Provide means to insure consistency between system models, safety analysis models and specific formats of dedicated PSA tools.

The expected benefit of the approach is to provide means to support shorter iteration loops between design and assessment activities in order to facilitate more co-operative design (i.e. safety and reliability recommendations can have early impact on design decisions). The technical means chosen intend to insure consistency between the models used by experts from different domains (system engineers and RAMS engineers).

In the following we present the methodology adopted to perform RAMS analysis on the helicopter case study. It relies on a model-based approach and model-driven engineering techniques to manipulate models. System models are directly exploited to build the basis of a safety model. Then this safety model is progressively enriched along different steps of classical RAMS analysis. The benefit is that information is collected and gathered progressively in the model which is consistent with the system model. In Figure 4 Incremental RAMS analysis process overview) we briefly present the different steps of a progressive safety model building process. Finally the model is transformed to AltaRica Language which is supported by RAMS tools embedded within the Open AltaRica framework. The interest of using the AltaRica formal framework is that it can support the analysis of complex systems thanks to powerful formalism and techniques. Moreover it relies on the OpenPsa format to store information in order to support tools' interoperability. One can notice that similar transformations could also be achieved to target other formalisms and tools.

1. In a first stage, the RAMS engineer builds the basis of its safety model, which consists in the architecture of its components (i.e. components and their interactions). This architecture can be directly imported from the actual system design model.
2. Then it starts entering probabilistic data (Failure modes and related failure models) for each type of components. The model is enriched and is now a basic failure model that can be used to perform pre-FMEA analysis.
3. In a next step, failure behaviors of components and failure propagation specification are added to the model. This time the model is ready to support qualitative and quantitative analyses (FTA and PSA).
4. PSA analysis is performed after the safety model has been transformed to AltaRica to permit formal manipulation and computation using OpenAltaRica toolset.

Finally results from analysis can be transmitted to system engineers and a dialogue can be engaged in order to decide possible evolutions of the model in order to comply with safety and other RAMS performance criteria. In the case study, implementation of redundancy has been decided so the architecture has been modified in system model, and then propagated in safety model. Previous safety data entered has been reused.

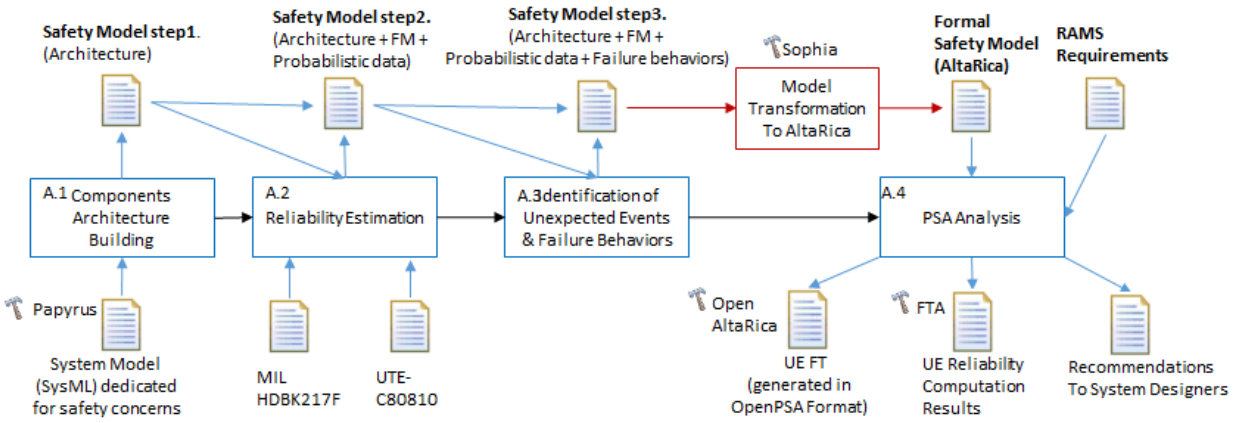

Figure 4 Incremental RAMS analysis process overview

In the next sections we describe this process in details. A first iteration of PSA analysis is achieved, resulting in the identification of the unsatisfied safety requirements. Then the system is modified and a second PSA iteration can be performed. This time, a large part of information is already present and the task of the RAMS engineer is facilitated.

IV.A. First PSA Analysis

IV.A.1. Building a Model of Safety Engineer from Designer and a Model of System Architecture to Safety Engineer

This first stage of the process deals with the building of the architecture of the safety model. This task is one of the most time-consuming in the industry, where people are often given a list of components and a document describing the system's architecture in textual format with possibly some figures to illustrate the explanations. RAMS engineers must then spend a lot of time trying to understand this documentation and they may make mistakes or misinterpretation during this process. Generally the collected information is then gathered into huge spreadsheets, which impairs their readability.

The resulting model is thus difficult to read and understand, not necessarily fully aligned with the initial model of system design and possibly errored. In our case, the initial system model is directly used to automatically build the architecture of the system's components. The automatic population of model architecture is a first stage towards providing consistency and synchronization between models used to tackle different concerns.
We use the Sophia framework to do so but other academic tools (SafeSysE, Medisis,…) could be used too. Noticeably, this approach is being adopted by several tool providers or safety consulting companies, and new products are becoming available (Safety Architect, Enterprise Architect, SIMFIA).
After this activity, RAMS engineer knows the description of the architecture, the list of the types of components and occurrences as well as interactions between them (their ports defining inputs and outputs and connections between them). At this stage, components are seen as black boxes.

IV.A.2. Identification of Unexpected Events

First of all, RAMS engineer has to define which events have to be assessed: these events are called "Unexpected events". This step is crucial because it defines the properties that will be qualified by the PSA analysis. The results of this task are a list of unexpected events and their allocation to safety requirements (see Table 2 and Figure 5 in this case study). It is important to define explicitly each event.
The "Unexpected events" are a concretization of safety requirements in RAMS domain, meant to express the system's criteria of acceptability. The "Unexpected Events" (UE) are expressed with formulas of Boolean logic and events are classified according to three categories: "Failure Event", "External Event" and "External Failure". Table 2 shows the list of identified UE:

Allocation between Requirement and Unexpected events :

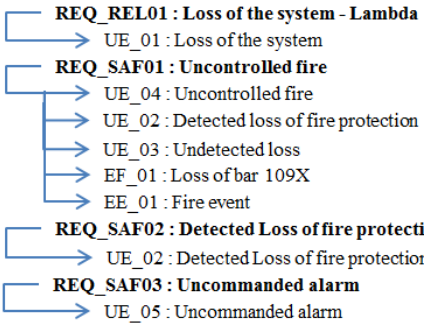

Figure 5 Requirements allocation to unexpected events

Table 2 Unexpected events list of Fire detection system

UE_Number	Unexpected Events (UE)
UE_01	Loss of the system
UE_02	Detected Loss of fire protection
UE_03	Undetected loss of system
UE_04	Uncontrolled fire
UE_05	Uncommanded alarm

Remark: We note that unexpected events are not independent. Example, UE_04 is depending of UE_02 and UE_03.

IV.A.2. Reliability Estimation & FMEAs

The next task consists in performing an estimation of the system's reliability. In this purpose we will first identify the failure modes of components and their distribution; then we will proceed with a FMEA analysis.
Reliability estimation:

We use the MIL HDBK 217 [22] standard to estimation the electronic components' reliability. This handbook is very powerfull although this estimation is getting a little bit old and obsolete. These assumptions are widely acceptable for our needs. Likewise we use an exponential law to characterize the intrinsic reliability for each electronic component used in the system.

The MIL HDBK 217 will provide failure rate λ for each component. The following frame details calculation for the Capacitor C1 of the system.

Capacitor C1: C1: Multilayer ceramic capacitor 100nF, operating voltage: 3,3V, maximum 50V, series resistance factor: 1. Capacitors are manufactured with a standard quality (P level).
This is *CDR Ceramic, Chip, Est.Rel in MIL HDBK 217F - 10.11 Capacitors, Fixed, Ceramic, temperature Compensating and Chip*. $\lambda_p = \lambda_b.\pi_{CV}.\pi_Q.\pi_E$ Failures/10^6 Hours
$\lambda_b = 0.016$; $\pi_{cv} = 2.35$; $\pi_q = 0.30$; $\pi_e = 8$ (A_{IF}); Therefore $\boldsymbol{\lambda_p = 9.02 .10^{-8}}$ **F/H**

Following the same reasoning on the architecture, we obtain the following result for MIL HDBK estimations of all components' reliability:

Table 3 Reliability component estimated from MIL HDBK 217

Component	λ_T	Component	λ_T
Capacitor C1	$8{,}18 .10^{-8}$	Lamp	$1{,}50 .10^{-6}$
Capacitor C2	$9{,}26 .10^{-8}$	Detector	$3{,}60 .10^{-5}$
Resistor R1	$1{,}20 .10^{-7}$	Buzzer	$8{,}00 .10^{-7}$
Resistor R2	$8{,}14 .10^{-8}$	Relay	$2{,}10 .10^{-6}$
Fuse	$1{,}60 .10^{-7}$	MTBF = $1/\Sigma\lambda_T$ = 24 400 hours	

The calculated reliability is intrinsic for each component, but we do not know the repartition this reliability has on internal failure modes. To overcome this problem, we use UTE-C80810 [23], which provides ratios of Failure mode by type of items. The Table 4 is a summary of this standard in our context.

Failure Mode and Effects Analysis, FMEA:

FMEA is a famous method to detail and qualify components in terms of RAMS concerns. A few different types of FMEA analyses exist, we will complete a Design FMEA focused on components and their dysfunctional behaviours. It provides, for each component: failure mode(s), effect(s) of failure modes (local effect, or system effect), and other data. For the use case, we use FMEA to detail failure modes and system effects of components. We allocate component's failure rate to respective failure modes. To do this, we use another reliability data base, the UTE C80-810, to get components failure distribution. The Table 4 is a summary of the extract used in UTE C80-810.

Using UTE C80-810, we conducted FMEAs and obtained the following results summarized in FMEA, Table 5 FMEA synthetize of component in fir fire detection system.

Table 4 Component failure distribution (UTE-C80 810 extract)

Failure Mode Distribution		
Type of Item	**Failure mode**	**Failure mode ratio**
Resistor, CTN	Open Circuit	40 %
	Drift	60 %
Caramic Capacitor	Open Circuit	30 %
	Short Circuit	70 %
Diode	Open Circuit	20 %
	Short Circuit	80 %
Transistor	Open Circuit	15 %
	Short Circuit	85 %
Detector	Loss	33 %
	Uncommanded Function	67 %
Other parts (connector, fuse, switch)	Loss	100 %

Table 5 FMEA synthetize of component in fire detection system

Component	Failure rate component	FM ref number	Failure mode	Distribution	System Effect	Failure rate
Capacitor C1	$8{,}18.10^{-08}$	1.1	Open Circuit	30%	Detected loss	$2{,}46.10^{-08}$
		1.2	Short Circuit	70%	Undetected loss	$5{,}73.10^{-08}$
Capacitor C2	$9{,}26.10^{-08}$	2.1	Open Circuit	30%	Detected loss	$2{,}78.10^{-08}$
		2.2	Short Circuit	70%	Undetected loss	$6{,}48.10^{-08}$
Resistor R1	$1{,}20.10^{-07}$	3.1	Open Circuit	40%	Detected loss	$4{,}78.10^{-08}$
		3.2	Drift	60%	Undetected loss	$7{,}18.10^{-08}$
Resistor R2	$8{,}14.10^{-08}$	4.1	Open Circuit	40%	Detected loss	$3{,}25.10^{-08}$
		4.2	Drift	60%	Undetected loss	$4{,}88.10^{-08}$
Fuse	$1{,}60.10^{-07}$	5.1	Loss	100%	Undetected loss1	$1{,}60.10^{-07}$
Lamp	$1{,}50.10^{-06}$	6.1	Loss	100%	Undetected loss1	$1{,}50.10^{-06}$
Sensor	$3{,}60.10^{-05}$	7.1	No output	33%	Undetected loss1	$1{,}19.10^{-05}$
		7.2	Uncommanded Function	67%	Uncommanded alarm1	$2{,}41.10^{-05}$
Buzzer	$8{,}00.10^{-07}$	8.1	Loss	100%	Undetected loss1	$8{,}00.10^{-07}$
Relay	$2{,}10.10^{-06}$	9.1	Open Circuit	95%	Undetected loss1	$2{,}00.10^{-06}$
		9.2	Short Circuit	5%	Uncommanded alarm1	$1{,}00.10^{-07}$

IV.A.3. Unexpected Events constructions and Failure Behaviors specification

Before quantifying unexpected events, we have to define which failure modes will affect the unexpected events. To achieve this qualitative analysis, we use information from FMEA and the list of unexpected events. As previously announced, "Unexpected Events" are expressed by Boolean logic and events as "Failure Event", "External Event" and "External Failure".

UE_01 = FM1.1 + FM1.2 + FM2.1 + FM2.2 + FM3.1 + FM3.2 + FM4.1 + FM4.2 + FM5.1 + FM6.1 + FM7.1 + FM7.2 + FM8.1 + FM9.1 + FM9.2

UE_02 = FM1.1 + FM2.1 + FM3.1 + FM4.1

UE_03 = FM1.2 + FM2.2 + FM3.2 + FM4.2 + FM5.1 + FM6.1 + FM7.1 + FM8.1 + FM9.1

UE_04 = EE_01 . (EF_01 + UE_02 + UE_03)
= EE_01 . (EF_01 + FM1.1 + FM1.2 + FM2.1 + FM2.2 + FM3.1 + FM3.2 + FM4.1 + FM4.2 + FM5.1 + FM6.1 + FM7.1 + FM8.1 + FM9.1)

UE_05 = FM7.2 + FM9.2

Where, UE_i are Unexpected events, FMj.k are Failure Modes, EE External Event and EF External Failure.

IV.A.4. PSA Analysis and Results

To help us perform the calculation of each UE, we choose to use a modeling language dedicated to RAMS analysis: AltaRica 3.0. AltaRica is a high level modeling language dedicated to safety analysis. The main motivation for using this language is that models designed with "classical" formalisms such as Fault Trees, Markov chains, Petri nets and the likes are too distant from systems under study. The language is based on the guarded transition system paradigm; that brings a formal mathematic model. Moreover, the AltaRica Association provides various powerful tools using this formalism and dedicated to support specific safety and methods of reliability assessment: Fault Tree compiler, Sequence generator, Markov chain generator, Stochastic simulator, Model checking, reliability allocation and stepwise simulator.

To assess our requirements, the safety model is transformed into AltaRica 3.0 formalism. Then, we use a Fault Tree compiler to build Fault Trees of each unexpected event in Open-PSA model exchange format. After this, we use xFTA, to compute the Top Event probability.

The AltaRica 3.0 formalism allows the description of components and system architectures in a simple way.
The Figure 6 *Elementary block (example the resistance R1)* shows the description of Resistor R1 in AltaRica 3.0 formalism. It can describe components, inputs, outputs, failure modes and failure behaviors. In the example bellow, ResistorR1 has two failure modes FM31 and FM32; the failure behavior on failure events occurrence is specified using guarded transition rules; failure propagation is specified using propagation assertions (here between entry e1 end exit s1). Once this is done, it is possible to specify a propagation model using assertions between components within the system. The formalism thus enables the modeling of hierarchically defined systems and sub-systems.

```
//------------the Resistor---------
class Resistor1
//state declaration
   Boolean FM31(init =  false);
   Boolean FM32 (init = false);
//flow declaration
   port e1, s1;
   event Resistor_ShortCut_occurs ( delay = exponential( 0.0000000478 ) ) ;
   event Resistor_OpenCircuit_occurs ( delay = exponential( 0.0000000718 ) );
   transition
      Resistor_ShortCut_occurs: Resistor_FailureMod1 ==  false -> Resistor_FailureMod1:=true ;
      Resistor_OpenCircuit_occurs: Resistor_FailureMod2 == false -> Resistor_FailureMod2:=true ;
   assertion
      s1 := (not(FM31) and not(FM31) and e1);
end
```

Figure 6 Elementary block (example the resistance R1)

Figure 7 System and Environment presents an equivalent graphical view of the system's structural architecture as is implemented in AltaRica 3.0. Connections between blocks reflect the assertions defined between system components. Moreover it is possible to define observers that can track failure events involved in Unexpected Events occurrence.

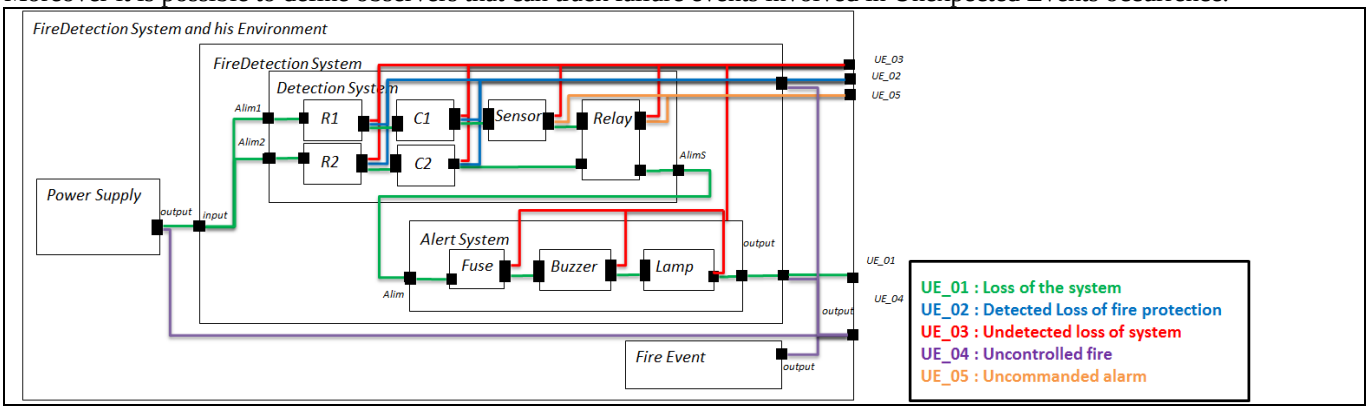

Figure 7 System and Environment

Using the Fault Tree compiler of AltaRica tools, we obtain five fault trees with Top Events corresponding to the previously identified UE. They are stored in the OpenPsa formalism. The Figure 8 *Fault tree of UE_01 Loss of the system* shows the generated fault tree, [24] for UE_01. We drive the same exercice for the four other unexpected events.

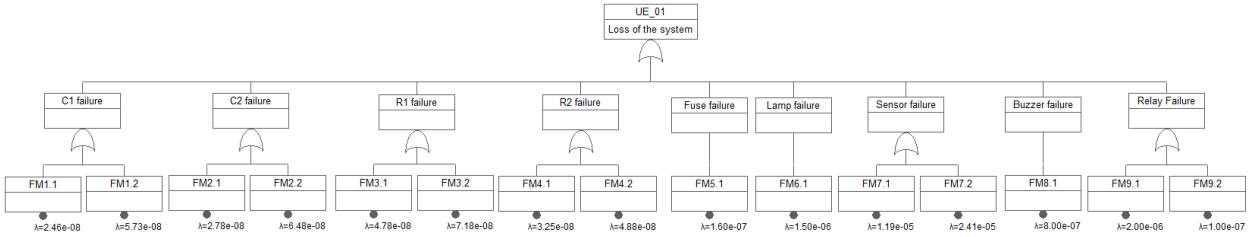

Figure 8 Fault tree of UE_01 Loss of the system

At this stage we use the xFTA tool to compute probabilistic data. XFTA is an assessment engine working on models, written in the Open-PSA format. XFTA version 1.1 implements an efficient algorithm to compute Minimal Cutsets. It implements also other probabilistic assessments:

- Calculation of the Top Event probability for different mission times.

- Calculation of Importance Factors for Basic Events (Birnbaum/Marginal Importance Factor, Critical Importance Factor, Diagnosis Importance Factor, Risk Reduction Worth, Risk Achievement Worth).
- Sensitivity Analyses by means of Monte-Carlo simulation.
- Calculation of Safety Integrity Levels for low demand mode and high/continuous demand mode for safety-related systems as required by Safety Standard IEC 61508 and daughters.

The reliability of the five top events, is obtained by executing the xFTA computation of the Top Event probability for different mission times. In the case study, the mission time is 1000h. The table *Table 6 : Unexpected event calculated by XFTA* shows corresponding results:

Table 6 : Unexpected event calculated by XFTA

UE Number	**Unexpected Events (UE)**	Value
UE_01	Loss of the system	$4{,}09.10^{-5}$
UE_02	Detected Loss of fire protection	$1{,}33\ .10^{-7}$
UE_03	Undetected loss of system	$1{,}66\ .10^{-5}$
UE_04	Uncontrolled fire	$8{,}51\ .10^{-8}$
UE_05	Uncommanded alarm	$2{,}42\ .10^{-5}$

In the same manner, we can easily obtain the calculated value for unexpected event for each requirement. Results obtained can be seen in table *Table 7 Summary result on safety requirement*:

Table 7 Summary result on safety requirement

Type of requirement	**Title** (link to Fire detection system req)	**Values (Objective)**	**Values (Calculated)**	**Requirement Validity**
Reliability	REQ_REL01 : Loss of the system	10^{-4} /FH	$4{,}09.10^{-5}$	**True**
Safety	REQ_SAF01 : Uncontrolled fire	10^{-9} /FH	$8{,}51.10^{-8}$	**False**
Safety	REQ_SAF02 : Detected Loss of fire protection	10^{-6} /FH	$1{,}33.10^{-7}$	**True**
Safety	REQ_SAF03 : Uncommanded alarm	10^{-4} /FH	$2{,}42\ .10^{-5}$	**True**

The analysis of these results allows one, according to the objective value, to conclude on the validation of requirements.
However, we can note that the safety objective REQ_SAF01 of 1. 10^{-9} is not met. Therefore, a modification needs to be applied on the architecture model to try and satisfy all requirements.
In this situation, the RAMS engineer should contact the System Architect to inform him of the situation. He could also suggest solutions to avoid this unsatisfied Safety requirement.
In our case, the RAMS engineer recommends to establish a redundancy of order 2 of the entire fire detection system.

IV.B. Architecture improvement and Second PSA Analysis

Following the first PSA Analysis, recommendations have been issued to System Engineers. These recommendations have been accepted and taken into account. The system architecture is modified accordingly. It is now time to reanalyse the modified system. We describe hereunder this second iteration. It is important to note, that this second iteration is easier to conduct because model and data are already structured. It is just necessary to update some information (steps B1 and B2), taking care of not missing any data dependency when updating the model. This can be instrumented in the future thanks to existing powerful MDE tools.

IV.B.1. Update Architecture models

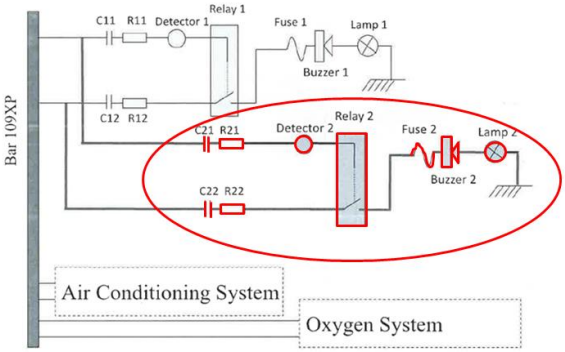

Figure 9 New fire detection system assembly

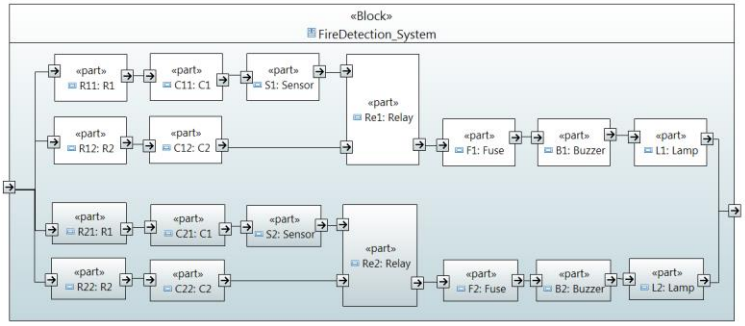

Figure 10 SysML architecture description Updated

In this new model, redundancy has been introduced as can be seen in Figure 9 New *fire detection system assembly*
Figure 10 SysML architecture description Updated (in red) and in the SysML model. The corresponding AltaRica architecture has also been adapted accordingly.

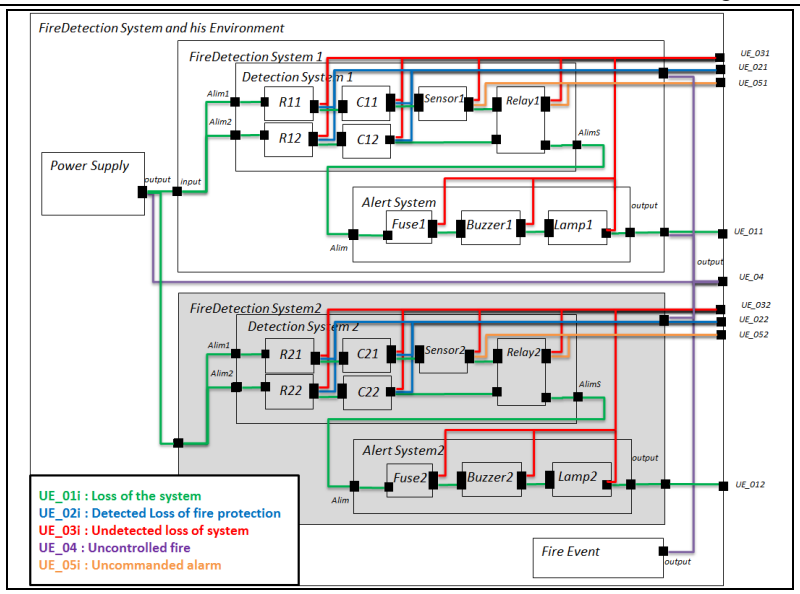

Figure 11 System and Environment, AltaRica 3.0 model updated for second iteration PSA

IV.B.2. Update Safety Data

Unexpected Events Update :

The expression of Unexpected Event from previous PSA analysis is preserved but the architecture forces to include a second level of decomposition of the unexpected events.

Table 8 FMEA Updated for second iteration PSA

UE_ Number	Unexpected Events (UE)	UE_ Number	Unexpected Events (UE)	Unexpected Event constructions
UE_01	Loss of the system	UE_011	Loss of the system1	UE_01 = UE_011 + UE_012
		UE_012	Loss of the system2	
UE_02	Detected loss fire protection	UE_021	Detected Loss 1 of fire protection	UE_02 = UE_021 + UE_022
		UE_022	Detected Loss 2 of fire protection	
UE_03	Undetected loss of system	UE_031	Undetected loss 1 of system	UE_03 = UE_031 + UE_032
		UE_032	Undetected loss 2 of system	
UE_04	Uncontrolled fire	UE_041	Uncontrolled fire 1	UE_04 = EE_01 . (EF_01 + UE_02 + UE_03)
		UE_042	Uncontrolled fire 2	
UE_05	Uncommanded alarm	UE_051	Uncommanded alarm 1	UE_05 = UE_041 + UE_042
		UE_052	Uncommanded alarm 2	

We can note that unexpected events appear twice in this update. In fact, the previous unexpected events are allocated in each redundancy branch.

Estimation of the reliability :

The new components are still estimated using MIL HDBK 217. Since components are just duplicated, the characteristics of their failure models are the same, and the estimation of their reliability does not change.

FMEAs update :

FMEA is completely similar to the previous one, but we have now duplicated components. Therefore, we get twice as many failure modes, but components Failure rates, distribution, system effects and reliability of failure mode are not impacted by the update.

IV.B.4. Second PSA Analysis and Results

At this stage, we used the same tool chain than as for the first PSA analysis. From AltaRica 3.0 model, we generate Fault Trees of the ten unexpected events. To obtain values of top events, we use again xFTA. The results are summarized in the Table 9 *Summary result on safety and reliability requirement*.

Table 9 Summary result on safety and reliability requirement

UE_ Number	Unexpected Events (UE)	Value	Requirement type	Title (link to Fire detection system req)	Values (Objective)	Values (Calculated)	Validity
UE_01	Loss of the system	$8{,}19.10^{-5}$	Reliability	Loss of the system	10^{-4} /FH	$8{,}19.10^{-5}$	**True**
UE_02	Detected loss fire protection	$2.66.10^{-7}$	Safety	Uncontrolled fire	10^{-9} /FH	$1{,}02.10^{-9}$	**True**
UE_03	Undetected loss of system	$5.32.10^{-5}$	Safety	Detected Loss of fire protection	10^{-6} /FH	$2.66.10^{-7}$	**True**
UE_04	Uncontrolled fire	$1{,}02.10^{-9}$	Safety	Uncommanded alarm	10^{-4} /FH	$4{,}82\ .10^{-5}$	**True**
UE_05	Uncommanded alarm	$4{,}82\ .10^{-5}$					

We note that all requirements of the case study are validated. Therefore, we inform the system designer that the current architecture meets its requirements. If any modification is done on the architecture, PSA analysis has to be conducted again within a new iteration.

V. FEEDBACK AND ANALYSIS

After leading PSA applications in this paper, we can provide feedback as pros and cons of this experimental work. PSA uses many concepts: we have consider the dependencies between concepts, especially if we iterate several analyses. By using MBSA approach and MDE techniques we can avoid human errors, misinterpretation and data omission, because we can define rules to insure dependencies between concepts.

The presented analysis enables one to save an important amount of time after the first PSA analysis by updating models. Even for the first analysis, the model is already populated by system components and their interactions. Moreover, we get a graphical navigable model which improves model understanding and makes the detection of dependencies between components easier. This could result in a significant gain in big projects, such as the development project of a helicopter.

There are many ways to conduct a PSA analysis. We suggest an original way to conduct it by using MBSA approaches and using different tools and languages. The application of the approach on the use case has enabled the validation of the Safety and Reliability requirements according to the current version of the architecture design description.

The paper has two major contributions: the presentation of the use case and the experimental applications of the PSA methodology. We present an industrial case in the aerospace business field. The use case presents real problematics and a real context. This use case has been provided by APSYS, a subsidiary company of Airbus Company, specialized in decisions help focused on Safety Assessment and Integrated Logistics Support.

By using the use case, we clearly define the engineering context, the characteristics of the stakeholders involved (system designer and RAMS engineering experts), their skills, roles, expectations and work products. We also offered an overview of a PSA application case. This methodology proposal presents some particularities:

- It supports iterative cycles to conduct Probabilistic Safety Analysis on several versions of architecture design. This iterative approach enables the conduction of the PSA analysis by updating models (after the first iteration).
- It is an MBSA application, because it uses a model-based tool chain dedicated to several concerns in RAMS analysis. The presented application uses a set of modeling formalisms, tools and methods dedicated to each concern and related expert stakeholders. In brief we use the SysML language to represent architecture design, Sophia profile as an intermediate viewpoint between architecture design and safety architecture, Altarica3.0 to express formal high level safety architecture models and OpenPSA format to capture fault trees. We use several tools: Papyrus, Sophia, AltaRica editor, XFTA.
- It respect standards dedicated to the aerospace field: ARP 4761, ARP 4754 and DO 178 to conduct PSA analysis. We apply recommended process, methods (FHAs, FMEAs, FTAs) and concepts (unexpected event, failure mode, failure event, reliability and safety properties).
- It is helpful to RAMS Engineers, because it helps build RAMS models faster, and conform to architecture design models. It also makes the exploitation of the collected information (failure models) and results easier, by other tools thanks to the OpenPSA exchange format used.

This paper presents a part of a PhD Work. It is entitled “System engineering and RAMS: proposal of methodology synchronization between models used in architecture design and models used in safety analysis”. This work is a suggestion for a synchronization methodology and the associated framework, which contains several types of tooling to support specific exchanges at particular steps in dedicated processes.

VI. CONCLUSIONS

This paper has shown within a real engineering context how it is possible to benefit from different expert skills to accelerate and improve the construction of safety and reliability models in order to provide support to PSA analysts. We have shown that RAMS and System engineering fields clearly need each other. After presenting the characteristics of stakeholders involved (system designer and RAMS engineering experts), role expectations and work products, we have applied an interaction pattern between these domains of expertise in order to improve cooperation and get better models in an easier and faster way. This has been illustrated in the study of a fire detection system case. The system architecture model helped design the basis of the RAMS model. The RAMS engineer has to precise the safety context, safety and reliability requirements concerned in the conducted PSA analysis, but the components architecture was consistent (by construction) with that of system engineers. The PSA analyses have been applied iteratively to illustrate the feasibility of exchanges between both experts, through model exchanges or recommendations from RAMS. It has been shown that such recommendations can be taken into account within architecture and reanalyzed in a short loop. The results are obtained using advanced RAMS tooling based on AltaRica formalism, XFTA for top event’s reliability. The study has involved several tools and proved that

interoperability between them is possible. In the future, we intend to exploit such tool chains and provide further support to impact the analysis of the change of models and assistance to conflict resolution.

ACKNOWLEDGMENTS

This work is part of my thesis contributions. The thesis is co-supervised by Agnès LANUSSE at CEA LIST, and Antoine RAUZY (Director of thesis). This thesis is funded by LISE laboratory CEA LIST and the DGA (the French Defense Procurement Agency). I would also like to thanks APSYS (Subsidiary company of Airbus Company specialized in decisions help focus in Safety Assessment and Integrated Logistics Support) for allowing the dissemination of cases studies.

REFERENCES

[1] A. Rauzy, A. Epstein Steven, *Open-PSA Model Exchange Format,* 2008.

[2] INCOSE, *Systems Enginneering Vision 2020,* 2007.

[3] GEIA/EIA, *EIA-632 Processes for Engineering a System*, 1999.

[4] ISO/IEC/IEEE, *ISO/IEC/IEEE 15288 Systems and software engineering - System life cycle processes,* 2015.

[5] G. Sebastien, «Modelisation uml executable pour les systemes embarques de l'automobile,» 2000.

[6] P. Roques, «MBSE with the ARCADIA Method and the Capella Tool,» at *8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)*, Toulouse, France, 2016.

[7] OMG, *OMG Systems Modeling Language (OMG SysML™),* 2012.

[8] Peter H. Feiler et al., *The Architecture Analysis \& Design Language (AADL),* 2006.

[9] ATTEST. members, *EAST-ADL Domain Model Specification,* 2010.

[10] S. Tucci-Piergiovanni, et al., «Model-Based Analysis and Engineering of AutomotiveArchitectures with EAST-ADL,» at *Handbook of Research on Embedded Systems Design :* , 2014, pp. 242-282.

[11] P. David, V. Idasiak et F. Kratz, «MéDISIS, l'intégration des analyses de SdF aux processus d'Ingénierie Systèmes Basée sur les Modèles,» at *17eme Congres Lambda Mu : Innovation et maitrise des risques*, La Rochelle, France, 2010.

[12] N. Yakymets, H. Jaber and A. Lanusse, «Model-based System Engineering for Fault Tree Generation and Analysis,» at *MODELSWARD 2013 - Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development, Barcelona, Spain, 19 - 21 February, 2013*, 2013.

[13] N. Yakymets, M. Perin and A. Lanusse, «Model-driven multi-level safety analysis of critical systems,» at *Systems Conference (SysCon), 2015 9th Annual IEEE International*, 2015.

[14] F. Mhenni, N. Nguyen and J.-Y. Choley, «SafeSysE: A Safety Analysis Integration in Systems Engineering Approach,» *IEEE Systems Journal,* May 2016.

[15] F. Belmonte et E. Soubiran, «A Model Based Approach for Safety Analysis,» at *Computer Safety, Reliability, and Security - SAFECOMP 2012, Magdeburg, Germany, September 25-28, 2012. Proceedings*, 2012.

[16] T. Prosvirnova, «AltaRica 3.0: a Model-Based approach for Safety Analyses,» 2014.

[17] H. Mortada, T.Prosvirnova and A. Rauzy, «Safety Assessment of an Electrical System with AltaRica 3.0,» at *Proceedings of the 4th International Symposium on Model-Based Safety Assessment, IMBSA 2014*, Munich (Germany), 2014.

[18] Y. Papadopoulos, M. Walker, D. Parker, et al., «Engineering failure analysis and design optimisation with HiP-HOPS,» *Engineering Failure Analysis ,* vol. 18, n°12, pp. 590-608, 2011.

[19] RTCA, *DO-178C Software Considerations in Airborne Systems and Equipment Certification,* 2006.

[20] SAEAerospace, *ARP4761 Guidelines and Methods for conducting the Safety Assessment Process on Civil Airborne Systems and Equipment ,* 1996.

[21] SAEAerospace, *ARP 4754 Guidelines for Development of Civil Aircraft and Systems,* 2010.

[22] Departement. of Defense of US, MIL HDBK 217 F, DoD, Éd., 1991.

[23] UTE, UTE C 80-810, RDF, 2000.

[24] E. Clement, A. Rauzy, T. Thomas, «ARBRE ANALYSTE: UN OUTIL D'ARBRES DE DEFAILLANCES RESPECTANT LE STANDARD OPEN-PSA ET UTILISANT LE MOTEUR XFTA,» *19E Congrès de Maîtrise des Risques et Sureté de fonctionnement,* Octobre 2014.