Improved Safety Analysis Integration in a Systems Engineering Approach

Anis Baklouti, Nga Thi Viet Nguyen, Faida Mhenni, Jean-Yves Choley, Abdelfattah Mlika

applied sciences

MDPI

Article

Improved Safety Analysis Integration in a Systems Engineering Approach

Anis Baklouti [1,2,*], Nga Nguyen [3], Faïda Mhenni [2], Jean-Yves Choley [2] and Abdelfattah Mlika [1]

1 Laboratory of Mechanics of Sousse, National Engineering School of Sousse, University of Sousse, Hammam Maarouf, Sousse 4054, Tunisia; abdelfattah.mlika@gmail.com
2 Quartz Laboratory, SUPMECA, 93400 Saint-Ouen, France; faida.mhenni@supmeca.fr (F.M.); jean-yves.choley@supmeca.fr (J.-Y.C.)
3 ETIS Laboratory, EISTI, 95000 Cergy, France; nn@eisti.eu
* Correspondence: bakloutianiseniso@gmail.com

Received: 31 January 2019; Accepted: 4 March 2019; Published: 25 March 2019

Abstract: The goal of the paper is the integration of safety analysis in a model-based systems engineering approach to ensure consistency between system design and safety artifacts. This integration permits the continuous improvement of the structure and behavior of the system. It also reduces system development time and prevents late detection of errors. To reach this purpose, the SafeSysE methodology is extended. In SafeSysE, a preliminary Failure Mode and Effects Analysis (FMEA) is automatically generated from a SysML model, and this FMEA is then completed by the safety expert but no further development was proposed. The contribution of this paper is to suggest recommendations based on the FMEA analysis in order to enhance the system design and make it comply with safety requirements. First, an updated system structure that may contain redundancy is proposed. Then, a redundancy profile is used to enrich the system model with redundancy information, which will allow the generation of a dynamic fault tree considering the system behavior. Finally, the generated dynamic fault tree should be analyzed in order to create a state machine diagram that describes the behavior of the system. The created state machine with an internal block diagram will help the system designers to better understand the system dysfunctions by simulating the system. The proposed methodology is applied to an Electro-Mechanical Actuator system which is used in the aeronautics domain.

Keywords: MBSE; SA; FMEA; dynamic fault trees; redundancy; redundancy profile; system description

1. Introduction

Safety Analysis (SA) aims at assessing system safety and ensuring a Satisfactory Safety Level (SSL) of designed systems. Different techniques and methods are used to evaluate system safety [1,2] such as HAZard and OPerability Analysis (HAZOP), Reliability Block Diagrams (RBD), Process Hazard Analysis (PHA), and the two most traditionally used techniques are Failure Mode and Effects Analysis (FMEA) and Fault Tree Analysis (FTA). FMEA is an inductive bottom-up safety analysis method that allows the identification of Failure Modes (FM) of system components or functions and the determination of their respective effects at the system level. The analysis, then, aims at removing the potential risks from the design of the system. FTA is a deductive top-down approach that starts with an undesired Top Level Event (TLE) and progressively deduces its root causes [3], through a graphical logical representation of fault events. FTA aims at identifying all possible routes that lead

to an undesired TLE and then mitigates the potential risks by making architectural and behavioral modifications. FTA can be used in a quantitative and qualitative approach.

In practice, there is a gap between Model Based Systems Engineering (MBSE) [4] and SA as the latter takes place very late in the design process. This gap leads to inconsistency between Systems Engineering (SE) models and SA models. This inconsistency between models inhibits the dynamic updating of a critical system design to consider the required modifications after safety analyses are performed. Moreover, in the design process, this gap between models prevents the consideration of safety constraints early enough, which can be very costly when problems are detected late. The design process of the system may be organized in four main phases, which are the early concept phase, the design phase, the development phase, and the production phase. According to INCOSE [4], the elimination of the design faults detected during the design phase is 6 times more costly than removing them in the early concept phase. This number can be multiplied by 100 and 1000 if the design fault is detected during the development, production, or test stage. Hence, the integration of safety assessment into the systems engineering process is very important. To narrow the gap between SE and SA and thus enhance consistency and reduce late design changes, many research works dealt with the integration between MBSE and SA, such as in [5–14].

In this paper, we propose a methodology that contributes in further reducing the gap between MBSE and SA. This work represents an extended version of the SafeSysE methodology [8,15,16]. In SafeSysE, the authors worked on the integration of SA in the MBSE process. FMEA and Static Fault Trees (SFT) are automatically generated from SysML diagrams, such as Internal Block Diagram (IBD) and activity diagrams. However, SafeSysE does not consider the feedback from the generated artifacts in order to recommend enhancements of system architecture and behavior. To efficiently guide the systems engineer in making system architectural modifications according to the safety analysis recommendations, SafeSysE needs to be completed. The aim of this work is to extend SafeSysE by using FMEA and FTA feedback to improve system design, as well as by a new Dynamic Fault Tree (DFT) generation approach.

The paper is organized as follows. Section 2 discusses the state of the art concerning the integration of safety analysis in the systems engineering process. Section 3 gives an overview of the proposed methodology. Section 4 describes the first and second steps of this methodology (i.e., the automatic generation of FMEA from the system description and then structural modifications proposal based on FMEA information) that aim at improving the system architecture. Section 5 describes the third and the fourth steps of the methodology aiming at enhancing the system behavior. A dynamic fault tree is generated, automatically, from a system model. To enrich this generation with redundancy information, a redundancy profile is created. Then, the dynamic fault tree analysis is performed and a state machine diagram which describes the behavior of the system is generated. The case study Electro-Mechanical Actuator (EMA) is applied in Section 6 to illustrate our approach. Finally, the paper is concluded in Section 7.

2. State of the Art

This section discusses the state of the art concerning the integration of safety analysis in the systems engineering approach in order to compare the existing work with the proposed methodology. The related work is divided into two parts: the integration of the SA in the MBSE process and the automatic generation of FTA.

For the integration of SA and MBSE, most of the works considered the extention of system models to enable the automatic generation of safety artifacts. In [12], Helle proposed a methodology that deals with the integration of the MBSA in the MBSE process. A SysML extension is created in order to include information about safety in the system model, which gives an opportunity to the systems expert to make some decisions without the assistance of the safety expert. A Safety Analyzer program is developed in order to extract relevant information from the system model. As outputs of the developed program, a group of minimal cut sets is given for all system failure modes.

David et al. presented a methodology in [17] called MéDISIS. On one hand, it tries to generate an FMEA from structural and behavioral diagrams in SysML models. On the other hand, it aims at constructing the dysfunctional models using AltaRica language [18] to compute the reliability indicators. The methodology starts by an automatically generating a preliminary FMEA. A set of structural diagrams (Internal Block Diagram (IBD) and Block Definition Diagram (BDD)) and behavioral diagrams (activity diagram and sequence diagram) are analyzed in detail in order to give, for each component and function, a complete list of failure modes with their possible causes and effects. Then, a safety expert completes the final FMEA. The second step of MéDISIS is the mapping between SysML models and AltaRica language in order to ensure the direct assignment of reliability indicators to the failure modes generated in the first step.

Concerning the generation of the fault trees from system models, we cite some works that allow the generation of the static and dynamic fault trees from system structure. The related work is divided into two categories: works that do not consider the redundancy in the generation of fault trees such as [19–24] and others that consider the redundancy during the generation of fault trees such as [25–29].

Mahmud et al. in [28] developed an algorithm that allows the generation of the Temporal Fault Trees (TFT) from state machines. A recent technique for introducing temporal logic to fault trees named Pandora is used to ensure the generation. This work considers the redundancy behaviour during the transformation from the state machine to TFT.

Pai and Dugan in [27] developed an approach which permits the dynamic fault trees generation from UML diagrams. The description of the system and the components dependencies are modeled by UML diagrams. A failure model, which gathers all components properties, is created using the system description and the failure properties of components. Then, based on the functional dependencies between components, the logical gates of FT are chosen. Finally, each logical gate is allocated to its respective nodes of the failure model.

Dehlinger and Dugan in [26] extended the approach of Anjali [21] in order to generate a DFT from Architecture Analysis and Design Language (AADL). To consider the redundant components and the temporal sequences of failures in the DFT, the static fault tree generated by Anjali is enriched. They consider the redundancy, the dependent failures, and the reconfiguration activities during the generation of FTs. These works present some limits such as they only use the static and spare gates and they do not manipulate any of the active, standby (with and without switch), and mixed redundancies.

In [29], Zhao proposed an approach which allows the generation of the SFT using a model transformation from UML diagrams. The UML diagrams is enriched by a profile in order to integrate information in a meta-model that describes the FT. Then, the model transformation is implemented using the Atlas Transformation Language (ATL).

In [25], Tajarrod and Latif-Shabgahi developed a method that uses the Simulink models in order to generate a SFT and DFT. On one hand, the Simulink models are enriched by a failure model. On the other hand, the subsystems that affect the top level event of the system are considered. In addition, the functional dependencies and the structure of the system are used to identify the corresponding logical gates. As in [26], this work considers the redundancy during the FT generation but its limit is that just the static logic gates are used, which prevents consideration of the failure components order.

3. Improved SafeSysE Methodology

In this section, an overview of the project is given. It aims at integrating safety in the systems engineering approach in order to ensure consistency between models and to improve the system architecture and behavior. As shown in Figure 1, the process is composed of four main steps.

The first step is the automatic generation of a Preliminary Failure Mode and Effects Analysis (PFMEA) from SysML diagrams by using a Safety Profile [16]. Then, a safety expert completes the PFMEA with different information in order to obtain an FMEA with Severity (S), Occurrence (O), and Detectability (D) parameters. For more detail about the FMEA generation please refer to Section 4.1. In the second step, the generated FMEA is performed and used to make some architecture modifications in order to improve its reliability. In this work, redundancy is used as a solution to reduce the risk of critical components (Section 4.2). When the system architecture is modified, it is necessary to specify its behavior. Therefore, a redundancy profile is created in order to generate a DFT from SysML diagrams (Section 5.1). In the fourth step, the generated DFT is analyzed and the minimal cut sequences are used to create a state machine diagram to describe the system behavior (Section 5.2).

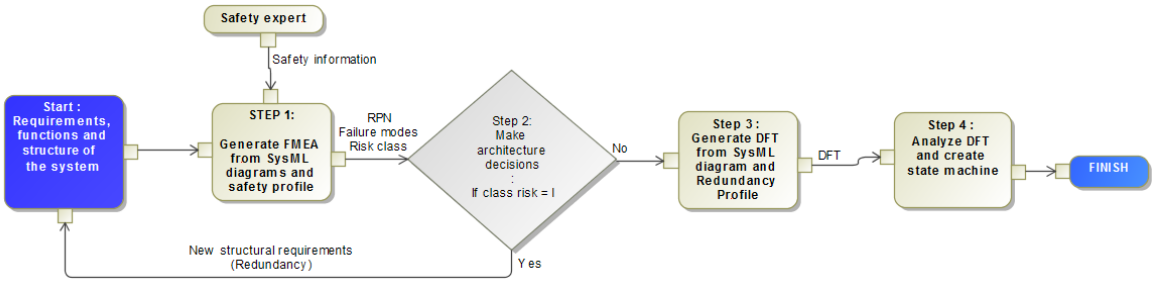

Figure 1. Improved SafeSysE: flowchart of the general methodology.

In the Improved SafeSysE, the generation of FMEA from SysML diagrams is the same as in SafeSysE [16]. Then, by using the Risk Priority Number (RPN) of FMEA, some architecture modifications are made as shown in [8]. The improvements in Improved SafeSysE is that we use the criticality instead of RPN because RPN can not always identify the true critical components. Furthermore, Improved SafeSysE extends [15] and describes the generation of a DFT from SysML diagrams, which considers redundancy between components. Once the DFT is generated, Improved SafeSysE allows the generation of a STate Machine (STM) using information from qualitative analysis of DFT.

The overall methodology is described by an activity diagram which represents the detailed process as shown in Figure 2. This process is divided into three parts: the systems engineering part, the safety assessment part, and the decision making part. The starting point of the process is a set of initial requirements which describes the needs of the system. Each activity has a data store which allows the storage of different information.

For the SE part, different activities like defining requirements, system architecture, system behavior, and functions are modeled by SysML diagrams. The first activity is the definition of the systems requirements which describes the functionality of the system. To identify these requirements, some SysML diagrams can be used such as the requirement diagrams, the block definition diagrams, and the use case diagrams. The second activity is the definition of the system functions. These functions are suggested by using the system requirements identified in the previous step. The system functions are modeled, in SysML, by a set of activity diagrams. Each activity diagram describes the decomposition of a given function into sub-functions and it represents the transformation of input flows into output flows. Once the system functions are defined, the system architecture is built by assigning each component to its corresponding functions. In SysML, the system architecture is modeled by a BDD and an IBD, which describe the components of the system and the interactions between each other. Finally, the system behavior in SysML is modeled by a set of state machines which aims at describing the behavior of all components or sub-systems of the system.

The decision making part and the SA part of the detailed process will be discussed in the next section.

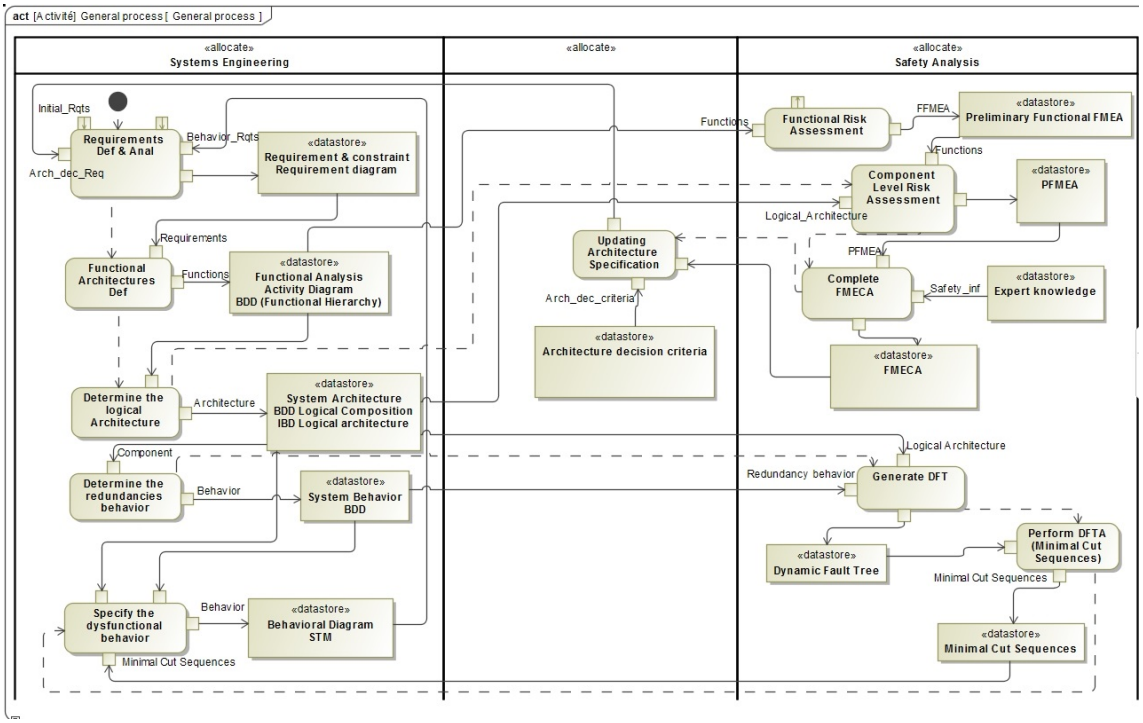

Figure 2. Detailed process of the general methodology.

4. Improved System Architecture Based on FMEA Recommendation

In this section, we present the first two steps of the Improved SafeSysE, which aim at generating FMEA from SysML diagrams, identifying the critical failure modes, then proposing modifications of the system architecture. This part offers a way to integrate these corrections in the systems engineering process.

4.1. Step 1: FMEA Generation from SysML Diagrams

FMEA is a reliability tool widely used in safety analysis. Its target is to identify the critical Failure Modes (FM) that lead to the dysfunction of the system. During the system design process, it also evaluates the effect of possible failure modes and then reduces the potential risks. FMEA is presented as a table that contains components, functions, FM, S, O, D, and the Risk Priority Number (RPN). The risk priority number is computed by the multiplication of the three parameters (RPN = S $\times$ O $\times$ D). It is widely used in engineering analysis. Each FM will be allocated with its RPN and then analyzed in order to give corrective actions [30]. The objective of these corrective actions is the elimination of the critical failure modes characterized by the higher RPN.

The first step of this methodology is the generation of FMEA from SysML diagram described in Figure 2. After defining the system requirements as indicated in the previous section, the system function is established. The stored functions are used to automatically generate a functional FMEA that contains a list of functions and a generic FM. After the establishment of the system function, the structure of the system is defined and each component is allocated to its functions. This logical architecture is used to generate automatically the Preliminary FMEA (PFMEA), which contains components, functions, and failure modes. This PFMEA ensures the allocation of components to their functions and failure modes. Then, a safety expert completes the PFMEA with necessary information (S, O, and D) to obtain FMEA. For each failure mode, the RPN and the risk class are defined and stored with other information in the data store. For more details about the generation of FMEA from SysML diagrams please refer to [16].

4.2. Step 2: Making Architecture Decision Using FMEA Information

Severity [30] is a rating between 1 (negligible effect) and 10 (catastrophic effect) and it indicates the severity of an effect of a potential FM. Four main categories are used to classify the impact of an occurrence of a FM which are "catastrophic", "critical", "marginal", and "negligible". The catastrophic category is the most critical category and it is used when multiple losses of life are expected. The critical category is used when there is just loss of one life. The marginal and negligible categories are used when major or minor injuries are detected. Occurrence is also rated between 1 and 10. It gives an idea of the frequency or the probability of the failure mode occurring during the system life time. Just like severity, the occurrence is classified into six different categories, which are "Frequent", "Probable", "Occasional", "Remote", "Improbable", and "Incredible". The frequent category corresponds to an inevitable failure with a range of failures bigger than 0.001 per year. The probable category corresponds to repeated failures with a range of failure between 0.001 and 0.0001 per year. The occasional category corresponds to occasional failures with a range of failure between 0.0001 and 0.00001 per year. For the remote, improbable, and incredible categories, the failure should be very unlikely or never happen in the life of the system. Detection is rated between 1 and 10 and it is defined by an opportunity for an unidentified failure because of difficulty in detecting it [30].

The IEC61508, which is an international standard published by the International Electro-technical Commission [31], divides the possible risk, depending on the occurrence and severity categories, into four class as shown in Table 1. Class I, which is the most dangerous, means that the risk is unacceptable in any circumstance. Class II means that risk is undesirable but tolerable only if the reduction of the risk is very costly. Class III and class IV mean that the risk is tolerable.

Table 1. IEC61508 risk class matrix.

Occurrence/Severity	Catastrophic	Critic	Marginal	Negligible
Frequent	I	I	I	II
Probable	I	I	II	II
Occasional	I	II	III	III
Remote	II	III	III	IV
Improbable	III	III	IV	IV
Incredible	IV	IV	IV	IV

Once the generation of FMEA is made, the FMEA information is analyzed and some modifications about the structure of the system are recommended according to architecture decision criteria. When the risk of the failure mode is classified as class I, the structure of the system should be updated in order to decrease the criticality of the failure mode. Hence, the component related to this failure mode will be dubbed. The redundancy is the solution used in this work to correct the structure of the system and enhance its safety. After making architecture decisions, three SysML diagrams will be manually updated, which are the requirement diagram, to say that the component "X" should be dubbed, the activity diagram, and the IBD to modify the structure of the system. Then, a new iteration should be made and a new FMEA will be generated and computed. When the system reaches the safety level, no more changes need to be made in the system architecture and it is time to move to Step 3.

5. Improved System Behavior Based on DFT Recommendation

In this section, we present the third and fourth steps of the methodology. This part aims at proposing a redundancy profile and generating a dynamic fault tree from system description. Its final goal is to give a description to the system behavior using the DFT information.

5.1. Step 3: Automatic Generation of DFT from SysML Diagrams and Redundancy Profile

In this part, the third step of the methodology is described. It aims to generate a DFT from a system model using the redundancy information from redundancy profile.

5.1.1. Fault Trees

A fault tree is composed of gates and events and it can be static or dynamic. The static fault tree is used when the order of the failure events is not important. Yet, the dynamic fault tree is used when the order of the failure events has an impact on the top event. A fault tree is composed by at least one event and we define three different types of events: the top undesired event, the intermediate events, and the leaf events. In addition, two types of logic gates are defined which are the static and dynamic gates and they aim to describe the propagation of failures leading to the top event. The static gates like "AND", "OR", and "VOTING" are used in the construction of the SFTs. The dynamic gates [32] such as "SPARE", "PAND", "FDEP", and "SEQUENCE" are used in addition to the static gates in the construction of the DFTs. In this work, we use the dynamic fault trees in order to model redundancy obtained in step two.

5.1.2. Redundancy Terminology

In the world of engineering, redundancy is known as the duplication of components or functions of a system to enhance its availability and/or reliability [31]. In general, there are three types of active redundancy strategies: active, standby, and mixed redundancy. The active redundancy is used when all redundant components are continuously powered together from time zero and the load is shared. Standby redundancy is used when redundant components do not start operating until the primary component fails. Finally, mixed redundancy is the combination between active and standby redundancy. It is used to improve the reliability of the system [33].

5.1.3. Redundancy Profile

To model the behavior of the system and the redundancy, additional information is needed. For this reason, we create an UML extension called "Redundancy Profile" in order to integrate redundancy information into SysML models. Figure 3 represents the created profile diagram.

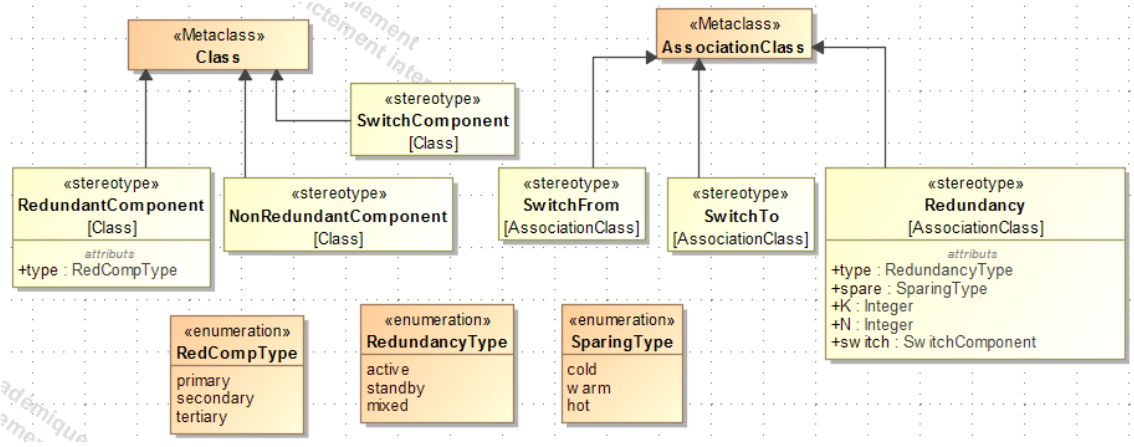

Figure 3. Profile diagram.

The redundancy profile decomposes components in three different types. These components extend "Class" meta-class of UML and represent extensions of SysML blocks. The first category is the Redundant Component that describes duplicated and linked components. The second category is the Non Redundant Component, which describes the non duplicated components that do not have a redundancy link. Finally, the Switch Component is used in the case of standby or mixed redundancy

and it describes components that allow switching from the main component to the standby component. In addition, three additional stereotypes will extend the "Association Class" meta-class of UML. The redundancy association class aims at defining the relation between two redundant components. Hence, the redundancy type is specified and it can be active, standby, or mixed. Then, the sparing type will be defined and it can be cold, warm, or hot. Furthermore, "Switch From" and "Switch To" are used to define the activation order of the different redundant components. In the case of standby and mixed redundancy, some additional information like the type of redundant component (primary, secondary, etc.) are used.

5.1.4. Description of the DFT Generation Method

In this part, the description of the generation of DFT from system model enriched with redundancy information is given. After updating the architecture of the system by analyzing FMEA information, the specification of the system behavior became a necessity. Considering that the solution adopted in this work is the redundancy, the generated DFT will consider, especially, the behavior of the redundant components. The method that describes this step is given. The input of this step is the internal structure of the system that provides the system components and the interaction between each other. This information will be provided by the IBD. An IBD is composed of three different parts which are the external ports, the internal ports, and the components of the system part. The external ports represent the external interfaces of the system with its environment. The internal ports are ports through which the system components interact with each other. Also, the external inputs and outputs are distinguished depending on the port direction. In this method, the IBD is viewed as an oriented graph with a set of vertices and a set of edges. The set of vertices is composed of the system components and the external ports, which are situated on the border of the system. The second input of this step is the information about redundancy that is provided by the redundancy profile and modeled using the BDD. Another input of the method is the external output of the system that represents the top level event of the DFT. The output of the algorithm is the DFT. When the system contains more than one output, a DFT will be generated for each output.

This method begins with an in-depth first search in the IBD in order to detect all the components and all the external inputs of the system that have an impact on the output. After that, the failure will be propagated through the system via the internal ports of the components. In a second phase, all of the obtained information will be classified into three sets. The first set is the External Input (EI), which contains all external inputs of the system that are connected with "o". The second set is the Non Redundant Component (NRC), which includes all non redundant components of the system that lead to the top level event. The third set is the Redundant Component (RC), which includes groups of redundant components; a group of redundant components is formed of redundant components and switches components which are related to each other to perform the same function. The failure of a non redundant component or an external input port leads to the failure in the top level event and it is represented in the DFT by a leaf event. Thus, a subtree is generated for each element in NRC and EI. A subtree is also generated for each redundancy group in RC. As a result, information like RedundancyType (standby, active, or mixed), RedComponentType (primary, secondary), and SparingType (cold, warm, hot) is needed.

When the redundancy type is active and at least "K" out of "N" components are needed to ensure the functioning of the system, the logical gate "Voting (K/N)" is used. In this case, the system fails if more than "K" components fail.

If the redundancy type is standby, two possible cases are previewed: First, if the switch between redundant components does not need a switch component, the logical gate "SPARE" is used (it returns true only if all components fail). To specify the rank of redundant component (primary, secondary), the RedComponentType is requested. Second, if the switch between redundant components is done by a switch component, the logical gate "PAND" and "SPARE" are used. Then, a subtree is generated

as shown in Figure 4. This subtree returns true if all redundant components fail or if the switch component fails before the primary component.

If the redundancy type is mixed, then the RedComponentType is requested to specify the rank of the different components. In this case, two possible cases are previewed: First, if the switch between redundant components does not need a switch component, then a subtree that combines active redundancy subtree and standby redundancy without switch subtree is generated. Second, if the switch between redundant components is done by a switch component, then a subtree that combines active redundancy subtree and standby redundancy with switch subtree is generated.

In order to create the final DFT, all generated subtrees are gathered using OR gate.

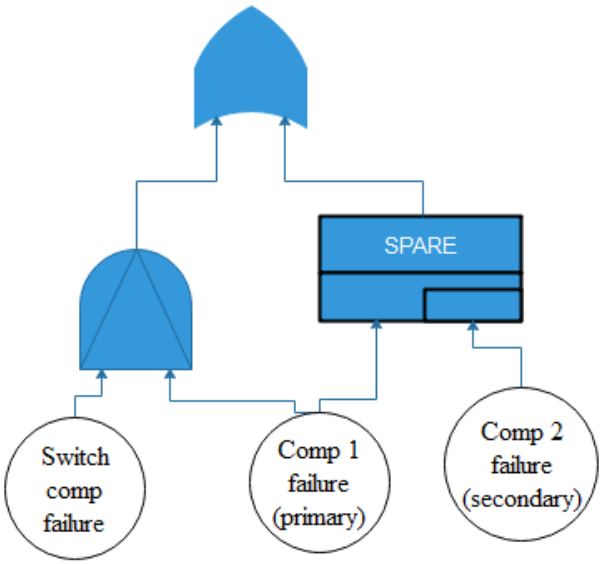

Figure 4. Subtree of a standby redundancy with switch component.

5.2. Step 4: Describe the System Behavior Using the DFT Information

The generation of the DFT is presented in Open-PSA (Probabilistic Safety Assessment) Model Exchange Format [34]. Fault tree in Open-PSA format can be analyzed with the XFTA tool [35] and it assures both the quantitative analysis, like the probabilistic assessments, and the qualitative analysis, such as the minimal cut sets and minimal cut sequences.

5.2.1. Qualitative and Quantitative Analysis

The quantitative analysis aims at computing the probability of a top level event failure. In this analysis type, two different approaches are defined, which are the Continuous-Time Quantitative Analysis (CTQA) and the Discrete-Time Quantitative Analysis (DTQA). The CTQA is an approach that considers the evolution of the failure of the system over time, while the DTQA is an approach that considers the whole lifespan of the system as a single event. The qualitative analysis aims at identifying the high-level event which is the necessary and sufficient combination of the basic events. The techniques that are used most of the times in the qualitative analysis are "minimal cut sets", "minimal cut sequences", and "common cause failures". The Cut Set (CS) is composed of component failures that cause the failure of the system. The Minimal Cut Set (MCS) is a cut set which, when any element is removed from it, what remains is no longer defined as a cut set [36]. The Minimal Cut SEQuence (MCSEQ) is a cut sequence which, when any element is removed from it, what remains is no longer a cut sequence. It causes the occurrence of the top level event of the Dynamic Fault Tree. The difference between the MCS and the MCSEQ is that the MCSEQ considers the occurrence order of the failure of the basic events.

5.2.2. Creation of the State Machine

In this subsection, the fourth step of the methodology will be described. The goal of this step is the creation of a state machine that describes the behavior of the system. Once the DFT is generated from system model, DFT will be analyzed in order to obtain the minimal cut sequences of the system. These MCSEQ are used to manually create one- or multi-state machines. The state machine starts with the normal operating state. Then, for each component failure which represents the transition, we move to the next step that shows the new behavior of the system after failure occurrence. We repeat the same method until browsing the whole minimal cut sequence which represents the breakdown of the system. The fourth step of the Improved SafeSysE aims at including all structural and behavioral information in the same SysML model. Also, it gives the possibility to use STM and IBD to make system simulations using the Cameo Systems Modeler tool [37]. It helps the system designers to better understand the failure and the dysfunction of the system.

6. Case Study

To better illustrate the methodology, as well as the contributed improvements in Improved SafeSysE, we take the same case study as SafeSysE [16] but we focus on the differences. In this section, we are validating the proposed methodology by an example of a scenario of an Electro-Mechanical Actuator (EMA) that controls an aileron system.

6.1. Step 1: The Generation of FMEA from SysML Diagrams

The first step of the case study is the same as the case study of [16]. It is composed of three components: an electronic and software part, an electric geared motor with encoder, and a mechanical transmission. Also, the input of the system are the "Electric Power" and the "CtrlData". The output of the system is the mechanical power that provides the output movement of the aileron. The structure of the system is given in Figure 5. The output of the first step is a generated FMEA that contains a list of components with their corresponding functions and functional failure modes. Also, it contains other information like the occurrence, the severity, the detectability, the RPN, and the risk class of the function failure mode. The initial FMEA is generated as represented in Table 2.

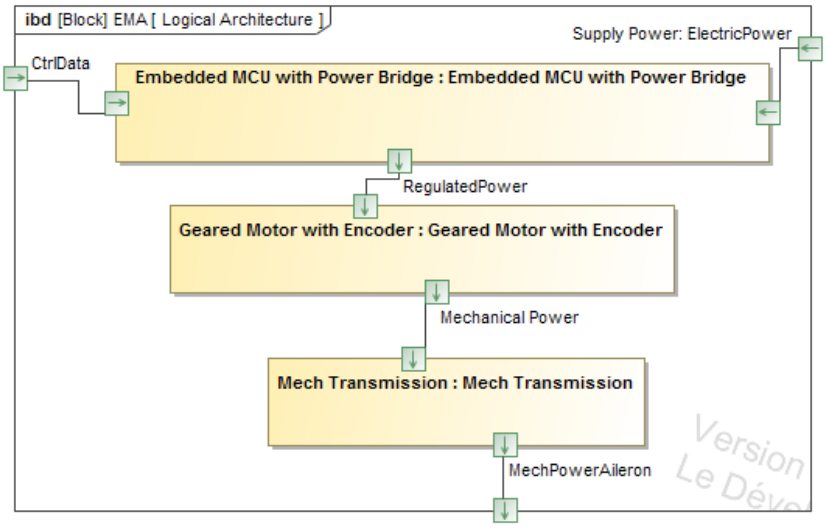

Figure 5. Initial structure of the system.

Table 2. Initial Failure Mode and Effects Analysis (FMEA) for Electro-Mechanical Actuator (EMA).

Component	Function	Function Failure Mode	Failure Mode	S	O	D	RPN	Risk Class
MCU	Regulate Electrical Energy	Fails to perform	Hardware defect	8	7	4	224	II
		perform incorrectly	Hardware defect	6	7	4	168	II
	Translate Pilot Instructions	Fails to perform	Software Error	8	4	6	192	III
			Hardware defect	9	7	7	441	I
			Synchronization Error	10	5	6	300	I
			Memory defect	7	4	5	140	III
		Performs incorrectly	Software fault	6	7	4	168	II
		Performs incorrectly	Synchronization Error	5	5	6	150	III
Geared Motor with encoder	Adapt Mechanical Energy	Fails to perform	Loss of structural integrity	10	6	4	240	I
		Fails to perform	Jamming	9	4	8	288	II
		perform incorrectly	Poor efficiency	2	8	3	48	II
	Measure Incidence	Fails to perform	Mechanical drive defect	7	4	5	140	III
			Power Loss	7	8	3	168	I
			Internal component failure	6	5	8	210	III
		Perform incorrectly	Mechanical drive defect	5	7	6	210	II
			Low or high voltage	5	3	7	105	IV
			Measure fault	4	7	5	140	III
	Transform Mechanical Energy	Fails to perform	Jamming	6	7	8	336	I
			Loss of structural integrity	10	3	5	150	II
			Short-circuit between two winding	8	5	4	160	III
		Perform incorrectly	Short-circuit in one winding	7	4	6	168	III
Mechanical Transmission	Transmit Mechanical Energy	Fails to perform	Loss of structural integrity	9	3	5	135	III
			Jamming	6	5	3	90	III
		Perform incorrectly	Poor efficiency	5	6	5	150	III

6.2. Step 2: Making Architecture Decision Using FMEA Recommendations

In this step, the generated FMEA is computed and the most critical failure modes are detected. The most critical failure modes are the failure modes that have class I as risk class. In this example, five different function failure modes are detected, which are 1/fails to perform the translation of pilot instructions because of an hardware defect of the MCU component, 2/fails to perform the translation of pilot instructions because of a synchronization error of the MCU component, 3/fails to perform the adaptation of the mechanical energy because of loss of structural integrity of the geared motor with encoder, 4/fails to perform the measure incidence because of a power loss of the geared motor with encoder, and 5/fails to perform the transformation of the mechanical energy because of a jamming of the geared motor with encoder.

To enhance the system architecture, the two critical components should be dubbed and the three concerned diagrams (requirement diagram, activity diagrams, and IBD) will be updated. The new IBD will contain two redundant MCUs with a switch component that permits the switching from the main component to the spare component, and two redundant geared motors with a switch component as shown in Figure 6.

Then, the new FMEA should be generated to verify the safety level of system 3. It contains components, funtions, and failure modes with new S, O, D, RPN, and risk class values. In addition, it includes the new components such as "Switch EMCU" and "Switch Motor" with their functions, failure modes, and its criticality. If the class risk of all function failure modes are different to class I, then the system structure is validated and it is time to move to the next step. Otherwise, a new iteration should be done and new structural modifications will be recommended. In our case and as shown in Table 3, all function failure modes have a risk class different to class I. The new system structure is validated and we move to the third step.

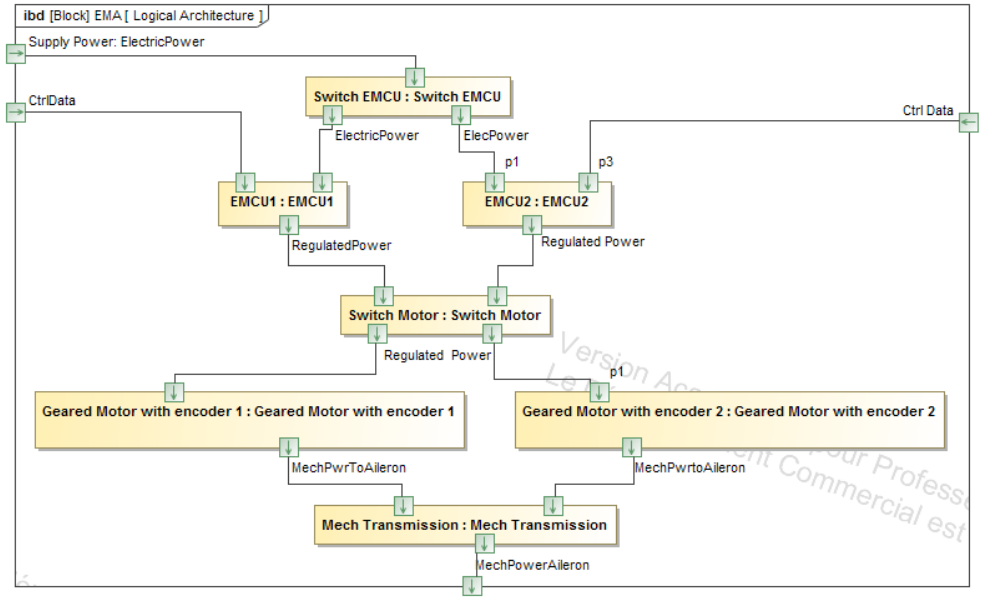

Figure 6. Updated structure of the system.

Table 3. New FMEA for EMA.

Component	Function	Function Failure Mode	Failure Mode	S	O	D	RPN	Risk Class
EMCU	Regulate Electrical Energy	Fails to perform	Hardware defect	8	4	4	128	III
		perform incorrectly	Hardware defect	6	4	4	96	III
	Translate Pilot Instructions	Fails to perform	Software Error	8	1	6	48	IV
			Hardware defect	9	2	7	126	III
			Synchronization Error	10	2	6	120	III
			Memory defect	7	1	5	35	IV
		Performs incorrectly	Software fault	6	2	4	48	III
		Performs incorrectly	Synchronization Error	5	3	6	90	III
Switch EMCU	Transmit Electric Energy	Fails to perform	Hardware defect	9	2	5	90	III
	Switch-over the Electric Energy	Fails to perform	Hardware defect	7	3	5	105	III
Geared Motor with encoder	Adapt Mechanical Energy	Fails to perform	Loss of structural integrity	10	3	4	120	II
		Fails to perform	Jamming	9	2	8	144	III
		perform incorrectly	Poor efficiency	2	4	3	24	IV
	Measure Incidence	Fails to perform	Mechanical drive defect	7	2	5	70	III
			Power Loss	7	4	3	84	III
			Internal component failure	6	3	8	144	III
		Perform incorrectly	Mechanical drive defect	5	4	6	120	III
			Low or high voltage	5	2	7	70	IV
			Measure fault	4	4	5	80	III
	Transform Mechanical Energy	Fails to perform	Jamming	6	4	8	192	III
			Loss of structural integrity	10	2	5	100	III
			Short-circuit between two winding	8	3	4	96	III
		Perform incorrectly	Short-circuit in one winding	7	2	6	84	III
Switch Motor	Transmit Electric Energy	Fails to perform	Hardware defect	9	2	5	90	III
	Switch-over the Electric Energy	Fails to perform	Hardware defect	7	3	5	105	III
Mechanical Transmission	Transmit Mechanical Energy	Fails to perform	Loss of structural integrity	9	3	5	135	III
			Jamming	6	5	3	90	III
		Perform incorrectly	Poor efficiency	5	6	5	150	III

6.3. Step 3: DFT Generation from System Model

In the third step, and since we have redundancy in the system, a BDD is created using the redundancy profile in order to describe the relation between redundant components and to facilitate the generation of the DFT. The new (updated) system contains two groups of redundant components with switch. The BDD describes, both, the redundancy of the geared motor and the redundancy of the MCU. Figure 7 describes the redundancy between the two redundant groups. "MCU1", "MCU2", "Geared motor 1", and "Geared motor 2" are redundant components. "MCU1" and "MCU2" are redundant between each other and the redundancy type is standby. "MCU1" is primary and "MCU2" is secondary. The "Switch MCU" is a switch component that allows the switching from "MCU1" to "MCU2" when failure of "MCU1" occurs. In the same way as MCU, "Geared motor 1" and "Geared motor 2" are redundant between each other and the redundancy type is standby. "Geared motor 1" is primary and "Geared motor 2" is secondary. The "Switch Motor" is a switch component that allows the switching from "Geared motor 1" to "Geared motor 2" when failure of "Geared motor 1" occurs.

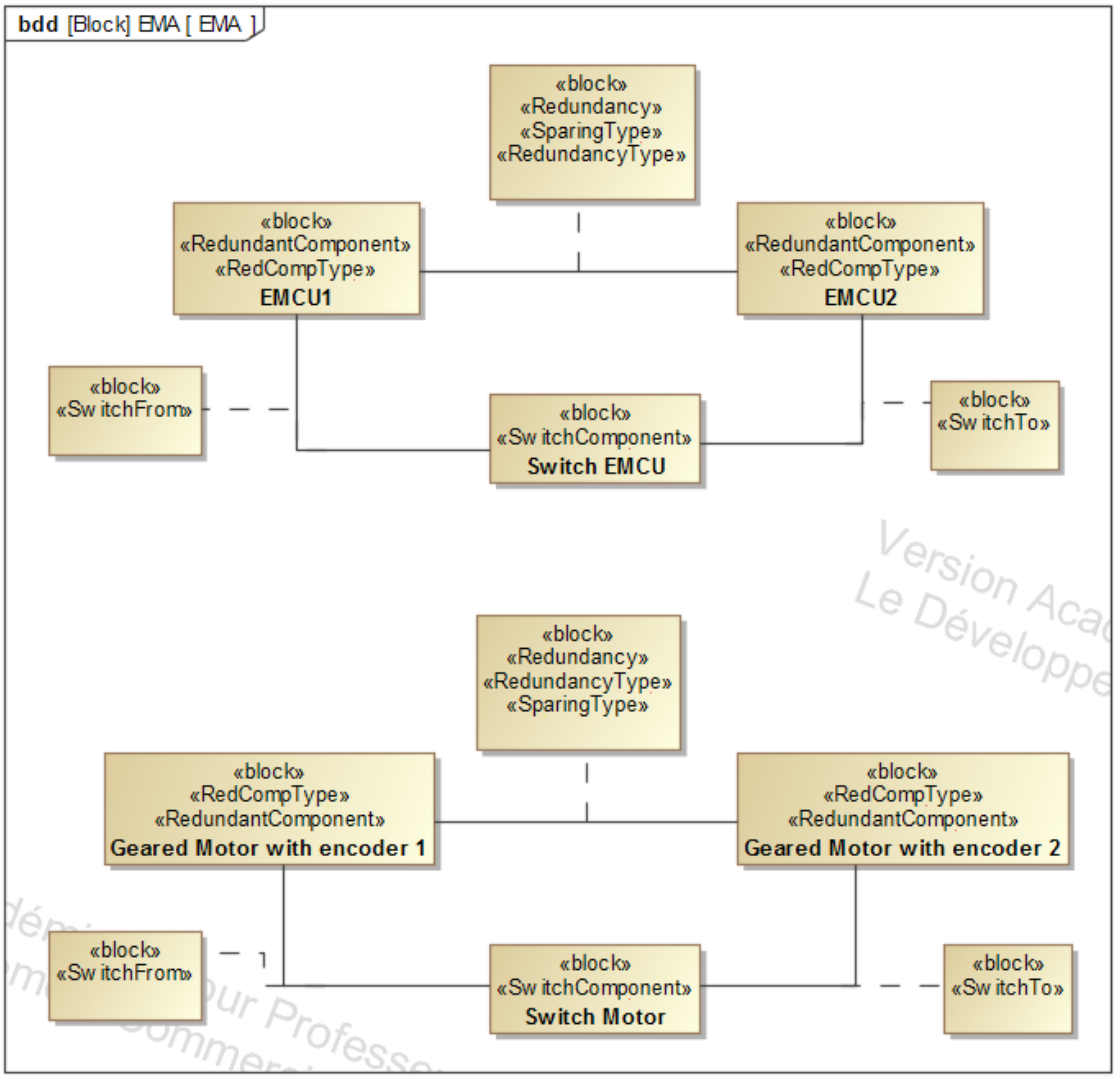

Figure 7. Block Definition Diagram (BDD) of Electro-Mechanical Actuator (EMA) system with redundancy information.

Then, the algorithm of the dynamic fault tree generation uses the structure of the system (IBD) represented in Figure 6 and the redundancy information (BDD) represented in Figure 7. Firstly, the error of the Output port is taken as an undesired event. Secondly, the depth first search through

the structure of the system is applied in order to determine the external inputs of the system. In the EMA example, two external inputs are reached which are the "CtrlData" and the "Electric Power". For each external input, a leaf failure event is generated. Also, the non redundant component errors are represented by a leaf failure event in the DFT. In this example, just the "Mechanical Transmission" component is a non redundant component. The two redundancy groups are standby redundancy with switch component, then a subtree of a standby redundancy with switch component is generated as shown in Figure 4. Knowing that the general distribution of the components is in series, all generated subtree are directly related to the main OR gate. The output of the algorithm is a DFT as shown in Figure 8 and it provides all possible failure combinations.

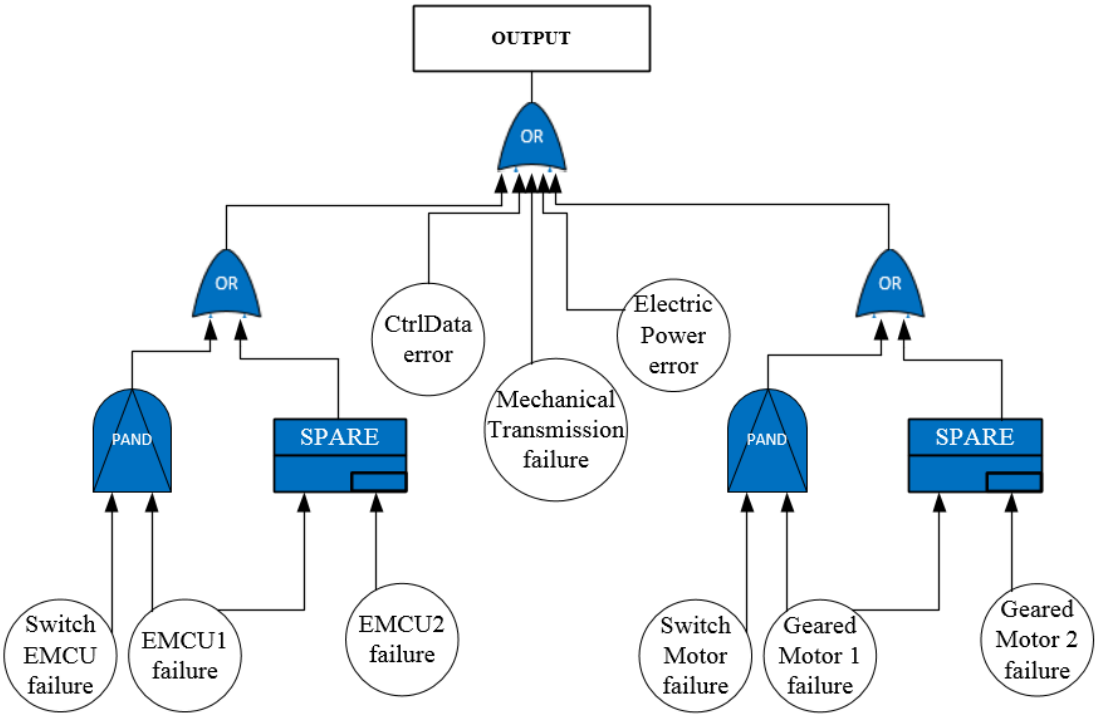

Figure 8. Dynamic Fault Tree (DFT) of the EMA system.

6.4. Step 4: Creation of the State Machine

In the fourth step, the DFT should be analyzed in a qualitative way. The result of the analysis is a set of minimal cut sequences. In this example, nine minimal cut sequences are detected, which are:

* EMCU1 failure—EMCU2 failure
* EMCU2 failure—EMCU1 failure
* Switch EMCU failure—EMCU1 failure
* Geared motor 1 failure—Geared motor 2 failure
* Geared motor 2 failure—Geared motor 1 failure
* Switch motor failure—Geared motor 1 failure
* CtrlData error
* Electric power error
* Mechanical transmission failure

Then, a state machine that describes the behavior of the system is created based on the minimal cut sequences. In each state, the behaviors of all components are described. When the failure occurs, the new behavior of the system is described in the next state. This iteration is repeated until the breakdown of the system (the end of the minimal cut sequence). The result of this work is shown in Figure 9. The minimal cut sequences with only one element are not considered in this work. Finally,

the state machine diagram can be used with the IBD to simulate the behavior of the system with a simulator.

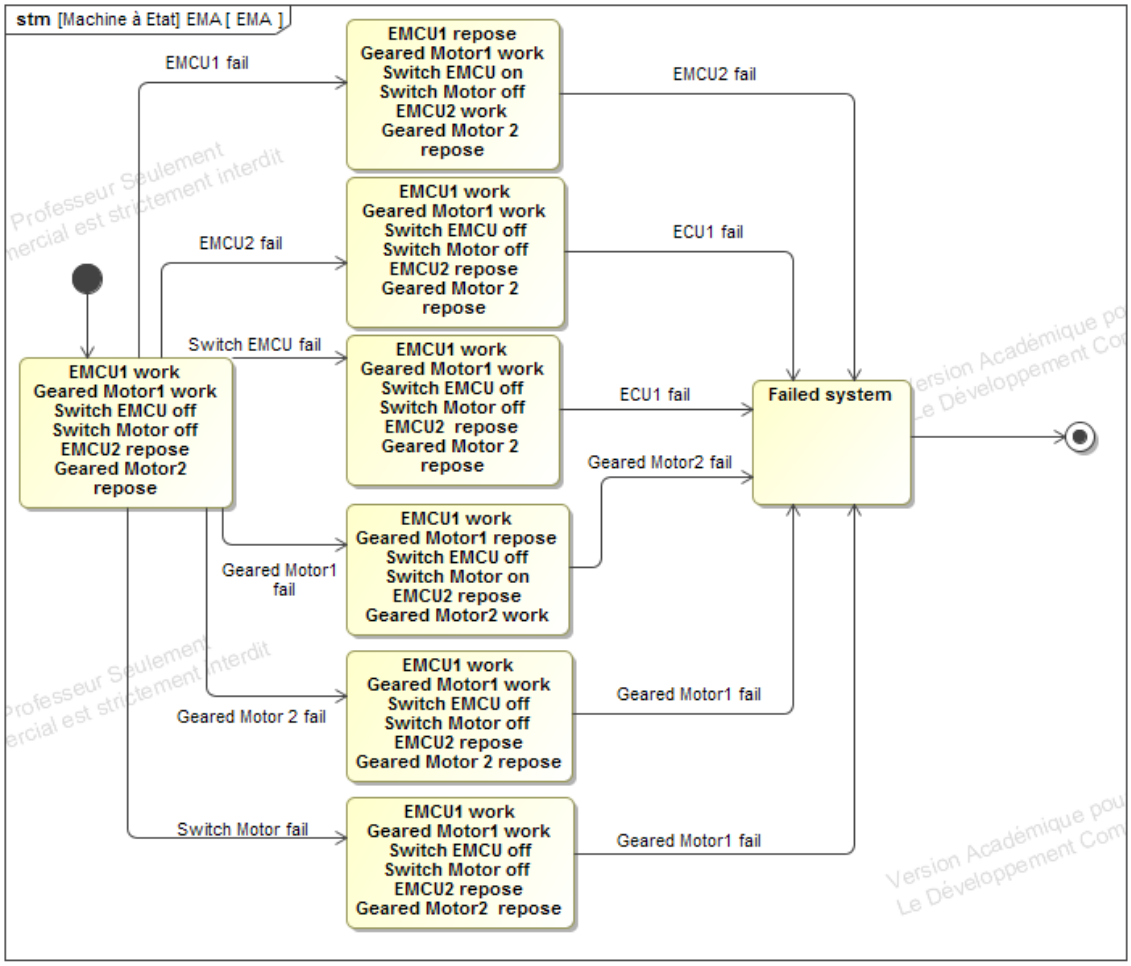

Figure 9. State machine of the EMA system.

7. Conclusions

Nowadays, the safety of mechatronic systems represents a real challenge for safety experts, systems engineers, and designers, especially because of the increasing complexity of the systems. The main contribution of this work is to narrow the gap between MBSE and MBSA by integrating safety information in the systems engineering process. To reach this objective, a methodology which is composed of four steps is developed. The first two steps aim at improving the system's structure by proposing structural modifications. The structural modifications are deduced from the analysed information in the FMEA generated from the system description. Once the system's structure is updated, the system's behavior should be specified. For this reason, a redundancy profile that describes the behavior of the redundancy in the system is created. The redundancy profile is used with the system description to automatically generate a DFT via a generation algorithm. In the final step, the generated DFT is analyzed and a set of minimal cut sequences is obtained. Then, a state machine diagram is created using the obtained minimal cut sequences. This state machine is the first and only diagram, in this work, that describes the system's behavior in the system's model. The methodology is applied to an Electro-Mechanical Actuator case study, step by step, in order to validate the approach and to show the continuous improvement of system structure and behavior, thanks to preliminary safety analysis results.

This work presents many advantages such as the increase of the competitiveness and the reduction of the safety analysis cost and time. In addition, it aims at enhancing the consistency between systems

engineering models and safety artifacts by creating a link between them. However, the proposed methodology has some limits. For example, adding redundancy is not always an optimal solution with regard to the additional weight and cost, and the proposed methodology does not provide a comparison between a high reliability component and a redundant architecture. Also, during the generation of the DFT from the system model, the proposed algorithm cannot consider the redundancy of the external input of the system. For these reasons, in future works, the consistency between the system model and the safety analysis artifacts should be enhanced by proposing structural solutions other than redundancy. In addition, the redundancy profile should be improved to also consider the redundancy of the external inputs of the system. Finally, the result of the DFT analysis will be used, not only for the description of the system behavior but also to allow the updating of the system's model.

Finally, we would like to thank the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions.

Author Contributions: Conceived the idea, A.B., N.N., F.M., J.-Y.C. and A.M.; carried out the analytical derivations and numerical examples, A.B.; writing—original draft preparation, A.B.; writing—review and editing, A.B., N.N. and F.M.; Supervision, A.M. and J.-Y.C.

Funding: : This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ericson, C.A., II. *Hazard Analysis Techniques for System Safety*; Wiley: New York, NY, USA, 2005; pp. 1–528.
2. Balz, E.; Goll, J. Use Case-Based Fault Tree Analysis of Safety-Related Embedded Systems. In Proceedings of the Software Engineering and Applications, Innsbruck, Austria, 15–17 February 2005.
3. The National Aeronautics and Space Administration. *Fault Tree Handbook with Aerospace Applications, Version 1.1*; NASA: Pasadena, CA, USA, 2002.
4. Haskins, C. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*; Wiley: New York, NY, USA, 2006.
5. Cressent, R.; David, P.; Idasiak, V.; Kratz, F. Designing the database for a reliability aware Model-Based System Engineering process. *Reliab. Eng. Syst. Saf.* **2013**, *111*, 171–182. [CrossRef]
6. Cressent, R.; Idasiak, V.; Kratz, F.; David, P. Dependability Analysis Activities Merged with System Engineering, a Real Case Study Feedback. In *Advances in Safety, Reliability and Risk Management*; CRC Press: Boca Raton, FL, USA, 2012; pp. 2003–2010.
7. Laleau, R.; Semmak, F.; Matoussi, A.; Petit, D.; Hammad, A.; Tatibouet, B. A First Attempt to Combine SysML Requirements Diagrams and B. *Innov. Syst. Softw. Eng.* **2010**, *6*, 47–54. [CrossRef]
8. Baklouti, A.; Mhenni, F.; Nguyen, N.; Choley, J.Y.; Mlika, A. Improved System Architecture and Behavior Based on FMEA Recommendations. In Proceedings of the European Safety and RELiability Conference, Portoroz, Slovenia, 18–22 June 2017.
9. Garro, A.; Tundis, A. On the Reliability Analysis of Systems and SoS: The RAMSAS Method and Related Extension. *IEEE Syst. J.* **2015**, *9*, 232–241. [CrossRef]
10. Mhenni, F.; Choley, J.Y.; Penas, O.; Plateaux, R.; Hammadi, M. A SysML-Based Methodology for Mechatronic Systems Architectural Design. *Adv. Eng. Inform.* **2014**, *28*, 218–231. [CrossRef]
11. Mhenni, F.; Choley, J.Y.; Nguyen, N. An Integrated Design Methodology for Safety Critical Systems. In Proceedings of the 2016 Annual IEEE Systems Conference (SysCon), Orlando, FL, USA, 18–21 April 2016; pp. 1–6.
12. Helle, P. Automatic SysML-based Safety Analysis. In Proceedings of the Fifth International Workshop on Model Based Architecting and Construction of Embedded Systems, New York, NY, USA, 20–22 September 2012; pp. 1–6.
13. Bozzano, M.; Papadopoulos, Y. Model-Based Safety and Assessment. In Proceeding of the 5th International Symposium, IMBSA, Trento, Italy, 11–13 September 2017.
14. Frederic, T.; Belmonte, F. Performing Safety Analyses and SysML Designs Conjointly: A Viewpoint Matter. In Proceeding of the Complex Systems Design & Management, Paris, France, 7–9 December 2011.

15. Nguyen, N.; Mhenni, F.; Choley, J.Y. Redundancy Handling with Model-Based Systems Engineering. In Proceeding of the 26th European Safety and Reliability Conference (ESREL 2016), Glasgow, Scotland, 25–29 September 2016.
16. Mhenni, F.; Nguyen, N.; Choley, J.Y. SafeSysE: A Safety Analysis Integration in Systems Engineering Approach. *IEEE Syst. J.* **2018**, *12*, 161–172. [CrossRef]
17. David, P.; Idasiak, V.; Kratz, F. Reliability study of complex physical systems using SysML. *Reliab. Eng. Syst. Saf.* **2010**, *95*, 431–450. [CrossRef]
18. Arnold, A.; Griffault, G.; Point, G.; Rauzy, A. The AltaRica Language and its Semantics. *Fundam. Inform.* **2000**, *34*, 109–124.
19. Papadopoulos, Y.; Maruhn, M. Model-based synthesis of fault trees from Matlab—Simulink models. In Proceedings of the International Conference on Dependable Systems and Networks, Göteborg, Sweden, 1–4 July 2001; pp. 77–82.
20. Rauzy, A. Mode Automata and Their Compilation Into Fault Trees. *Reliab. Eng. Syst. Saf.* **2002**, *78*, 1–12. [CrossRef]
21. Joshi, A.; Vestal, S.; Binns, P. Automatic Generation of Static Fault Trees from AADL Models. In Proceedings of the DSN Workshop on Architecting Dependable Systems, Edinburgh International Conference Centre, Edinburgh, UK, 25–28 June 2007.
22. Yakymets, N.; Jaber, H.; Lanusse, A. Model-Based System Engineering for Fault Tree Generation and Analysis. In Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Funchal, Portugal, 22–24 January 2013.
23. Li, S.; Li, X. Study on Generation of Fault Trees from AltaRica Models. *Procedia Eng.* **2014**, *80*, 140–152. [CrossRef]
24. Hofig, K.; Zeller, M.; Heilmann, R. ALFRED: A Methodology to Enable Component Fault Trees for Layered Architectures. In Proceedings of the 41st Euromicro Conference on Software Engineering and Advanced Applications, Funchal, Portugal, 26–28 August 2015; pp. 167–176.
25. Tajarrod, F.; Latif-Shabgahi, G. A Novel Methodology for Synthesis of Fault Trees from MATLAB-Simulink Model. *World Acad. Sci. Eng. Technol.* **2008**, *41*, 630–636.
26. Dehlinger, J.; Dugan, J.B. Analyzing Dynamic Fault Trees Derived From Model-Based System Architectures. *Nucl. Eng. Technol.* **2008**, *40*, 365–374. [CrossRef]
27. Pai, G.; Dugan, J. Automatic Synthesis of Dynamic Fault Trees From UML System Models. In Proceedings of the 13th International Symposium on Software Reliability Engineering, Annapolis, MD, USA, 12–15 November 2002; pp. 243–254.
28. Mahmud, N.; Papadopoulos, Y.; Walker, M. A translation of state machines to temporal fault trees. In Proceedings of the International Conference on Dependable Systems and Networks, Chicago, IL, USA, 28 June–1 July 2010; pp. 45–51.
29. Zhao, Z.; Petriu, D. UML Model to Fault Tree Model Transformation for Dependability Analysis. In Proceedings of the International Conference on Computer and Information Science and Technology, Ottawa, ON, Canada, 11–12 May 2015; pp. 1–9.
30. Xiao, N.; Huang, H.Z.; Li, Y.; He, L.; Jin, T. Multiple failure modes analysis and weighted risk priority number evaluation in FMEA. *Eng. Fail. Anal.* **2011**, *18*, 1162–1170. [CrossRef]
31. The International Electrotechnical Commission (IEC). *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*; Parts 1–7, IEC 61508; IEC: Geneva, Switzerland, 1998.
32. Delić, M.; Ilić, S.; Glišović, J.; Ćatić, D. Dynamic Fault Tree Analysis of Lawnmower. In Proceedings of the 9th International Quality Conference, Faculty of Engineering, Kragujevac, Serbia, 5 June 2015; University of Kragujevac: Kragujevac, Serbia, 2015.
33. Abouei Ardakan, M.; Zeinal Hamadani, A. Reliability Optimization of Series-Parallel Systems with Mixed Redundancy Strategy in Subsystems. *Reliab. Eng. Syst. Saf.* **2014**, *130*, 132–139. [CrossRef]
34. Epstein, S.; Rauzy, A. *Open-PSA Model Exchange Format*; PSA: Clamart, France, 2017.
35. Rauzy, A. *XFTA: An Open-PSA Fault Tree Engine*; AltaRica Association: Toulouse, France, 2015.

36. Ruijters, E.; Stoelinga, M. Fault Tree Analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Comput. Sci. Rev.* **2015**, *15–16*, 29–62. [CrossRef]
37. Magic, N. *Cameo Systems Modeler, USER GUIDE 18.1*; No Magic, Inc.: Allen, TX, USA, 2015.