POLITECNICO DI TORINO

Department of Mechanical and Aerospace Engineering

Master’s Degree Thesis

A Model-Based RAMS Estimation Methodology for Innovative Aircraft on-board Systems developed in a MDO Environment

Candidate	Bruno Francesco
Academic Tutor	Prof. Fioriti Marco
Academic Tutor	Prof. Corpino Sabrina
Company Tutor	Boggero Luca
Company Tutor	Donelli Giuseppa

July 2020

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ACE	=	Actuator Control Electronics
AEA	=	All Electric Aircraft
ASC	=	Annual Support Cost
BDD	=	Block Definition Diagram
CA	=	Criticality Analysis
CAI	=	Cowl Anti-Icing
CCA	=	Common-Cause Analysis
CDF	=	Cumulative Distribution Function
CPACS	=	Common Parametric Aircraft Configuration Schema
DFMEA	=	Design Failure Mode and Effect Analysis
DLR	=	Deutsches Zentrum für Luft- und Raumfahrt
ECS	=	Environmental Control System
EHA	=	Electro-Hydrostatic Actuator
ELAC	=	Elevator Aileron Computer
EMA	=	Electro-Mechanical Actuator
ETBA	=	Energy Trace and Barrier Analysis
FAC	=	Flight Augmentation Computer
FCC	=	Flight Control Computer
FCDC	=	Flight Control Data Concentrator
FCS	=	Flight Control System
FFBD	=	Functional Flow Block Diagram
FH	=	Flight Hours
FHA	=	Functional Hazard Analysis
FMEA	=	Failure Mode and Effects Analysis
FMECA	=	Failure Mode, Effect and Criticality Analysis
FTA	=	Fault Tree Analysis
HAZOP	=	Hazard and Operability study
HTS	=	Hazard Tracking System
IBD	=	Internal Block Diagram
ISA	=	International Standard Atmosphere
MA	=	Markov Analysis
MaxTTR	=	Maximum-Time-to-Repair
MBSA	=	Model-Based Safety Assessment

MBSE	=	Model-Based Systems Engineering
MDO	=	Multidisciplinary Design Optimization
MEA	=	More Electric Aircraft
MLW	=	Maximum Landing Weight
MR	=	Maintenance Ratio
MTBF	=	Mean Time Between Failures
MTBMA	=	Mean-Time-Between-Maintenance-Actions
MTOW	=	Maximum Take-off Weight
MTTF	=	Mean Time to Failure
MTTR	=	Mean Time to Repair
MZFW	=	Maximum Zero Fuel Weight
OEW	=	Operative Empty Weight
PDF	=	Probability Density Function
PFMEA	=	Process Failure Mode and Effect Analysis
PoF	=	Physics of Failure analysis
PSSA	=	Preliminary System Safety Analysis
RAC	=	Risk Assessment Code
RAMS	=	Reliability, Availability, Maintainability and Safety
RBD	=	Reliability Block Diagram
RPN	=	Risk Priority Number
SCF	=	Safety-Critical Function
SCI	=	Safety-Critical Item
SE	=	Systems Engineering
SEC	=	Spoilers Elevator Computer
SFMEA	=	System Failure Mode and Effect Analysis
SRF	=	Safety-Related Function
SRI	=	Safety-Related Item
SSA	=	System Safety Analysis
SSF	=	Safety-Significant Function
SSI	=	Safety-Significant Item
SysML	=	System Modelling Language
THS	=	Trimmable Horizontal Stabilizer
WAI	=	Wing Anti-Icing
XML	=	eXstensible Markup Language
ZSA	=	Zonal Safety Analysis

1. Introduction

The present document is intended to describe the thesis work carried out in cooperation with the German Aerospace Center (DLR). The purpose of this research work was the development of a methodology for the evaluation of innovative on-board systems dependability, focused on a model-based approach. The on-board systems under analysis were supposed to be also integrated into a multidisciplinary environment. Therefore, another aim was to evaluate how the on-board systems dependability affected the overall aircraft performance ad costs.

The thesis starts with the explanation of some topics essential for its comprehension (such as the different aircraft architectures characteristics, an overview of the multidisciplinary design, etc.). Then it continues with an in-depth analysis of the state of the art concerning the topics covered. Afterwards, it continues with a description of the developed methodology and finally, it illustrates the obtained results.

1.1. Aircraft Architectures

The reduction of aircraft fuel consumption is increasingly becoming one of the most important objectives addressed by aeronautical manufacturer. Different new technologies and solutions are being developed (e.g. more efficient engines, alternatives to kerosene-based fuels, hybrid electric aircraft) with the aim of achieving a lower environmental impact and lower fuel costs. The recent trend, for aerospace companies and research centers, is to develop aircraft capable of using electrical power as principal source of power, to increase their performance and to reduce their fuel consumption, noise, and air pollutant emissions. In addition, are being investigated solutions to make aircraft safer and more reliable, capable of also reducing maintenance costs.

A significant improvement of operating costs can be reached by acting on on-board systems. Nowadays, especially in conventional civil airplanes, they are driven from a combination of four types of secondary power source: pneumatic, mechanical, hydraulic and electrical [1]. They are all derived from the gas turbine engines and their energy consumption is approximately 5% of the total fuel burnt [2]. The pneumatic power is obtained from the engines' high-pressure compressors and delivered to the Environmental Control System (ECS) and the Wing Anti-Icing (WAI) and Cowl Anti-Icing (CAI) systems. The mechanical power is instead transferred to hydraulic pumps, some fuel pumps, and to the main electrical generator, by means of gearboxes. Afterwards, hydraulic and electrical power are distributed throughout the aircraft to drive subsystems such as flight control actuators, landing gear, avionics, aircraft lighting and galley loads [3], [4]. Supplying all these kinds of secondary power sources requires many complex systems and a failure in one of them may lead to unavailability of important subsystems, resulting in a grounded aircraft and flight delay. Having more than one power source to be distributed throughout the aircraft, the number of redundancies to obtain the necessary safety level is higher. Moreover, power off-takes – especially bleed air off-takes – cause a reduction of engine efficiency, resulting in an increase of fuel consumption and air pollutant emissions. Therefore, the aim for the next years is to develop innovative on-board system architectures, capable of using the electrical power instead of most of the others [5]; after that, the goal for future aircraft is to replace every kind of power source with the electrical one. The first concept characterizes the More Electric Aircraft (MEA), whereas the second defines the All Electric Aircraft (AEA). A comparison between the Conventional on-board systems architecture and More Electric is represented in Fig. 1.

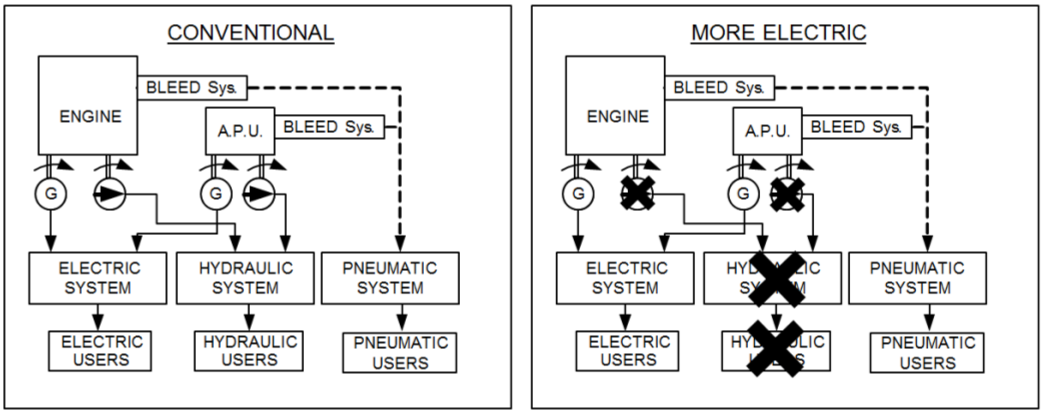


Fig. 1: Comparison between Conventional and More Electric architectures [6][7]

In the last decade, the MEA concept has already been adopted by Boeing with the B787 Dreamliner, in which the no-bleed systems architecture allows to eliminate the traditional pneumatic system. Therefore, the power source of most functions (such as the air-conditioning and the WAI) is converted to electric power. This new architecture offers a number of benefits, including: improved fuel consumption (with a predicted fuel saving of about 3%), reduced maintenance costs and improved reliability, due to the use of modern power electronics and fewer components in the engine installation [8]. The same has been done by Airbus with the A380 Flight Control System (FCS), in which one of the hydraulic systems has been replaced with a set of electrically powered actuators; the type of actuators that has been selected is the Electro-Hydrostatic Actuator (EHA). The reduction of the total number of hydraulic components in the FCS architecture has involved different benefits, including improvements of reliability, weight savings and increased safety [9].

1.2. Multidisciplinary Design Optimization

The development of different new on-board system architectures may influence many disciplines and parameters (e.g. aerodynamic performance, fuel consumption, aircraft geometry, engine efficiency and costs). Therefore, it should be done through a Multidisciplinary Design Optimization (MDO) approach. An MDO is a field of engineering that focuses on numerical optimization for the design of systems [10]. It uses optimization methods to solve design problems, allowing incorporating all relevant disciplines simultaneously. Among these, the Reliability, Availability, Maintainability, and Safety (RAMS) discipline is one of the most important for the development of any on-board systems architecture.

1.3. RAMS

The acronym RAMS refers to an engineering discipline that integrates different analyses aimed at defining systems Reliability, Availability, Maintainability and Safety. Those attributes are essential features of all the engineering products and processes. Therefore, they must be always taken in account, especially during the design phases of an aircraft. In the next sections they will be thoroughly discussed to highlight their peculiarities and the role that each one of them plays in RAMS discipline.

1.3.1. Reliability

Reliability is “the probability that an item will perform its intended function for a specific interval under stated conditions” [11]. However, this definition does not consider the effect of the age of the system. Furthermore, considering repairable systems, it is valid only if maintenance is performed. Therefore, reliability describes quantitatively the probability that no failures arise during a given period of operation. This period can be defined as a time interval (based on clock time, operating hours, cycles, etc.) or as another kind of measurement (e.g. miles traveled). The term *failure* refers to an event that occurs when the system behavior deviates from its expected function [12]. This one, in turn, represents what the system is intended for and is described by its specification. An important parameter to take in account when analyzing reliability is the **Mean Time Between Failure (MTBF)**. It represents the expected length of time in which a system will be operational between failures. Its reciprocal is the **Failure Rate (λ)**, which is defined as the number of failures of an item per a certain measure-of-life unit (e.g. time, cycles, miles, etc.). It is a useful mathematical term that frequently appears in engineering and statistical calculations.

$$\lambda(t) = \frac{1}{MTBF} \tag{1.1}$$

Reliability can be principally divided in two different types: *Mission Reliability* and *Logistic related Reliability*. The first is the probability that the system will perform the mission essential functions under the conditions stated in the mission profile. The latter, instead, is the probability that no corrective maintenance or no schedule supply demand will occur after the completion of a mission profile. They respectively allow enhancing system effectiveness and minimizing the burden of owning and operating it.

1.3.2. Maintainability

Maintainability is "the probability that an item will be retained in, or restored to, a specific condition within a given period of time if prescribed procedures and resources are used" [11]. It is described as an inherent characteristic of design and installation; its aim is to determine the type and amount of maintenance required to retain that design in, or restore it to, a specified condition. There is a difference between maintainability and maintenance terms: the first refers to a design consideration, whereas the latter is a consequence of that design. Maintenance can be defined as all the actions necessary for retaining an item in, or restoring it to, an optimal designed condition; those actions also include diagnosis, repair and inspection. Maintenance can be categorized as:

- *Corrective*, if performed on a non-scheduled basis to restore equipment form a malfunction.
- *Preventive*, if inspection, detection and correction are systematically performed before failures occurrence or either before they develop into major defects.
- *On Condition*, if performed after estimating the condition of in-service equipment with a continuous monitoring.

The speed and ease with which maintenance can be performed depend on physical design features:

- **Accessibility**: describes how easily an item to be repaired can be reached.
- **Visibility**: describes if the item being worked on can be seen.
- **Testability**: describes if system faults can be detected and isolated.
- **Complexity**: describes how many of subsystems and parts included into the system, defining also which of them have standard or special purpose.
- **Interchangeability**: describes if a failed or malfunctioning unit can be readily replaced with an identical unit without the necessity of recalibration.

In addition to these physical design features, the frequency with which maintenance is needed also have an impact on its speed and ease. Frequency is principally affected by reliability and preventive maintenance schedule. Maintainability can be quantified by means of different mathematical indices:

- **Mean Time to Repair (MTTR)**: is the ratio between the total corrective maintenance time and the total number of corrective actions completed in a certain amount of time.
- **Maximum-Time-to-Repair (MaxTTR)**: is the maximum corrective maintenance time within which most of the corrective actions (either 90% or 95%) can be accomplished.

- **Maintenance Ratio (MR)**: is a useful measure of the relative maintenance burden and is expressed as the ratio between the total number of man-hours expended in direct labor and the number of end-item operating hours during a certain amount of time.
- **Mean-Time-Between-Maintenance-Actions (MTBMA)**: is an index frequently used in availability calculations and is expressed as the mean distribution of time intervals between the different maintenance actions.
- **Annual Support Cost (ASC)**: is the direct annual cost of maintenance personnel, repairs, and transportation for all corrective and preventive maintenance actions; it also quantifies the maintenance burden of a system.

These values can be used to support maintainability analysis and must be readily obtainable from planned testing. This allows the evaluation of candidate system architectures, logistics and maintenance practices. However, it is important to highlight that these relationships merely categorize data derived from testing.

1.3.3. Availability

Availability is "a measure of the degree to which an item is an in an operable state and can be committed at the start of the mission, when those mission is called for at a random point in time" [13]. It is an important parameter since its analysis can be used to support the establishment of both reliability and maintainability; it also allows carrying out a trade-off between these two parameters. There are different kinds of availability:

- **Inherent Availability (Ai)**: defines the availability only with respect to *Operating Time* and *Corrective Maintenance*; it is useful to describe combined reliability and maintainability characteristics or to define one in terms of the other during early conceptual design phases.
- **Operational Availability (Ao)**: defines the availability for all the time in which the equipment is intended to be operational (*Total Time*); it also takes in account operation environment factors.
- **Achieved Availability (Aa)**: defines the availability during testing and initial production testing, when system is not operating in its intended environment.

The most widely used, especially in military field, is the operational availability; its mathematical definition is described in [14] as:

$$Ao = \frac{Up\ Time}{Total\ Time} = \frac{Up\ Time}{Up\ Time + Down\ Time} \tag{1.2}$$

The *Up Time* represents the period in which an item can perform its primary functions and is sum of the *Operating Time* and *Standby Time* (in which the equipment is not operating but can be operable). *Down Time*, instead, is the opposite of *Up Time* and is sum of *Total Corrective Maintenance*, *Total Preventive Maintenance*, and *Total Administrative and Logistics Down Time* (spent waiting for parts, administrative processes, etc.). Therefore, the equation can be rewritten as follows:

$$Ao = \frac{OT + ST}{OT + ST + TCM + TPM + TALDT} \tag{1.3}$$

One problem associated with the operational availability is that it becomes costly and time-consuming to define the different necessary parameters. Nevertheless, its expression allows relating reliability and maintainability elements in one parameter.

1.3.4. Safety

Safety is "the freedom from conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment" [15]. Those conditions are defined as *hazards* and must be prevented or mitigated to provide an adequate level of safety. System Safety Engineering is an engineering discipline which employs specialized knowledge and skills to identify and eliminate hazards or to reduce the associated risks when they cannot be eliminated. A risk is defined as a combination of the severity of a mishap and the probability that it will occur. The entire System Safety process – which characterizes System Safety Engineering – is presented in Fig. 2, showing the eight elements that compose it and their logical sequence.

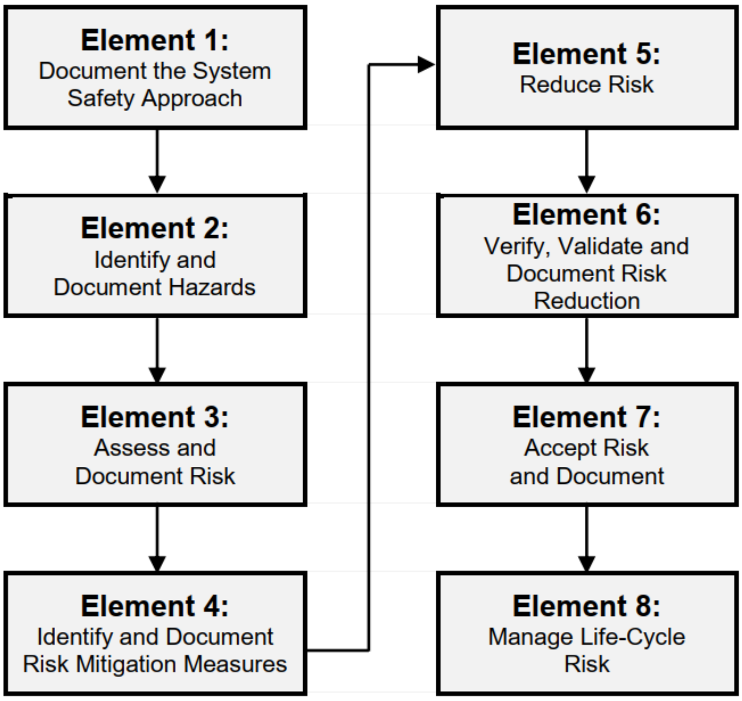
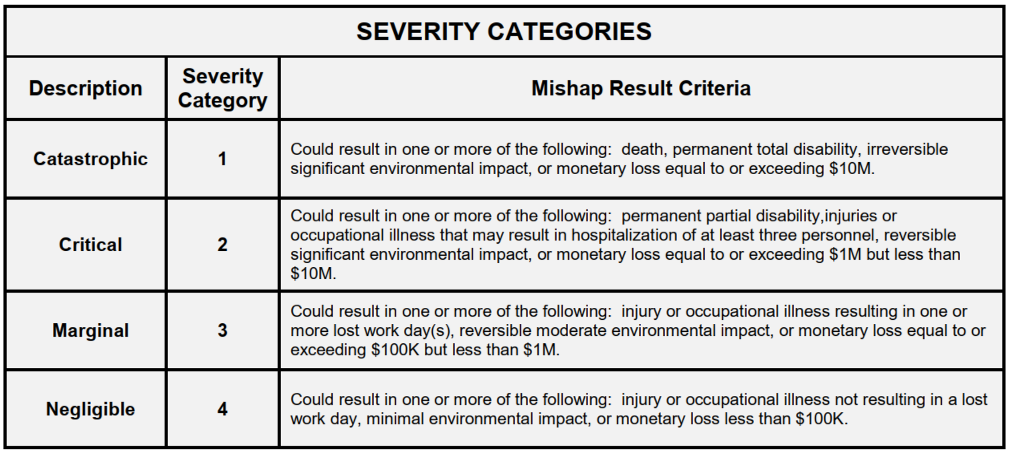
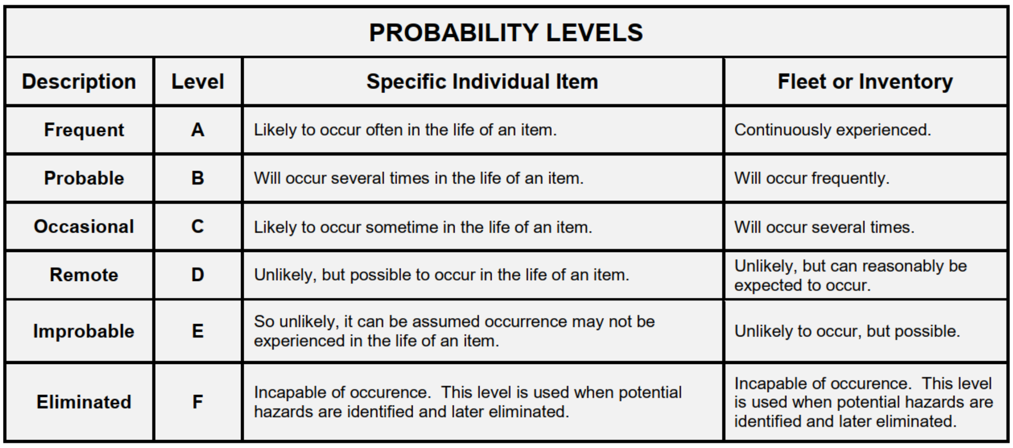

Element 1: Document the System Safety Approach
Element 2: Identify and Document Hazards
Element 3: Assess and Document Risk
Element 4: Identify and Document Risk Mitigation Measures
Element 5: Reduce Risk
Element 6: Verify, Validate and Document Risk Reduction
Element 7: Accept Risk and Document
Element 8: Manage Life-Cycle Risk

Fig. 2: Elements of System Safety process [15]



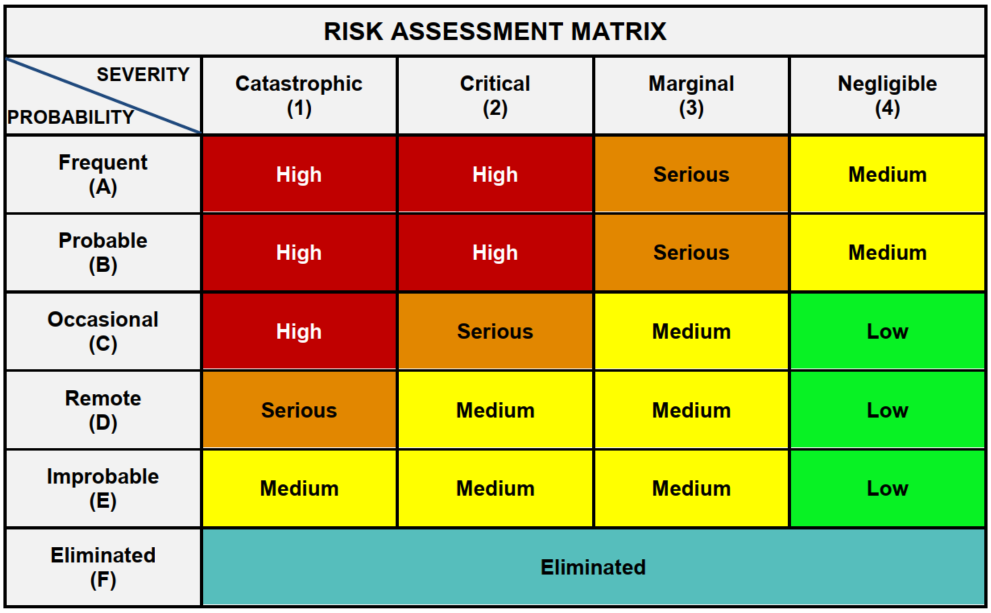
SEVERITY CATEGORIES		
Description	**Severity Category**	**Mishap Result Criteria**
Catastrophic	**1**	Could result in one or more of the following: death, permanent total disability, irreversible significant environmental impact, or monetary loss equal to or exceeding $10M.
Critical	**2**	Could result in one or more of the following: permanent partial disability,injuries or occupational illness that may result in hospitalization of at least three personnel, reversible significant environmental impact, or monetary loss equal to or exceeding $1M but less than $10M.
Marginal	**3**	Could result in one or more of the following: injury or occupational illness resulting in one or more lost work day(s), reversible moderate environmental impact, or monetary loss equal to or exceeding $100K but less than $1M.
Negligible	**4**	Could result in one or more of the following: injury or occupational illness not resulting in a lost work day, minimal environmental impact, or monetary loss less than $100K.

Fig. 3: Risk severity categories [15]



PROBABILITY LEVELS			
Description	**Level**	**Specific Individual Item**	**Fleet or Inventory**
Frequent	**A**	Likely to occur often in the life of an item.	Continuously experienced.
Probable	**B**	Will occur several times in the life of an item.	Will occur frequently.
Occasional	**C**	Likely to occur sometime in the life of an item.	Will occur several times.
Remote	**D**	Unlikely, but possible to occur in the life of an item.	Unlikely, but can reasonably be expected to occur.
Improbable	**E**	So unlikely, it can be assumed occurrence may not be experienced in the life of an item.	Unlikely to occur, but possible.
Eliminated	**F**	Incapable of occurence. This level is used when potential hazards are identified and later eliminated.	Incapable of occurence. This level is used when potential hazards are identified and later eliminated.

Fig. 4: Risk probability levels [15]



RISK ASSESSMENT MATRIX				
SEVERITY PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			

Fig. 5: Risk Assessment Matrix [15]

Identification of hazards shall consider the entire system life cycle and potential impacts to personnel, infrastructures and environment. The documentation shall be done in the Hazard Tracking System (HTS). The risks assessment and documentation shall be done defining their severity category and probability level. *Severity* is defined as the magnitude of potential consequences of a mishap, whereas *Probability* as the likelihood of occurrence of a mishap.

All the possible severity categories and probability levels are defined respectively in Fig. 3 and Fig. 4. The assessed risks can be expressed using a Risk Assessment Code (RAC) which combines one severity category with one probability level. An example of RAC could be “2A”, which indicates a *critical* risk that happens *frequently*. Severity categories and frequency levels are combined in the Risk Assessment Matrix, represented in Fig. 5. The following step in Systems Safety process is to identify and document the potential risk mitigations. The main goal is to eliminate the hazard; when it is not possible, the risk should be reduced to the lowest acceptable level.

1.3.5. RAMS Analyses and Methods

Even though RAMS is a very relevant discipline, especially if integrated in MDO environment, currently it is still difficult to define the dependability [16] of aircraft on-board systems implementing innovative architectures; moreover, improving those configurations during the design process to avoid their possible faults is still a challenge. To define systems dependability – with a major focus on reliability and safety – different kinds of analyses and models have been developed in the last years. The most commonly used, especially from aircraft manufacturers such as Boeing and Airbus [17], are:

- **Common-Cause Analysis (CCA)**
- **Energy Trace and Barrier Analysis (ETBA)**
- **Failure Mode and Effects Analysis (FMEA)**
- **Fault Tree Analysis (FTA)**
- **Functional Hazard Analysis (FHA)**
- **Hazard and Operability study (HAZOP)**
- **Markov Analysis (MA)**
- **Physics of Failure (PoF) analysis**
- **Reliability Block Diagram (RBD)**
- **Zonal Safety Analysis (ZSA)**

Among those, the techniques necessary to assess the safety of civil airborne systems and equipment are described in [18]. In section 2.1 the FHA, FTA, FMEA, and RBD will be described more in detail to better understand those techniques procedures and their purposes.

Other methods, instead, are handbook-based and rely on documents like the MIL-HDBK-217 [19]. They are still used in different commercial and military avionic application to estimate reliability of on-board equipment, especially of the electronic ones. However, they have been strongly criticized by the U.S. National Academy of Sciences due to their inaccuracies and grave deficiencies [20]. Another important work that is worth to be highlighted, is a complete RAMS estimation methodology developed by Prof. Sergio Chiesa, from Polytechnic of Turin [21]. It uses statistical data to define reliability, maintainability, availability and safety of conventional aircraft subsystems (e.g. structure, engines, on-board systems, etc.). This methodology will be further discussed in section 2.3.

All these techniques and methods define in some way the dependability of aircraft systems, whether they have conventional or innovative architecture. Nevertheless, most of them rely on a document-based approach, which makes difficult and laborious to use the information gained through their analysis to improve system architectures. Moreover, this kind of approach strongly increases the possibility of human errors. Quite the opposite, a model-based approach would make development activities easier, enhancing design quality, system specification and communication within the development team.

1.4. Model-Based Systems Engineering

The Model-Based Systems Engineering (MBSE) is an emerging approach in Systems Engineering field. It is depicted as a model-centric approach, whose main goal is to develop a coherent model of the system, contrasting with the traditional document-based one [22]. The term *model* refers to a representation of one or more concepts that may be realized in physical world. It is a powerful instrument, which can be created using a modeling tool and consists of elements that represent system requirements, test cases, design and their relationships [23]. The document-based approach is focused on the generation of textual specifications, design documents and drawings that are exchanged between costumers, developers, testers and users. The MBSE, instead, enable the generation of a coherent model of the system, which specification, design and verification information. The great advantage of using models instead of documents, is the possibility of evolving and refining them whenever it becomes necessary. That is the reason why MBSE is expected to play an increasing role in the practice of Systems Engineering in the next decades. The potential benefits it can provide can be summarized in:

- Enhanced communications, especially across the development team and other stakeholders.
- Improved quality, including completeness, unambiguity and verifiability of requirements the traceability between them, the design, the analysis and testing.

- Increased productivity, including the reusing of existing models and the reduction of errors and time.
- Reduction of development risk, with a more accurate cost estimation for system development.

As support to this kind of approach, it has been developed the System Modelling Language (SysML) [24]. It is a general-purpose graphical modelling language, which is intended to facilitate the application of MBSE, to create cohesive and consistent models of systems. The great advantage of SysML is the capability of representing the behavior, structure, constraints and requirements of the considered system. All these aspects can be depicted using specific diagrams included in SysML [25]:

- **Activity Diagram:** represents behavior in terms of actions executed and transformation of actions inputs to outputs.
- **Block Definition Diagram (BDD):** represents structural elements, their composition and their classification.
- **Internal Block Diagram (IBD):** represents interconnections and interfaces between the parts of structural elements.
- **Parametric Diagram:** represents constraints on property values (such as $W = m \cdot g$) to support engineering analysis.
- **Requirements Diagram:** represents requirements and their relationships with other requirements, design elements and test cases.
- **Sequence Diagram:** represents behavior in terms of a sequence of messages exchanged between different systems or between parts of a system.
- **State Machine Diagram:** represents behavior of a specific entity in terms of transitions between its states, triggered by events.
- **Use Case Diagram:** represents functionality in terms of how the system is used by external entities.

Each diagram can graphically represent particularly aspects of the system model. The kinds of elements and associated symbols that can appear on a diagram are constrained on its kind. As an example, blocks can be represented on Block Definition Diagrams, but not on Activity Diagrams. Another advantage of SysML consists in the possibility of simulating its models if they are supported by an execution environment (such as the Foundational UML subset).

All these peculiarities enable to use this modeling language not only with the aim of designing innovative on-board system architectures, but even to study their reliability and safety.

1.5. Research Objectives

The objective of this research is to outline a series of guidelines which enables to perform RAMS analyses using the SysML. Specifically, the analyses that have been taken in account are: FHA, FTA, FMEA and RBD. Those guidelines shall be suitable for both already existing models and the ones that still have not been developed. Moreover, they shall also provide the necessary steps to extract the characteristic information of each analysis and put them on the relative documents, whatever they are (e.g. worksheets, text-based documents, diagrams, etc.).

The research also aims to integrate the results obtained from RAMS analyses into an MDO environment, so that it could be possible to evaluate their impact on other aircraft design parameters. Furthermore, different on-board system architectures will be taken in account and compared to evaluate both their differences in terms of safety, reliability and performance.

2. Literature Review

This chapter is focused on description and the analysis of the state of the art concerning the RAMS discipline. Specifically, it will describe more in detail some of the RAMS analyses listed in section 1.3.5. Afterwards, there will be an in-depth analysis of some research works aimed at developing methodologies to perform RAMS analyses with a model-based approach. Finally, an already existing RAMS estimation methodology will be discussed and the reasons that makes it inappropriate for the analysis of innovative on-board systems will be highlighted.

2.1. RAMS Analyses

Among the RAMS analyses listed in section 1.3.5, only FHA, FTA, FMEA and RBD have been chosen to be examined in depth. The reason is that they can be used as part of the safety assessment process which is performed during the development of a new aircraft. Specifically, the FHA can be first carried out at both aircraft and system levels to define the safety requirements. Then, to demonstrate that the design of a system will meet the requirements, a Preliminary System Safety Analysis (PSSA) can be performed using the FTA. Once the system development is complete, both FMEA and FTA can be used to carry out a System Safety Analysis (SSA) with the aim of verifying that the proposed design meets the specified requirements. Finally, the RBD can be used to define quantitatively the design reliability during a typical mission and evaluate its efficiency. It must be noticed that the implementation of innovative on-board system architectures may not influence the definition of safety requirements. That is because they may be the same as the ones defined for the conventional architecture.

2.1.1. Functional Hazard Analysis (FHA)

The Functional Hazard Analysis (FHA) is a predictive technique whose purpose is to identify and classify the system functions and the safety consequences associated with functional failure or malfunction (e.g. hazards) [15]. It is also used to identify environmental and health related consequences of functional failure or malfunction. Safety consequences will be classified in terms of severity, with the purpose of defining the system safety-critical functions (SCFs), safety-critical items (SCIs), safety-related functions (SRFs) and safety-related items (SRIs). The term *safety-critical* refers to a condition, event, operation, process or item whose mishap severity consequence is either Catastrophic or Critical (see Fig. 3). Whereas, the term *safety-related* refers to a condition, event, operation, process or item whose mishap severity consequence is either Marginal or Negligible (see Fig. 3). Items and functions which are either safety-critical or safety-related can be respectively identified as safety-significant items (SSIs) and safe-significant functions (SSFs).

The FHA is defined as one of the preliminary activities in the safety assessment process [26]. As illustrated in Fig. 6, it is first carried out for the whole aircraft. Then, it is performed again for each aircraft system, following the functions allocation.

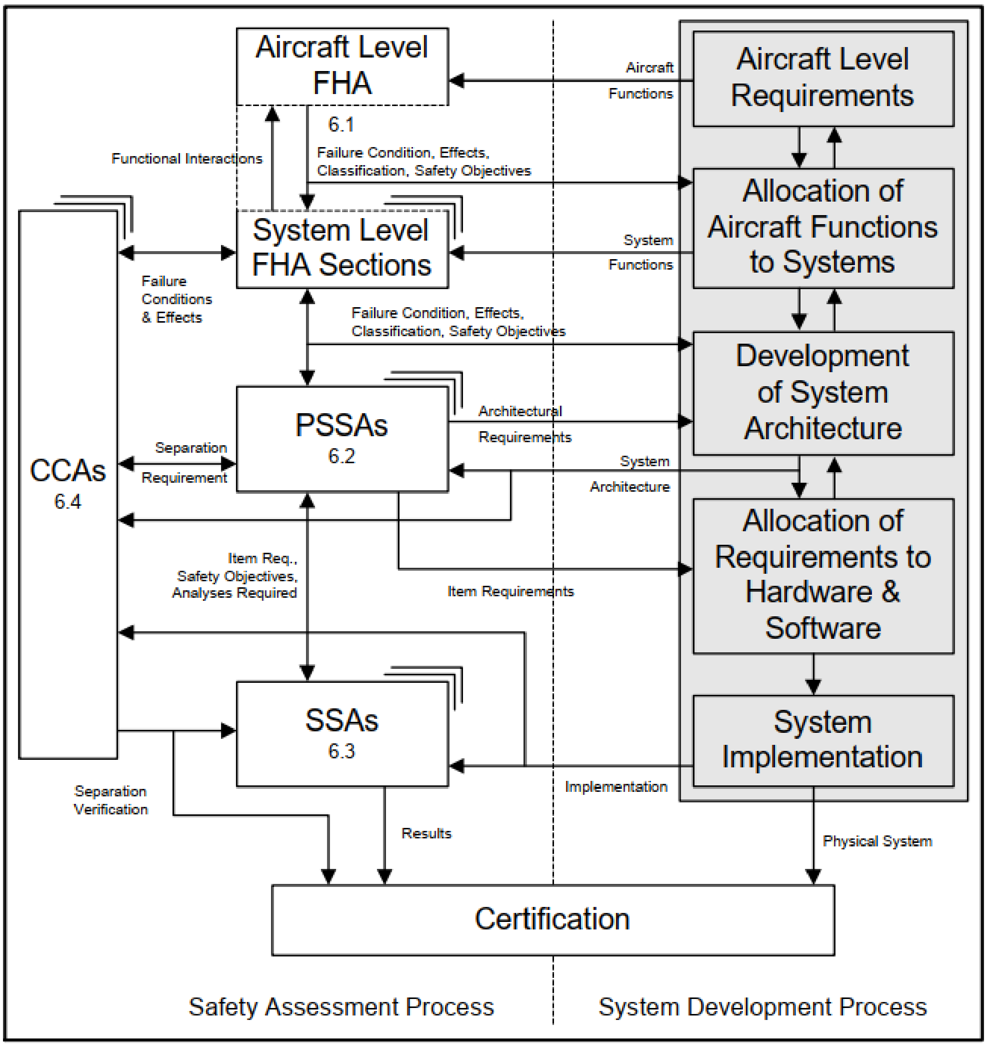

Aircraft Level FHA
6.1
Aircraft Functions
Aircraft Level Requirements
Functional Interactions
Failure Condition, Effects, Classification, Safety Objectives
Allocation of Aircraft Functions to Systems
System Level FHA Sections
System Functions
Failure Conditions & Effects
Failure Condition, Effects, Classification, Safety Objectives
Development of System Architecture
PSSAs
6.2
Architectural Requirements
CCAs
6.4
Separation Requirement
System Architecture
Allocation of Requirements to Hardware & Software
Item Req., Safety Objectives, Analyses Required
Item Requirements
SSAs
6.3
System Implementation
Implementation
Separation Verification
Results
Physical System
Certification
Safety Assessment Process
System Development Process

Fig. 6: Safety Assessment Process model [26]

The different steps to perform the FHA are detailed in [27]. The first one consists in gathering and interpreting the System Architecture data to identify and describe the functions performed. Those functions can be summarized using a functional hierarchy, a Functional Flow Block Diagram (FFBD), and a function/item matrix of the system. Then, it is necessary to evaluate the functional failures for hazards. There are different types of functional failure that shall be considered during the analysis of each function:

- **Fails to operate**: the considered function does not perform when the appropriate input is given.
- **Operates early or late**: the considered function operates earlier or later than it should have.

- **Operates out of sequence**: the considered function occurs before or after the wrong function (the one which operated too in late than it should have); n this case, the function occurs without receiving the appropriate inputs.
- **Unable to stop operation**: the considered function continues to operate even though the system should move on to the next function.
- **Degraded function or malfunction**: the considered function does not finish or completes only partially; in this case, the function generates improper outputs.

The next step is to identify safety-significant subsystems and interfaces associated with the functional failures described before. Those subsystems and interfaces are considered as SSIs and shall be allocated to an SSF. After that, it is necessary to identify the existing and recommended requirements and design constraints to assess, reduce, or eliminate the mishap risk associated with the considered hazard. These requirements and constraints shall be in form of fault tolerance, detection, annunciation, or recovery. The following step is to decompose each subsystem-level SSF defined before to the component level. This requires an understanding of how the component functions interact to perform the subsystems functions; moreover, it is necessary to analyze the functional failure at component level to identify new hazards and to further characterize the hazards identified previously. As with the subsystem-level allocation, each functional failure at component level should be associated with a single component. After identifying hazards and causal factors at all system levels and allocating them to the applicable components, it is necessary to identify the risk levels. In the end, the final FHA report shall be generated. To accomplish this last step, it is necessary to use a proper worksheet. This document can be found in different forms in literature. An example is represented in Fig. 61.

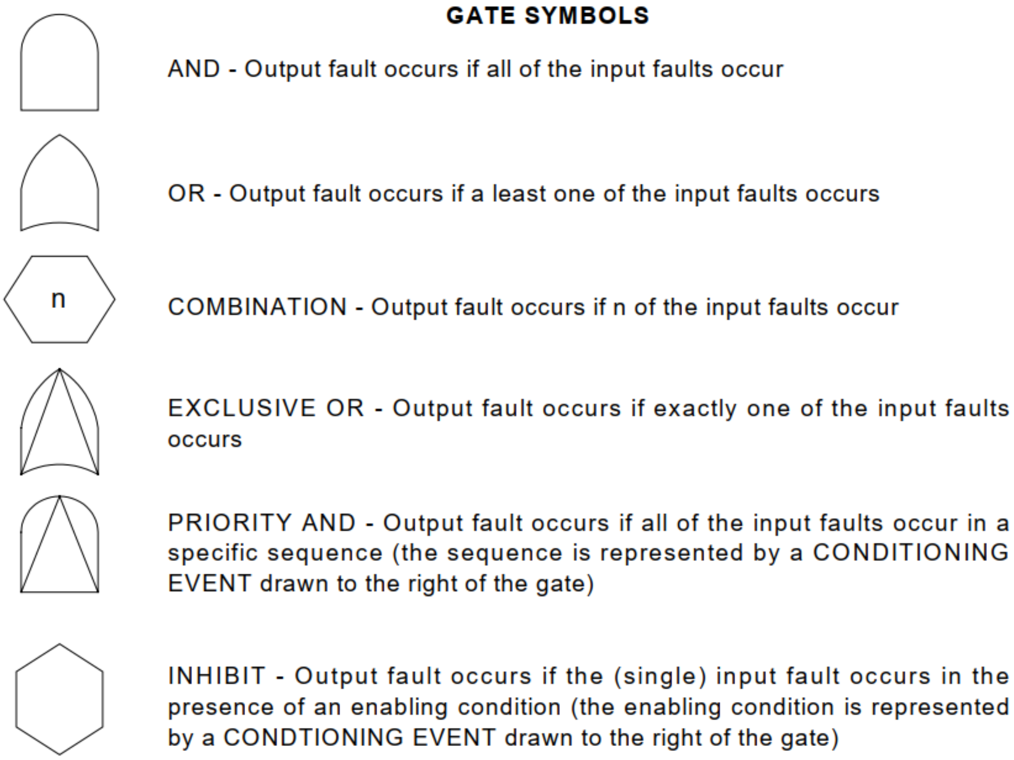
Even if the principles of FHA appear deceptively simple, it is possible to encounter different problems while performing it. Some of them can be resumed as:

- **Difficulty in defining functions**: it may result hard to identify functions at right level of abstraction from the available requirements documentation; if functions are expressed at too abstract level it results difficult to identify new hazardous failure modes; whereas, if they are expressed at too detailed level the FHA process takes too long time.
- **Difficulty in determining the effects**: it may result hard to define the effect of a function failure; particularly, the effect propagation to the next-high levels may be difficult to identify if the design is not clearly defined.
- **Difficulty in coupling or integrating**: it may result hard to identify the possible couplings or interactions between functions, since the FHA does not give any support or structure for addressing functional dependencies.

These problems are described more in detail in [29]. The same document provides an approach to avoid them while performing the FHA.

2.1.2. Fault Tree Analysis (FTA)

Fault Tree Analysis (FTA) is an analytical technique, whereby an undesired state of the system is analyzed in the context of its environment and operation to find all the possible ways in which it can occur [30]. Usually the undesired state is critical for system safety or reliability. The FTA can be performed using a graphical model called *fault tree*. It is a qualitative model, but it can also be evaluated quantitatively. It represents how various combinations of faults lead to a predefined undesired event, called *top event*. Faults can be defined as occurrences associated to abnormal conditions or defects (such as components failures, human errors, software errors, etc.), which may cause the system failure. The relationships between the different events are described using complex entities known as *gates*. They are represented with specific symbols in the fault tree model, as shown in Fig. 7.



AND - Output fault occurs if all of the input faults occur

OR - Output fault occurs if a least one of the input faults occurs

COMBINATION - Output fault occurs if n of the input faults occur

EXCLUSIVE OR - Output fault occurs if exactly one of the input faults occurs

PRIORITY AND - Output fault occurs if all of the input faults occur in a specific sequence (the sequence is represented by a CONDITIONING EVENT drawn to the right of the gate)

INHIBIT - Output fault occurs if the (single) input fault occurs in the presence of an enabling condition (the enabling condition is represented by a CONDTIONING EVENT drawn to the right of the gate)

Fig. 7: List of Gate symbols [30]

Each gate denotes a specific relationship between its input events and the resulting output event. The two basic types are the AND-gate and the OR-gate and are usually the most used. The others, instead, are special cases of these two. The AND-gate indicates that the output event can occur only if all the input events occur. Whereas the OR-gate, indicates that the occurrence of one of the input events is enough to make the output event occur. As an example, consider the FTAs represented in Fig. 8 and Fig. 9.

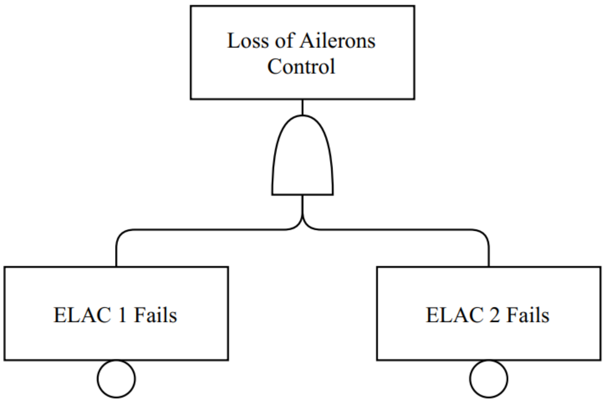


Fig. 8: AND-gate example, considering loss of ailerons control as top event

The first uses an AND-gate to show that the loss of ailerons control occurs only when both the Elevator Aileron Computers (ELACs) fail. The latter, instead, adopt the OR-gate to show that the ailerons control can be lost if the hydraulic actuators fail or both the ELACs fail; one of this two occurrences is enough to make the top event occur.

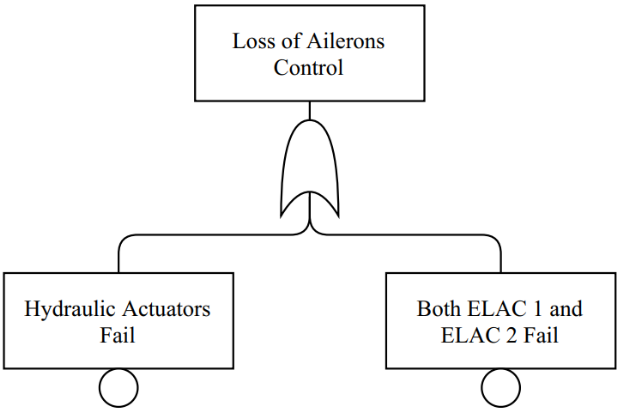


Fig. 9: OR-gate example, considering loss of ailerons control as top event

Events are generally represented in a fault tree using rectangles. However, some specific symbols can be used to represent *primary events*. These ones are events which have not been further developed; so, it is not specified what caused their occurrence. Examples of this kind of events are both ELAC 1 and ELAC 2 failure in Fig. 8, since it is not detailed what makes them fail. Primary events symbols are summarized in Fig. 10. Among theme, the most used is the *basic event*, which is represented with a circle under the fault description.

A fault tree can be constructed for a system which is being designed, as well as for one that is being implemented or is already operating. However, it is important to underline that the fault tree model does not represent all the possible system failures or all the possible causes for a single failure. Instead, it is tailored to a predefined top event and includes only faults which contribute to its occurrence.

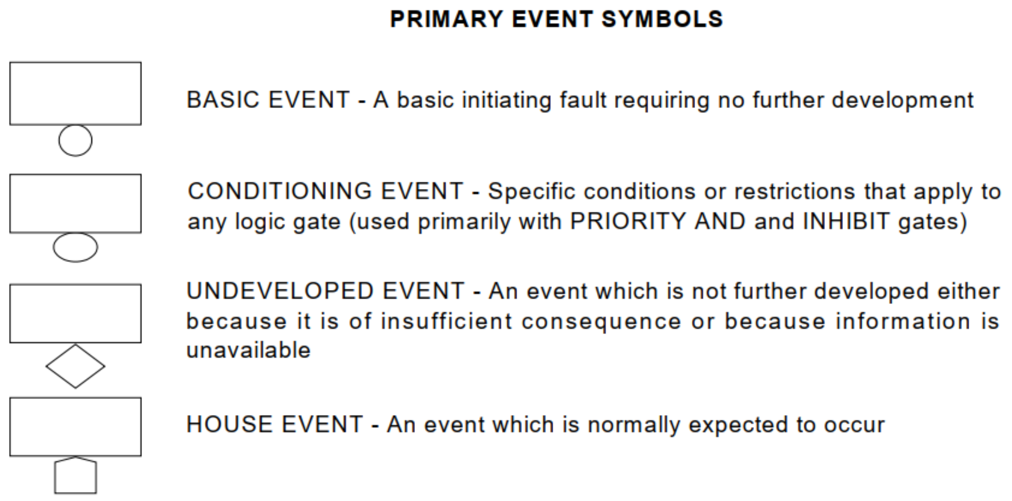

PRIMARY EVENT SYMBOLS
BASIC EVENT - A basic initiating fault requiring no further development
CONDITIONING EVENT - Specific conditions or restrictions that apply to any logic gate (used primarily with PRIORITY AND and INHIBIT gates)
UNDEVELOPED EVENT - An event which is not further developed either because it is of insufficient consequence or because information is unavailable
HOUSE EVENT - An event which is normally expected to occur

Fig. 10: List of Primary Event symbols [30]

The FTA can be categorized as a deductive system analysis. It means that is starts postulating the system failure in a certain way; then it attempts to find out which subsystem or component behavior contribute to make that failure happen. Indeed, the FTA starts with the definition of a predefined undesired event, which represents the system failure state. After that, faults which may lead to the top event are identified and connected each other using the gates. Starting from those faults, the same process is applied to trace back what caused them, until primary events are reached. The approach adopted in FTA results in being the opposite compared with the one used in inductive system analyses. These ones, indeed, starts defining the causes of a failure and then trace forward the resulting consequences. Therefore, this process can be repeated different times to evaluate the possible consequences that might occur after changing the initiating causes. Examples of inductive methods are the RBD, FMEA and FHA [31].

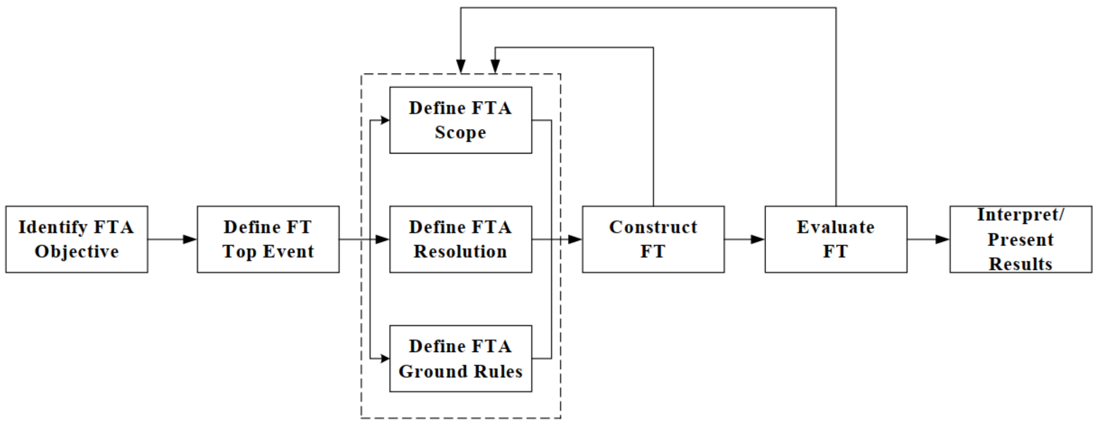


Fig. 11: FTA procedure steps [30]

As represented in Fig. 11, it is necessary to follow different steps to perform a successful FTA. The first five involve the problem formulation for an FTA, whereas the remaining three involve the FT construction, evaluation and interpretation of results. One of the most important steps is the definition of top event, which directs all the rest of the analysis. Indeed, if it is defined incorrectly, the entire FTA will be incorrect, involving wrong decisions being made. Generally, defining the system success criteria first, allows defining more easily system failures, as well as the undesired event. It is often useful to define several potential top events and then to decide the appropriate one. Moreover, if the mission has different phases, it may be necessary to define separate top events for each one of them.

2.1.3. Failure Mode and Effects Analysis (FMEA)

The Failure Mode and Effects Analysis (FMEA) is a procedure by which potential system failure modes are analyzed to determine the results or effects on that system. A *failure mode* describes the way the considered failure occurs and its impact on equipment operation. Each failure mode shall be classified according to its severity [32].

There are two general approaches for accomplishing an FMEA: a hardware approach and a functional approach. The first lists individual hardware items and analyzes their possible failure modes. The latter recognizes the different function that an item must perform and analyzes the failure modes of those functions. A combination of both two can be also considered.

FMEA shall be an integral part of system design process and shall be updated to reflect design changes. It shall be used to assess high risk items and to define preventive maintenance actions, test considerations and activities to minimize failure risk. The procedure to perform FMEA can be summarized in different steps:

a. **Define the system to analyze**, including internal interface functions, expected performance and failure definitions. The definition should also include tasks to be performed for each mission, mission phase, operational mode, environmental profiles, expected mission time, functions and outputs of each item.
b. **Construct functional and reliability block diagrams**, which illustrate operation, interrelationships and interdependencies of functional entities; it shall be done for each item configuration.
c. **Identify all the potential failure modes** of the considered item and its interfaces.
d. **Define failure modes effect** on other items or functions, on the system and on the mission to be performed.
e. **Evaluate each failure mode** in terms of the worst potential consequences and assign a severity classification category (severity categories are summarized in Fig. 3).

f. **Identify failure detection methods** and compensating provisions for each failure mode.
g. **Identify corrective actions** to eliminate the failure or control the risk.
h. **Identify effects of corrective actions** on other system attributes (e.g. requirements for logistic support).
i. **Document the analysis** and summarize the problems which could not be corrected by design; also, identify special controls necessary to reduce failure risk.

The FMEA results must be documented in a worksheet, such as the one represented in Fig. 62. Each column of this table concerns some specific information assessed using the analysis:

- **Identification number**: shall contain a serial number assigned for traceability purposes.
- **Item/function identification**: shall contain the name or the nomenclature of the item or the function being analyzed.
- **Function**: shall contain a concise statement of the function performed by the item; it shall include both inherent function and the relationship to interfacing item.
- **Failure mode and causes**: shall contain a description of all the predictable failure modes and of the most probable causes associated with them; if a failure mode has more than one cause, all the independent causes shall be identified and described.
- **Mission phase/ operational mode**: shall contain a concise statement of the mission phase and operational mode in which the failure occurs.
- **Failure effect**: shall contain a description of the consequences on item, function or status, of each assumed failure mode; failure effects shall focus on the block diagram element which is affected by the failure under consideration.
- **Local effect**: shall describe the impact that the assumed failure has on the operation and function of the considered item; it is possible for the local effect to be the failure mode itself.
- **Next higher level**: shall describe the impact that the assumed failure has on the operation and function of items which are in next the higher level; the assembly or function complexity is described with *indenture levels*, which progress from the more complex (system) to the simpler (component) divisions.
- **End effects**: shall describe the total effect that the assumed failure has on the operation, function or status of the uppermost system; the end effect described may also be the result of a double failure.
- **Failure detection method**: shall contain a description of the methods by which the occurrence of the failure mode is detected; instruments used for failure detection (such

as visual or audible warning devices, automatic sensing devices, etc.) shall be also identified.

- **Compensating provisions**: shall contain design provisions or operator actions which circumvent or mitigate the effect of the failure.
- **Severity class**: shall contain the severity classification assigned to each failure mode, according to the failure effect.
- **Remarks**: shall contain pertinent remarks that relate to and clarify any other column in the worksheet line; it shall also contain notes regarding recommendations for design improvements; notation of unusual conditions, failure effects of redundant items, recognition of particularly critical design features may be included.

The worksheets shall be organized to first display the higher indenture level of analysis and then proceed down through decreasing levels of the system. In literature, different kinds of FMEA worksheets can be found. Most of them differ for the number of columns and the required information; however, they always contain the basic columns necessary to perform a generic FMEA: item, function, failure mode, failure cause, failure effect, and severity class.

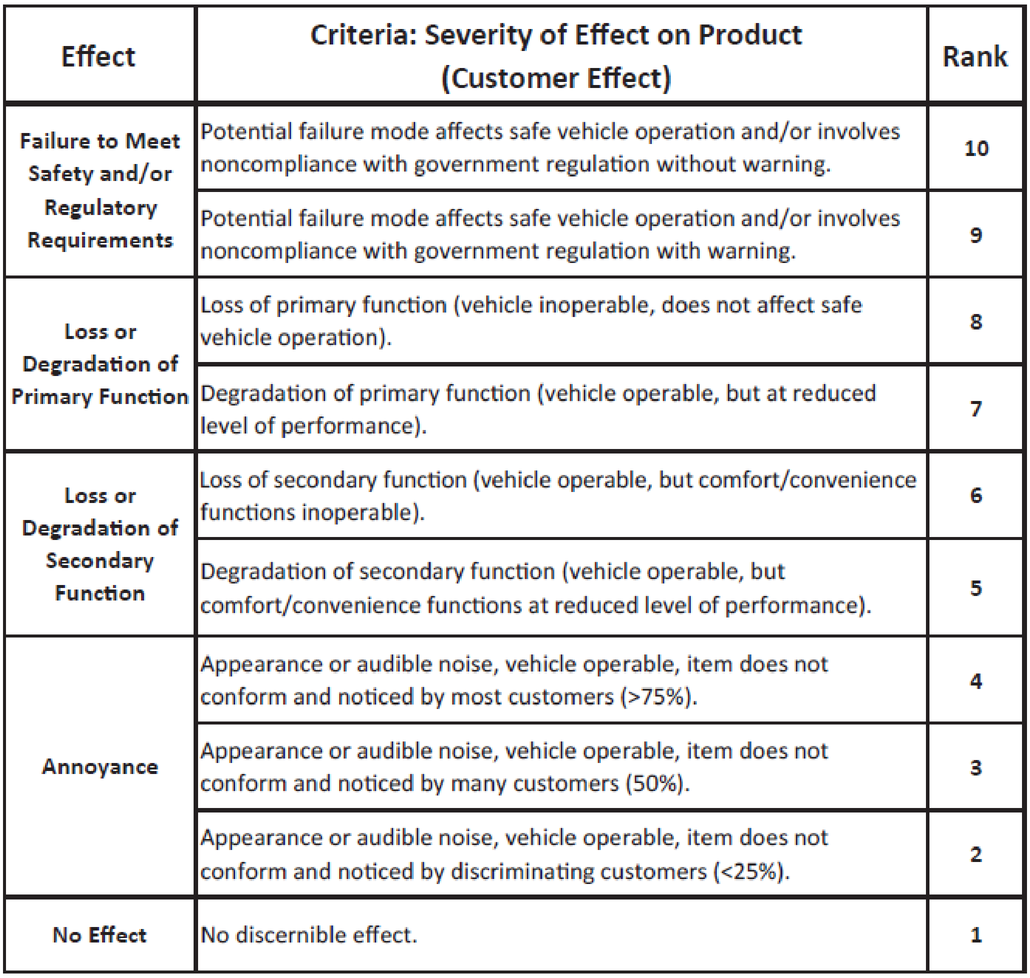
As stated in [33] there are different types of FMEA and the most common between them are:

- **System FMEA (SFMEA)**: highest-level analysis of an entire system, made up of various subsystems; the focus is on system-related deficiencies, (e.g. system safety and integration, interaction between subsystems and the surrounding environment, human interactions, etc.).
- **Design FMEA (DFMEA)**: analysis at subsystem level or at component level; the focus is on design related deficiencies (e.g. design improvement, safety and reliability assurance during useful life of equipment and interfaces between the adjacent components, etc.).
- **Process FMEA (PFMEA)**: analysis at manufacturing or assembly process level; the focus is on manufacturing related deficiencies (e.g. manufacturing process improvement, ensuring the product is built in a safe manner, with minimal down time, scrap and rework, etc.).

An example of a worksheet suitable for Design FMEA is represented in Fig. 63. It is possible to notice that it contains some different columns respect with the one shown in Fig. 62. Specifically, there are two new columns devoted to *design controls*. They shall contain information about actions or methods currently planned, or already in place, to reduce or eliminate the risk related to each potential cause. One of them concerns *prevention-type* controls, which are intended to reduce the likelihood that the problem will occur. The other concerns *detection-type* controls, which are intended to increase the likelihood that the problem will be detected before it reaches the end user. Moreover, in addition to Severity, Design FMEA requires two more parameters:

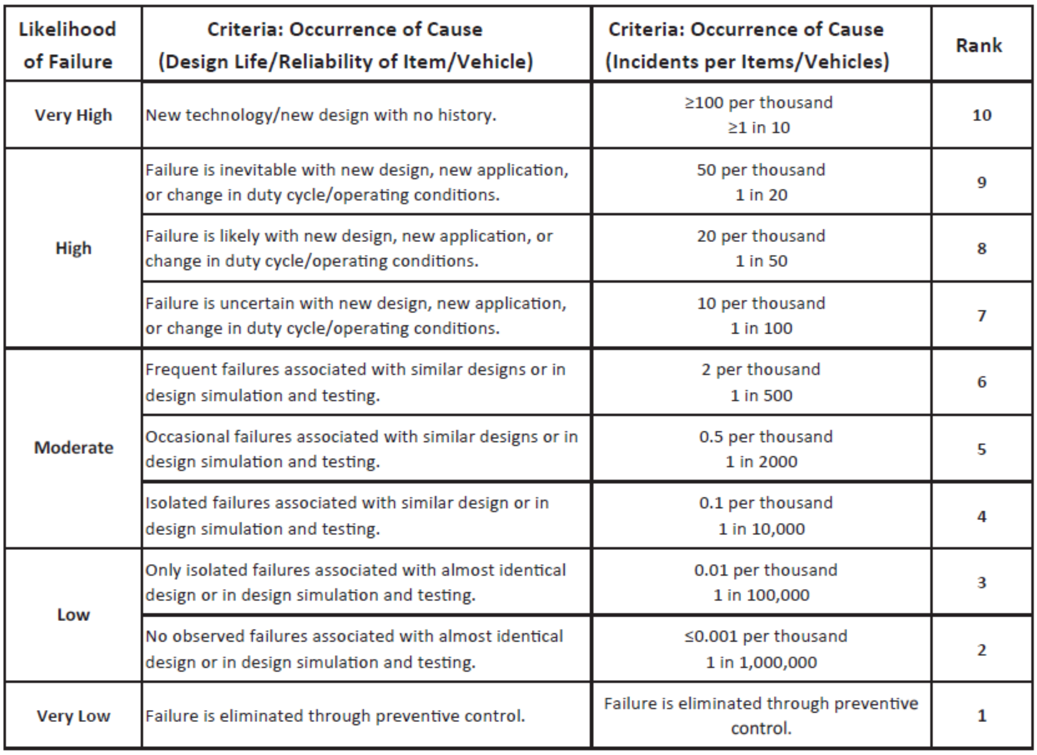
- **Occurrence**: a ranking number associated with the probability that a failure mode and its related cause will be present in the item being analyzed.
- **Detection**: ranking number associated with the best control from the list of detection controls.

Each one of these parameters gives a quantitative description of some of the other information displayed in the worksheet. Severity is related to the effect of a failure and describes how much serious it is; Occurrence is related to the cause of a failure and describes the likelihood that it will occur; Detection is related to the detection controls adopted and highlights how much they are effective. The possible ranks that can be assumed by these parameters are described in [33] and are also represented in Fig. 12, Fig. 13 and Fig. 14. It is important to notice that in this case the Severity rank has different values in respect with the categories described in Fig. 3.



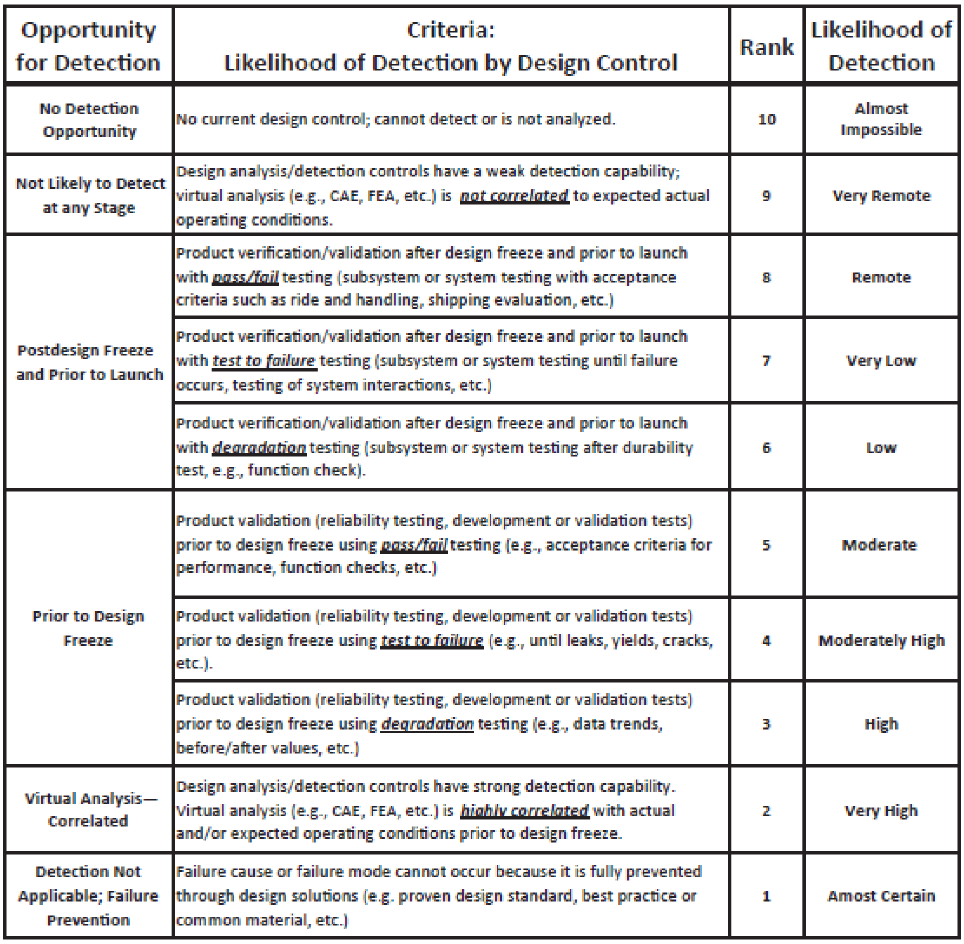
Effect	Criteria: Severity of Effect on Product (Customer Effect)	Rank
Failure to Meet Safety and/or Regulatory Requirements	Potential failure mode affects safe vehicle operation and/or involves noncompliance with government regulation without warning.	10
	Potential failure mode affects safe vehicle operation and/or involves noncompliance with government regulation with warning.	9
Loss or Degradation of Primary Function	Loss of primary function (vehicle inoperable, does not affect safe vehicle operation).	8
	Degradation of primary function (vehicle operable, but at reduced level of performance).	7
Loss or Degradation of Secondary Function	Loss of secondary function (vehicle operable, but comfort/convenience functions inoperable).	6
	Degradation of secondary function (vehicle operable, but comfort/convenience functions at reduced level of performance).	5
Annoyance	Appearance or audible noise, vehicle operable, item does not conform and noticed by most customers (>75%).	4
	Appearance or audible noise, vehicle operable, item does not conform and noticed by many customers (50%).	3
	Appearance or audible noise, vehicle operable, item does not conform and noticed by discriminating customers (<25%).	2
No Effect	No discernible effect.	1

Fig. 12: Severity rank table [33]



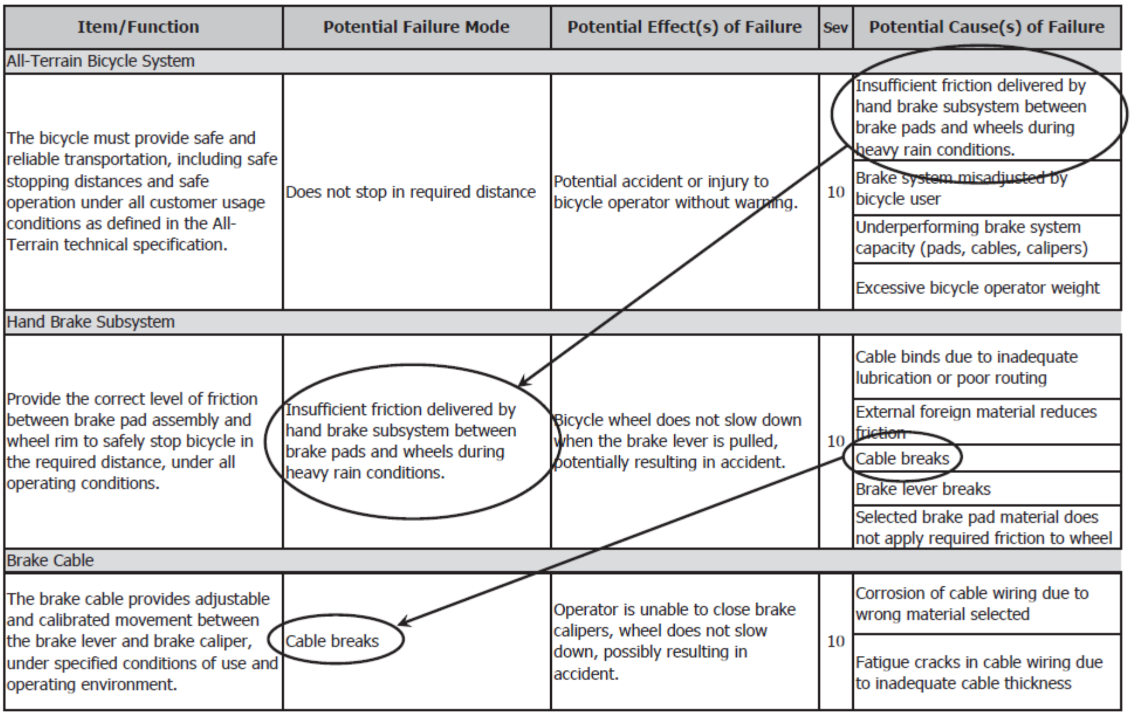
Likelihood of Failure	Criteria: Occurrence of Cause (Design Life/Reliability of Item/Vehicle)	Criteria: Occurrence of Cause (Incidents per Items/Vehicles)	Rank
Very High	New technology/new design with no history.	≥100 per thousand ≥1 in 10	10
High	Failure is inevitable with new design, new application, or change in duty cycle/operating conditions.	50 per thousand 1 in 20	9
	Failure is likely with new design, new application, or change in duty cycle/operating conditions.	20 per thousand 1 in 50	8
	Failure is uncertain with new design, new application, or change in duty cycle/operating conditions.	10 per thousand 1 in 100	7
Moderate	Frequent failures associated with similar designs or in design simulation and testing.	2 per thousand 1 in 500	6
	Occasional failures associated with similar designs or in design simulation and testing.	0.5 per thousand 1 in 2000	5
	Isolated failures associated with similar design or in design simulation and testing.	0.1 per thousand 1 in 10,000	4
Low	Only isolated failures associated with almost identical design or in design simulation and testing.	0.01 per thousand 1 in 100,000	3
	No observed failures associated with almost identical design or in design simulation and testing.	≤0.001 per thousand 1 in 1,000,000	2
Very Low	Failure is eliminated through preventive control.	Failure is eliminated through preventive control.	1

Fig. 13: Occurrence rank table [33]



Opportunity for Detection	Criteria: Likelihood of Detection by Design Control	Rank	Likelihood of Detection
No Detection Opportunity	No current design control; cannot detect or is not analyzed.	10	Almost Impossible
Not Likely to Detect at any Stage	Design analysis/detection controls have a weak detection capability; virtual analysis (e.g., CAE, FEA, etc.) is *not correlated* to expected actual operating conditions.	9	Very Remote
Postdesign Freeze and Prior to Launch	Product verification/validation after design freeze and prior to launch with *pass/fail* testing (subsystem or system testing with acceptance criteria such as ride and handling, shipping evaluation, etc.)	8	Remote
	Product verification/validation after design freeze and prior to launch with *test to failure* testing (subsystem or system testing until failure occurs, testing of system interactions, etc.)	7	Very Low
	Product verification/validation after design freeze and prior to launch with *degradation* testing (subsystem or system testing after durability test, e.g., function check).	6	Low
Prior to Design Freeze	Product validation (reliability testing, development or validation tests) prior to design freeze using *pass/fail* testing (e.g., acceptance criteria for performance, function checks, etc.)	5	Moderate
	Product validation (reliability testing, development or validation tests) prior to design freeze using *test to failure* (e.g., until leaks, yields, cracks, etc.).	4	Moderately High
	Product validation (reliability testing, development or validation tests) prior to design freeze using *degradation* testing (e.g., data trends, before/after values, etc.)	3	High
Virtual Analysis—Correlated	Design analysis/detection controls have strong detection capability. Virtual analysis (e.g., CAE, FEA, etc.) is *highly correlated* with actual and/or expected operating conditions prior to design freeze.	2	Very High
Detection Not Applicable; Failure Prevention	Failure cause or failure mode cannot occur because it is fully prevented through design solutions (e.g. proven design standard, best practice or common material, etc.)	1	Amost Certain

Fig. 14: Detection rank table [33]



Item/Function	Potential Failure Mode	Potential Effect(s) of Failure	Sev	Potential Cause(s) of Failure
All-Terrain Bicycle System				
The bicycle must provide safe and reliable transportation, including safe stopping distances and safe operation under all customer usage conditions as defined in the All-Terrain technical specification.	Does not stop in required distance	Potential accident or injury to bicycle operator without warning.	10	Insufficient friction delivered by hand brake subsystem between brake pads and wheels during heavy rain conditions. Brake system misadjusted by bicycle user Underperforming brake system capacity (pads, cables, calipers) Excessive bicycle operator weight
Hand Brake Subsystem				
Provide the correct level of friction between brake pad assembly and wheel rim to safely stop bicycle in the required distance, under all operating conditions.	Insufficient friction delivered by hand brake subsystem between brake pads and wheels during heavy rain conditions.	Bicycle wheel does not slow down when the brake lever is pulled, potentially resulting in accident.	10	Cable binds due to inadequate lubrication or poor routing External foreign material reduces friction Cable breaks Brake lever breaks Selected brake pad material does not apply required friction to wheel
Brake Cable				
The brake cable provides adjustable and calibrated movement between the brake lever and brake caliper, under specified conditions of use and operating environment.	Cable breaks	Operator is unable to close brake calipers, wheel does not slow down, possibly resulting in accident.	10	Corrosion of cable wiring due to wrong material selected Fatigue cracks in cable wiring due to inadequate cable thickness

Fig. 15: Excerpts of a bicycle FMEAs [33]

The arithmetic product of Severity, Occurrence and Detection defines another parameter called Risk Priority Number (RPN). It is a numerical ranking of each potential failure mode and shall be reported in its appropriate column in the Design FMEA worksheet.

$$RPN = Sev \times Occ \times Det \tag{2.1}$$

However, the RPN is not a perfect representation of risk associated with a failure mode and its related cause since it is subjective and not continuous.

2.1.4. Failure Mode, Effect and Criticality Analysis (FMECA)

It is also worth considering another commonly used version of FMEA called the Failure Mode, Effect and Critically Analysis (FMECA). It is a procedure which documents all possible failure in a system design, determines the effect of each failure on system operation and ranks those failures according to the criticality category of failure effect and probability of occurrence [34]. The difference between this version and the one described previously is the introduction of another kind of analysis. Indeed, the FMECA can be defined as the combination of FMEA and the Criticality Analysis (CA). This one is a procedure which determines the magnitude of criticality to system operational success related to a system component. Two steps are necessary to perform the CA:

a. **Identify critical failure modes** of all components considered in the FMEA for each equipment configuration. If the effect of failure modes on mission success or crew safety cannot be determined, they will be considered critical only if they are cause of failure of one or more of the system's inputs of outputs.

b. **Compute the Criticality Number** (C_r) for each system component with critical failure modes. The C_r represents the number of system failures – of a specific type –which are expected per million missions due to the component's critical failure modes.

The critical failure modes identification should be performed in accordance with the criticality categories described in [34]. Whereas, the Criticality Number can be calculated as follows:

$$C_r = \sum_{n=1}^{j} (\beta \alpha K_E K_A \lambda_G t \cdot 10^6)_n \tag{2.2}$$

where:

- j is the total number of critical failure modes in the system component.
- n is the index of summation for critical failure modes in the system component.
- t is the operating time (in hours or the number of operating cycles) of the component.
- λ_G is a generic failure rate (in failures per hours or cycles) of the component.
- β is the conditional probability that the failure effects of the critical failure mode occur (considering that the critical failure mode has occurred).
- α is the fraction of all the failures (λ_G) due to the failure mode under consideration, that the component experiences.
- K_E is an environmental factor which adjusts λ_G; it considers the difference between the environmental stresses present when λ_G was measured and the environmental stresses under which the component is going to be used.
- K_G is an operational factor which adjusts λ_G; it considers the difference between the operating stresses present when λ_G was measured and the operating stresses under which the component is going to be used.

The product of α, K_E, K_G, and λ_G represents the failure rate of each critical failure mode. These parameters shall be replaced with the failure rates gained through the test program, as they become available.

2.1.5. Reliability Block Diagram (RBD)

The Reliability Block Diagram (RBD) is an inductive model, in which the considered system is represented by means of blocks. These ones correspond to distinct elements, such as components or subsystems and are combined according to system-success pathways [30]. The RBD purpose is to show how the relationships among system essential elements allow achieving the operational success [35]. It represents an approach to analyze complex systems and to determine their reliability starting from the already known reliability of their elements. Within the diagram, blocks can be combined using different kinds of configurations. Each one of them can involve an increase or a decrease of system reliability. Some of those configurations are briefly described below.

In series configuration, blocks are arranged in a single row. It means that all the components represented with these blocks must function to make the system function. It is enough that one of them fails to make the entire system fail. A representation of series configuration can be found in Fig. 16.

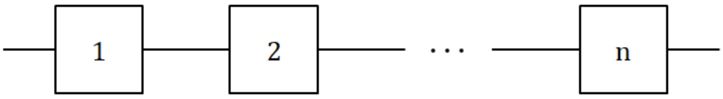


Fig. 16: Series Configuration representation

System reliability (R_s), in this case, is the probability that every component will carry out its intended function. So, it can be expressed as:

$$R_s = R_1 R_2 \cdots R_n = \prod_{i=1}^{n} R_i \tag{2.3}$$

Since reliability is a probability, its value must be between zero and one. Therefore, Eq. 2.3 highlights that in a series configuration the more is the number of components (n), the less the system results being reliable.

In the parallel configuration, blocks are arranged in a single column without any connection between them. It means that to make the system fail, all the components represented with blocks must fail. If only one of them functions correctly, the entire system functions. A representation of parallel configuration can be found in Fig. 17.

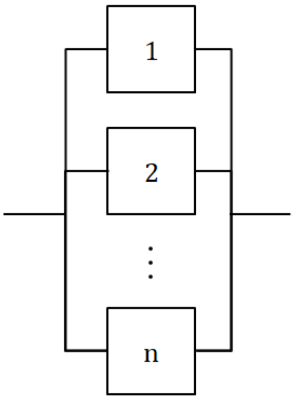


Fig. 17: Parallel Configuration representation

System reliability, in this case, is the complementary value of system failure probability (F_s). This one is defined by the probability that all the components fail:

$$F_s = (1 - R_1)(1 - R_2) \cdots (1 - R_n) = \prod_{i=1}^{n} (1 - R_i) \tag{2.4}$$

So, system reliability in a parallel configuration can be expressed as:

$$R_s = 1 - F_s = 1 - \prod_{i=1}^{n} (1 - R_i) \tag{2.5}$$

Eq. 2.4 highlights that the more is the number of components (n), the less is the system failure probability. Reliability, instead, is complementary and tends to increase when the number of components increases, as shown in Eq. 2.5. The parallel configuration is also called redundant configuration. That is because each block replicates the other blocks functions, avoiding system failure in case one or more of them fail.

Series and Parallel configurations can be combined to define adequately reliability of complex systems. An example of combined configuration is represented in Fig. 18.

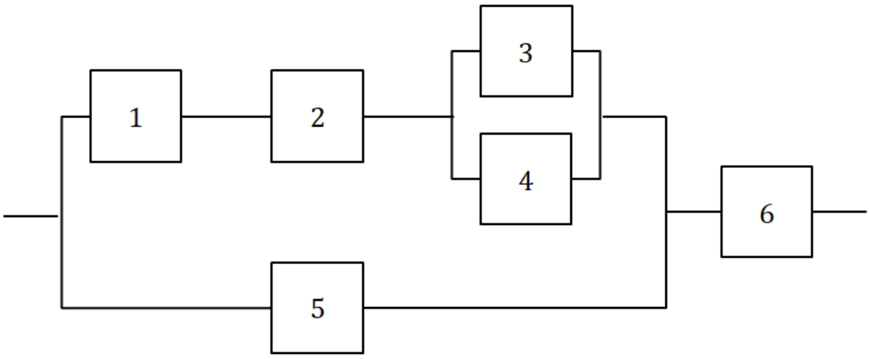


Fig. 18: Combined series-parallel configuration example

To compute the entire system reliability, in this case, it is necessary to divide the diagram in different subsystems which have only a series or a parallel configuration. As an example, the diagram shown in Fig. 18 is composed by a subsystem which has blocks 3 and 4 in parallel. This one, in turn, is part of another subsystem which has a series configuration. Therefore, its reliability can be computed as:

$$R_{sub} = R_1 R_2 [1 - (1 - R_3)(1 - R_4)] \tag{2.6}$$

Applying the same method for the remaining blocks, the system reliability results being:

$$R_s = R_6 \{1 - [1 - R_5][1 - R_{sub}]\} \tag{2.7}$$

Combining series and parallel configurations allows increasing and decreasing the system reliability. This shall be in line with the design objectives and functions.

A system characterized by n identical and independent components in a parallel configuration, that works only if k of them work, is called k-out-of-n system. Fig. 19 represents an example of a system composed by different k-out-of-n subsystem in a series configuration.

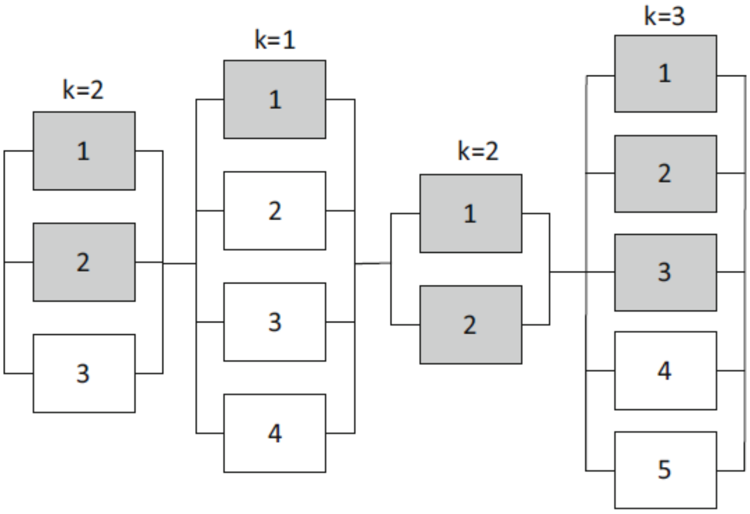


Fig. 19: Four *k*-out-of-*n* subsystems in a series configuration example [36]

Reliability, in this case, can be calculated using the *binomial probability distribution*:

$$P(x) = \binom{n}{x} R^x (1-R)^{n-x} \tag{2.8}$$

$$\binom{n}{x} = \frac{n!}{x!\,(n-x)!} \tag{2.9}$$

The Eq. 2.8 represents the probability that exactly x components are operating; whereas Eq. 2.9 is the total number of ways (or combinations) in which can be obtained x successes. Finally, system reliability can be calculated as:

$$R_s = \sum_{x=k}^{n} P(x) \tag{2.10}$$

The Eq. 2.10 represents the probability that, among n components, k or more of them will function without failures.

2.2. Model-Based RAMS Analyses

In the last years different methods were developed to perform RAMS analyses following a model-based approach. Some proposed the employment of modeling and simulation tools (such as Simulink or SCADE) to perform system safety analyses activities [37]. This approach has been called Model-Based Safety Assessment (MBSA). To support this kind of analysis the traditional "V" model has been modified, as shown in Fig. 20.

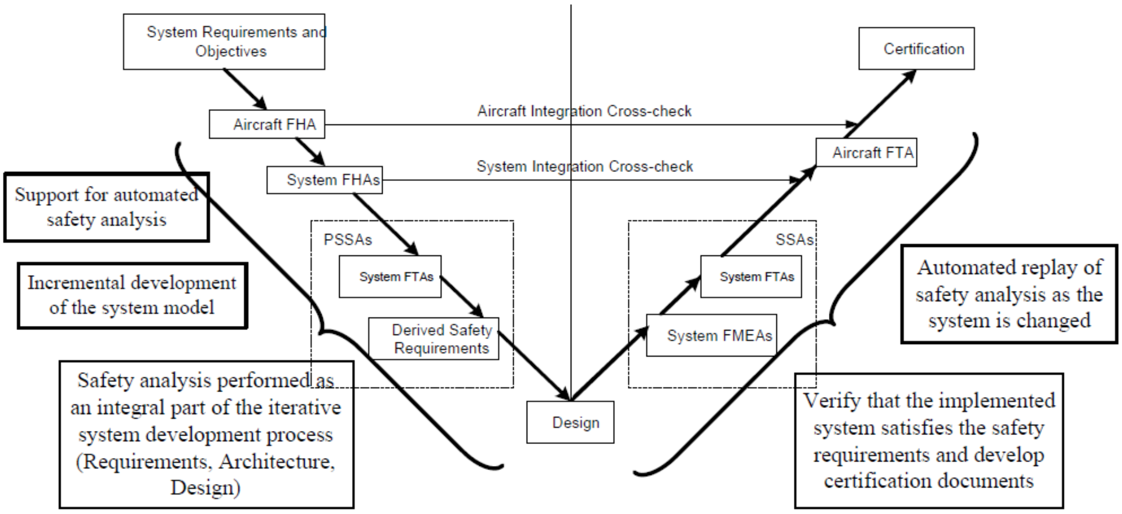

System Requirements and Objectives
Aircraft FHA
System FHAs
PSSAs
System FTAs
Derived Safety Requirements
Design
System FMEAs
System FTAs
SSAs
Aircraft FTA
Certification
Aircraft Integration Cross-check
System Integration Cross-check
Support for automated safety analysis
Incremental development of the system model
Safety analysis performed as an integral part of the iterative system development process (Requirements, Architecture, Design)
Automated replay of safety analysis as the system is changed
Verify that the implemented system satisfies the safety requirements and develop certification documents

Fig. 20: Modified "V" model for MBSA [37]

Other studies, instead, suggest the use of SysML language to perform some common RAMS analyses (such as FHA, FTA, FMEA, etc.) Among these, in [38] is defined a method to analyze already existing SysML models and automatically produce an FMEA. This gives the possibility to automatize the evaluation of system safety and reliability. But, the necessity of an already defined model limits the capability of performing FMEA and makes its application more complex. In [39], instead, State Machine Diagrams and Internal Block Diagrams are used to produce FTA and FMECA. The states have been used to represent functional and dysfunctional behaviors and they have been allocated to the system components represented in the IBD. Moreover, triggers and guards have been used to define the logical gates to consider in the FTA. However, both FTA and FMECA generation has been done using some specific tools of the software Magic Draw and it is not detailed how the information defined in the model are used to perform the analyses.

The work in [40] aims representing different RAMS analyses with SysML diagrams. It starts with the definition of both functional and dysfunctional system behaviors using Block Definition Diagrams. Then it is defined how to use Activity Diagrams and Sequence Diagrams to perform the FHA. In the end, it is also declared the purpose of performing both FTA and FMEA, but it is not specified how to do that with the presented model-based approach.

Finally, other research works employ different SysML diagrams to perform either FTA or FMEA. As an example, in [41] the FMEA worksheet is filled using information gained from different IBDs. However, this method does not enable to adequately specify causes and effects of a possible failure. Furthermore, it does not allow to identify all the possible failure modes that may affect the considered system. In [42], instead, a combination of BDDs, State Machine Diagrams and Activity Diagrams is used to define fault trees. Even in this case, the fault events that is possible to define are limited, since they are constrained only to generic failures of system components. Moreover, the logical gates can be defined only through the allocation of components to actions, which does not allow taking in account some combination of faults that may lead to the top event.

2.3. RAMS Statistical Methodology

A complete RAMS estimation methodology has been already developed by Prof. Sergio Chiesa, form Polytechnic of Turin. Its aim is to evaluate reliability, safety and maintainability of aircraft systems. It is based on a **top-down** approach, which means that systems failure rate and maintenance hours are calculated starting from the aircraft failure rate and maintenance hours. Indeed, the methodology starts using statistical data concerning several conventional aircraft to define some peculiar coefficients. Those ones are then used to evaluate the failure rate and maintenance hours of the aircraft under analysis. Afterwards, it uses again statistical data to define an average weight for each aircraft system. Finally, those weights are related to the aircraft failure rate and maintenance hours to determine those parameters for all the considered systems.

$$\lambda_i = \lambda \cdot \frac{K_i \left({}^{W_i}/_{MEW} \right)}{\sum_i \lambda_{i\_non\ normalized}} \tag{2.11}$$

$$K_i = \frac{\left({}^{\lambda_i}/_{\lambda} \right)_{avarage}}{\left({}^{W_i}/_{MEW} \right)_{avarage}} \tag{2.12}$$

The Eq. 2.11 and 2.12 show the relationships used to define systems failure rate. It is possible to notice that they both rely on the aircraft Maximum Empty Weight (MEW) and failure rate, highlighting the use of the top-down approach. Even if it results being one of the most complete RAMS estimation methodology, its dependency on statistical data makes it inappropriate for the evaluation of innovative on-board systems dependability. Instead, it results being a valid instrument to estimate the dependability of conventional systems that are still essential in modern aircraft.

Starting from the work of Prof. Chiesa, a new RAMS estimation methodology has been developed. This one considers also the implementation of some new technologies such as: EHAs, composite structures and Laminar Flow Wings (LFW) [43]. However, this thesis work takes in account only one innovative technology concerning on-board systems, that is the EHA. Moreover, the approach used for the estimation of systems dependability is still top-down. Therefore, it is not suitable for the evaluation of innovative on-board systems dependability.

3. Model-Based RAMS Estimation Methodology

The proposed RAMS estimation methodology aims defining safety and reliability using the four analyses described in section 2.1. Specifically, in sections 3.1, 3.2, 3.3 and 3.4 will be presented the necessary steps to perform respectively the FHA, FTA, FMEA and RBD using a model-based approach. It will be defined how to model and depict the most important information required by each analysis. In this way, it will be possible to easily analyze safety and reliability directly from models or to collect the information necessary to perform those analyses[1].

3.1. Model-based FHA

The model-based approach developed to perform an FHA relies on Activity Diagrams, which are analogous to functional flow diagrams but provide some enhanced capabilities. As an example, Activity Diagrams allow modeling the type of matter, energy or data exchanged while performing certain *actions*. Moreover, they provide the capability to express relationships between *activities* and structural aspects (such as *Blocks*, *parts*, etc.) of system under analysis. This aspect involves the possibility to define easier which may be the causes and effects related to a functional failure. The steps necessary to perform a model-based FHA can be summarized in:

1. Define an *Activity*
2. Develop an Activity Diagram
3. Define a new *Activity* to represent a functional failure
4. Use *send signal action* to identify a functional failure.
5. Use *accept event action* to define the functional failure effect.
6. Define hazard severity class and probability as *attributes* of the *signal*

These steps allow defining different Activity Diagrams, which represent different functional failures and describe their effects on other functions. Each diagram provides the essential information necessary to compile a generic FHA worksheet.

3.1.1. Define an *Activity*

The first step to perform a model-based FHA is to define an *Activity*. It is used for describing a behavior which specifies the transformation of inputs to outputs. The execution of an *Activity* can be described through a controlled sequence of *actions*, which can accept inputs and produce

[1] In this chapter, all the terms highlighted with the *Italic* font represent a characteristic element of SysML language.

outputs. The *actions* execution order can be defined using *control flows*, whereas *object flows* can be used to describe which items flow between them.

3.1.2. Develop an Activity Diagram

After defining the *Activity*, an Activity Diagram shall be developed to represent its *actions*, their execution order and the items that flow between them. An example concerning a simple braking system is shown in Fig. 21.

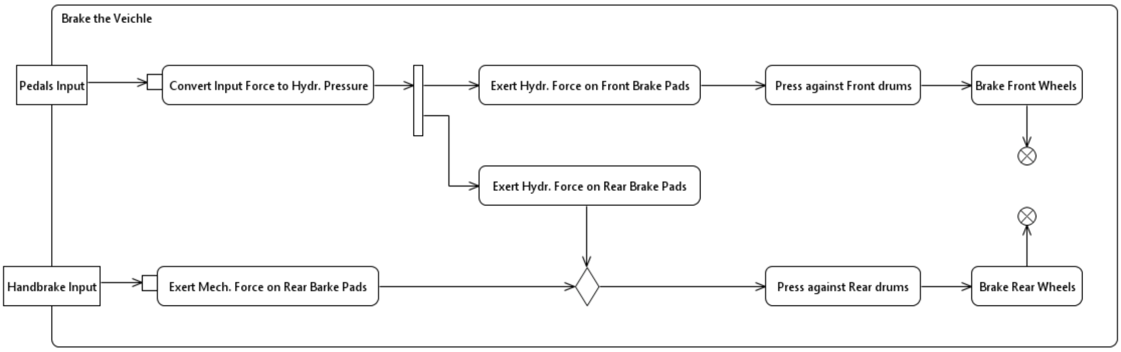

Brake the Veichle
Pedals Input
Convert Input Force to Hydr. Pressure
Exert Hydr. Force on Front Brake Pads
Press against Front drums
Brake Front Wheels
Exert Hydr. Force on Rear Brake Pads
Handbrake Input
Exert Mech. Force on Rear Barke Pads
Press against Rear drums
Brake Rear Wheels

Fig. 21: Simple braking system Activity Diagram

The *Activity* has two *input parameters*: one describes the input provided by pedals, whereas the other represents the input provided by handbrake. After receiving an input, the *Activity* proceeds with a sequence of *actions* which aim to exert force on brake pads, press them against the drums and then brake the wheels. It can be noticed that there are two different ways to brake the rear wheels. Both can be used, since the *merge node* (depicted as a white diamond) provides an output as soon as it receives an input from one of the two *control flows* to which it is connected. Therefore, it is possible to use either hydraulic force or mechanical force to press the rear brake pads on drums and brake their relative wheels.

3.1.3. Define a new *Activity* to represent a functional failure

To represent a functional failure, it is necessary to define a new *Activity* (starting from the one defined before) and represent it with another Activity Diagram. The aim is to show the failure occurrence and its effect on the behavior described before. This procedure can be applied to describe each functional failure defined in the FHA.

3.1.4. Use *send signal action* to identify the functional failure

The functional failure shall be represented using a *send signal action*. It is a specialized kind of *action* that generates and sends a *signal* to a specific target when it becomes enabled. The *signal* at issue shall represent the functional failure. *Send signal actions* are usually depicted using a convex pentagon shaped like a signpost.

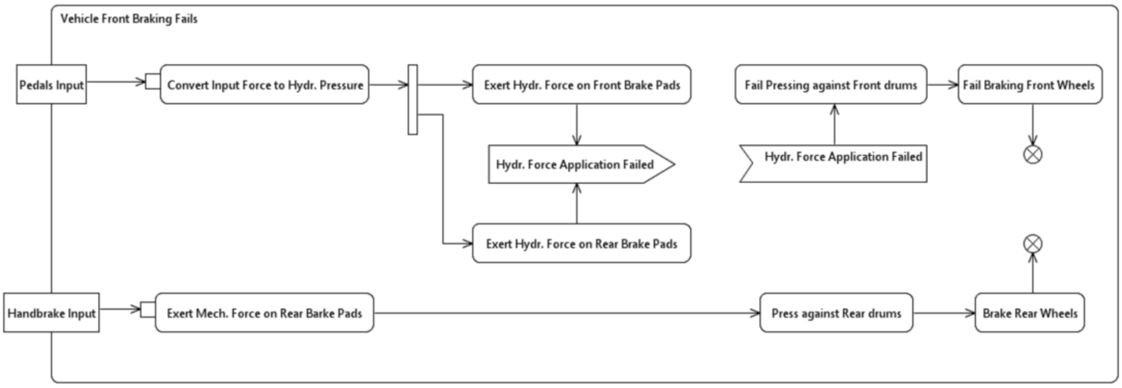

Vehicle Front Braking Fails
Pedals Input
Convert Input Force to Hydr. Pressure
Exert Hydr. Force on Front Brake Pads
Fail Pressing against Front drums
Fail Braking Front Wheels
Hydr. Force Application Failed
Hydr. Force Application Failed
Exert Hydr. Force on Rear Brake Pads
Handbrake Input
Exert Mech. Force on Rear Barke Pads
Press against Rear drums
Brake Rear Wheels

Fig. 22: Simple braking system – 1st functional failure example

The Activity Diagram in Fig. 22 shows the braking system inability to exert hydraulic force on both rear and front brake pads. This failure is represented with a *send signal action*, which is enabled when the *actions* aimed to apply hydraulic force do not behave as intended. Therefore, it is possible to understand that the functional failure occurs when the braking system is not able to exert hydraulic force on both front and rear brake pads.

3.1.5. Use *accept event action* to define the functional failure effect

The functional failure effects can be represented with an *accept event action*. This is another specialized kind of *action* that waits for an asynchronous event before it continues executing. In this case, the *accept event action* shall wait for a *signal event*, which is triggered by a *send signal action*. Its execution involves a sequence of *actions* which describe the functional failure effects. The *accept event action* shown in Fig. 22 starts executing after accepting the *signal event* which has been triggered from the corresponding *send signal action*. The sequence of *actions* that follows, describes the inability to press front brake pads on drums, and consequently to brake the relative front wheels. However, it is still possible to brake the rear wheels since the system is still able exert mechanical force on rear brake pads. The example in Fig. 23, instead, depicts another functional failure. In this case, the inability to exert both hydraulic and mechanical forces involve the impossibility to brake the entire vehicle.

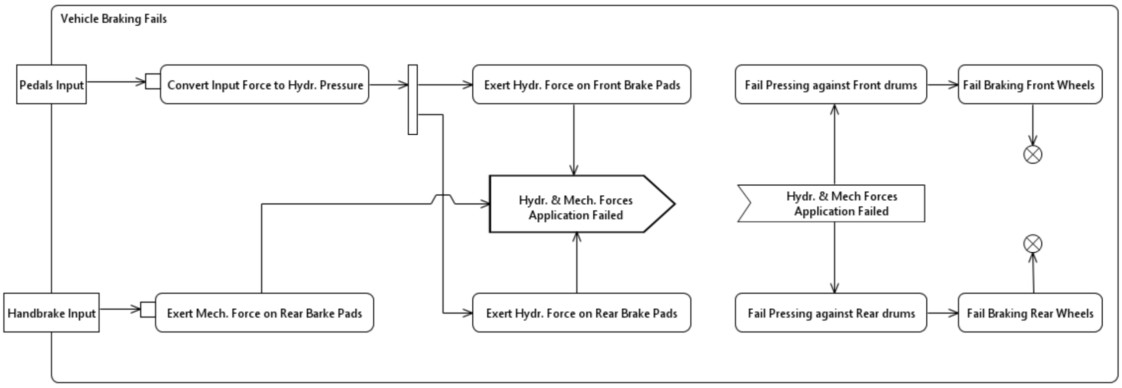

Vehicle Braking Fails
Pedals Input
Convert Input Force to Hydr. Pressure
Exert Hydr. Force on Front Brake Pads
Fail Pressing against Front drums
Fail Braking Front Wheels
Hydr. & Mech. Forces Application Failed
Hydr. & Mech Forces Application Failed
Handbrake Input
Exert Mech. Force on Rear Barke Pads
Exert Hydr. Force on Rear Brake Pads
Fail Pressing against Rear drums
Fail Braking Rear Wheels

Fig. 23: Simple braking system – 2nd functional failure example

The overall effect of each functional failure can be summarized in the *Activity* name. That is because the sequence of *actions* that follows the *accept event action* describes how the failure influences the system behavior. Consequently, the *Activity* name shall change with the aim of defining the modifications which it is representing.

3.1.6. Define hazard severity class and probability as *attributes* of the *signal*

The hazard severity class and probability related to each functional failure can be defined as *attributes* of the *signal* which represents the failure itself. The first describes the hazard severity, whereas the latter defines how frequently the hazard can happen. Their possible values are summarized in Fig. 3 and Fig. 4.

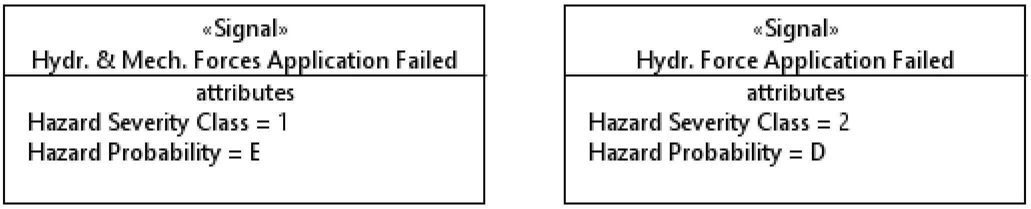
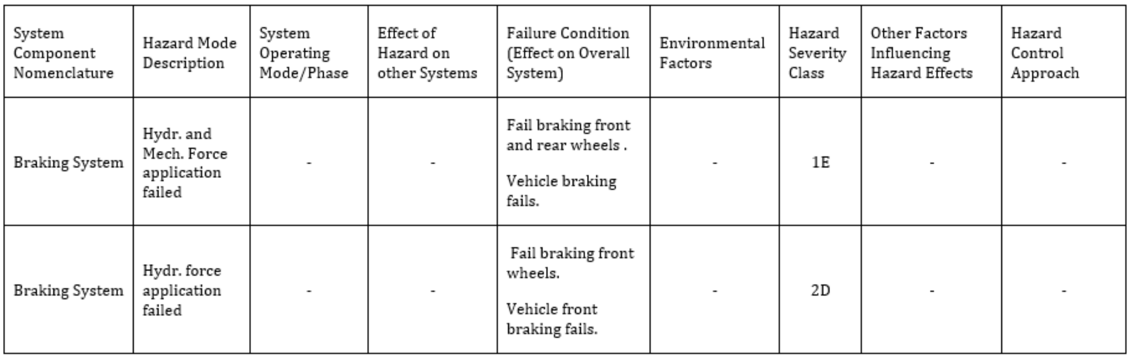

«Signal»
Hydr. & Mech. Forces Application Failed
attributes
Hazard Severity Class = 1
Hazard Probability = E
«Signal»
Hydr. Force Application Failed
attributes
Hazard Severity Class = 2
Hazard Probability = D

Fig. 24: Hazard severity class and probability represented as *attributes* of *signals*

Hazard severity class and probability can be depicted in a BDD. Specifically, they can be shown in the *attribute* section of their relative *signal*. In Fig. 24 are depicted both the *signals* used to represent functional failures in Fig. 22 and Fig. 23. The "Hydr. Force Application Failed" *signal* results being critical and remote. That means it may involve injuries or partial disability and even if unlikely, it is reasonably expected to occur. The "Hydr. & Mech. Forces Application Failed" *signal*, instead, results being catastrophic and improbable. That means it may involve death or severe injuries, but its occurrence is unlikely, even if possible.

The steps explained in this section allow representing functional failures and their effects using Activity Diagrams. The information provided by each of them can be used to compile a single row of a generic FHA worksheet. However, some data cannot be depicted only relying on Activity Diagrams. Information such as the item nomenclature and its related functions can be described using a BDD. Also, the system operating mode can be defined using a State Machine Diagram in which *states* are related to the *Activity* that represents the failure. Whereas the effect on other systems can be described relating the different Activity Diagrams. Another way to represent some of these data is the employment of *swimlines* to relate *actions* to the *Blocks* that perform them.



System Component Nomenclature	Hazard Mode Description	System Operating Mode/Phase	Effect of Hazard on other Systems	Failure Condition (Effect on Overall System)	Environmental Factors	Hazard Severity Class	Other Factors Influencing Hazard Effects	Hazard Control Approach
Braking System	Hydr. and Mech. Force application failed	-	-	Fail braking front and rear wheels . Vehicle braking fails.	-	1E	-	-
Braking System	Hydr. force application failed	-	-	Fail braking front wheels. Vehicle front braking fails.	-	2D	-	-

Fig. 25: Simple braking system document-based FHA

The presented method aims principally to allow performing an FHA using a model-based approach. That means using diagrams to represent functions and hazard modes instead of using documents that only list them. In this way, the comprehension of functions interactions and their possible failure effects should be more intuitive. In addition, this method gives also the possibility perform an FHA with a document-based approach. Indeed, it provides the necessary information required to compile a generic FHA worksheet (as shown in Fig. 25).

3.2. Model-based FTA

The graphical model used in the FTA is easily understandable. However, it is not as easier to develop. Indeed, defining faults and their logical relationships may become difficult, especially when architecture and functions of the system under analysis are not clear. Using a model-based approach, instead, would make more intuitive performing an FTA. That is because models – and the related diagrams – can provide a visual representation of system architecture and functions. To perform an FTA using a model-based approach it is necessary to follow different steps:

1. Define an *Interaction* and name it as the **top event**.
2. Represent the *Interaction* using a Sequence Diagram.
3. Use *reply messages* to describe faults occurrences.
4. Use *combined fragments* to represent logical gates (e.g. AND, OR etc.).
5. Use *guards* in *interaction operands* to describe faults.

These steps allow defining different Sequence Diagrams, which represent the occurrences of as much top events. The information they show are also useful to identify faults interactions and depict them into a fault tree.

3.2.1. Define an *Interaction* and name it as the top event

The combination of faults that may lead to an undesired event can be modeled in SysML using an *Interaction*. This element specifies how parts of a system (at any level of its hierarchy) should interact and how the system itself can interact with its environment. Moreover, it can be represented with a Sequence Diagram. The *Interaction* shall be named as the **top event**, since it results being the behavior involved by the combination of faults that constitute it. The context for the *Interaction* shall be an *instance* of the *Block* that owns it.

3.2.2. Represent the *Interaction* using a Sequence Diagram

Once the *Interaction* has been defined, it can be represented by means of a Sequence Diagram. This kind of diagram depicts a sequence of *messages* exchanged between the structural elements in the model. A *message* is depicted as an arrow and can represent an invocation for a service or the sending of a signal. Once a *message* has been received, it can trigger the execution of a behavior or it may be simply accepted. The exchange of *messages* occurs between *lifelines*, which represent the relevant lifetime of a *property* (either a *part property* or a *reference property*) of the *Interaction's* owning *Block*. The example in Fig. 26 shows a model-based FTA applied to a simple braking system. The *lifelines* in the diagram are used to represent parts of the system (such as the "Master Cylinder" or the "Cable System") along time. Whereas, *messages* are used to invoke *operations* (such as "Exert Mechanical Force") necessary to make the system work.

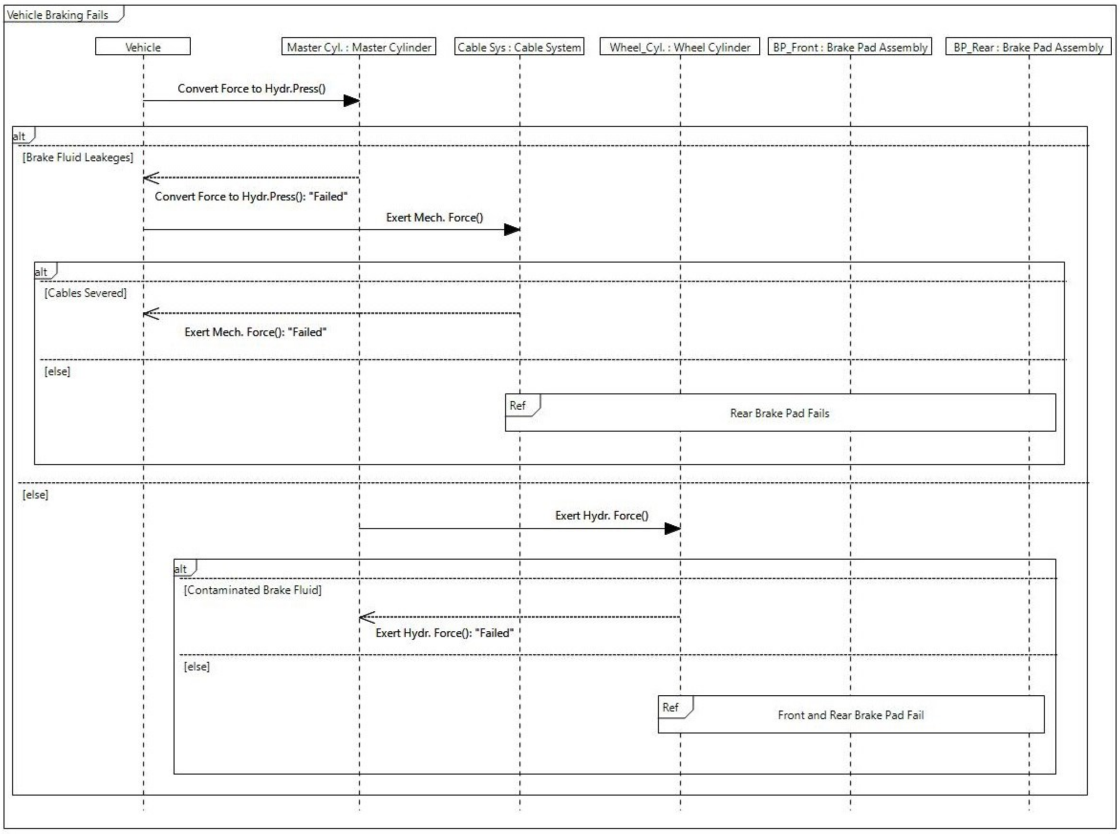

Vehicle Braking Fails
Vehicle
Master Cyl. : Master Cylinder
Cable Sys : Cable System
Wheel_Cyl. : Wheel Cylinder
BP_Front : Brake Pad Assembly
BP_Rear : Brake Pad Assembly
Convert Force to Hydr.Press()
alt
[Brake Fluid Leakeges]
Convert Force to Hydr.Press(): "Failed"
Exert Mech. Force()
alt
[Cables Severed]
Exert Mech. Force(): "Failed"
[else]
Ref
Rear Brake Pad Fails
[else]
Exert Hydr. Force()
alt
[Contaminated Brake Fluid]
Exert Hydr. Force(): "Failed"
[else]
Ref
Front and Rear Brake Pad Fail

Fig. 26: Simple braking system Sequence Diagram

3.2.3. Use *reply messages* to describe faults occurrences

In a Sequence Diagram *messages* can be used to invoke *operations* and describe the system intended behavior. However, in case of a fault occurrence the system behavior may change, involving a modification in the exchange of *messages* between *lifelines*. To represent a fault occurrence, it is possible to use *reply messages*. These elements are usually used as a reply to a *synchronous message* and are depicted as an open arrowhead on a dashed line. They are always sent from the *Lifeline* that performs a certain behavior to the *Lifeline* that invoked that behavior. Therefore, they can be used to represent a fault occurrence whenever the behavior that has been invoked cannot be accomplished. As an example, in Fig. 26 the "Vehicle" invokes the "Exert Mech Force" behavior to press the rear brake pads on the drum and brake. However, in case of severed cables, the "Cable System" is not able to accomplish that behavior. So, the fault occurrence is represented by means of a *reply message*, which highlights that the "Cable System" invoked behavior failed.

3.2.4. Use *combined fragments* to represent logical gates

In the fault tree model the relationships between events are represented by means of logical gates. Their features can be represented using *combined fragments*, which specify rules for the ordering of *messages* and their occurrences. The type of ordering logic is defined by *interaction operators*, whereas the *operands* identify the *messages* subject to that rules. SysML defines many *interaction operators*, but only *alt* and *par* have been considered to perform a model-based FTA. These two *interaction operators* allow to represent the OR and the AND logical gates. Specifically, the *par* can be used to represent the AND-gate. Since its *operands* can occur in parallel, the faults indicated using *reply messages* must occur simultaneously to make the next fault event happen. Whereas the *alt* can be used to represent the OR-gate. In this case, exactly one of its *operands* can occur, depending on the value of its *guard*. Therefore, one of the fault occurrences described within each *operand* will be enough to make the next event happen. The *alt interaction operator* can be also used to represent the AND-gate. Specifically, it can be done using the same kind of *operator* nested within the *operand* of another *alt*. In this way, it is possible to specify that fault occurs while another one is occurring. Therefore, them both must occur to make the next event happen. An example of this *alt operator* peculiar use is shown in Fig. 26. In case of "Brake Fluid Leakages" the master cylinder is not able to convert force into hydraulic pressure. Consequently, the vehicle demands to exert mechanical force on rear brake pads to the cable system. But, if cables result being severed, the vehicle is not able to brake. Both "Brake Fluid Leakages" and "Cable Severed" faults are necessary to reach the top event, and this implies that they must be connected using an AND-gate. Fig. 27, instead, provides an example of *par interaction operator* use. Wheel cylinders requests both front and rear brake pads to press against the respective drums simultaneously. However, if front and rear friction plates result being worn out it is not possible to brake. Therefore, both fault occurrences are necessary to reach the top event.

3.2.5. Use *guards* in *interaction operands* to describe faults

Each *combined fragment* used to represent a logical gate consists of an *interaction operator* and its *operands*. Each *operand* in turn has *guard*, a constraint expression that indicates the conditions under which it is valid. Whereas *reply messages* can be used to represent fault occurrences, their description can be defined using *operands guards*. The fault tree in Fig. 28 has been defined starting from the Sequence Diagram in Fig. 27. Both the basic events coincide with the *guards* related to the *operands* which contain the *reply messages* that highlight the fault occurrence.

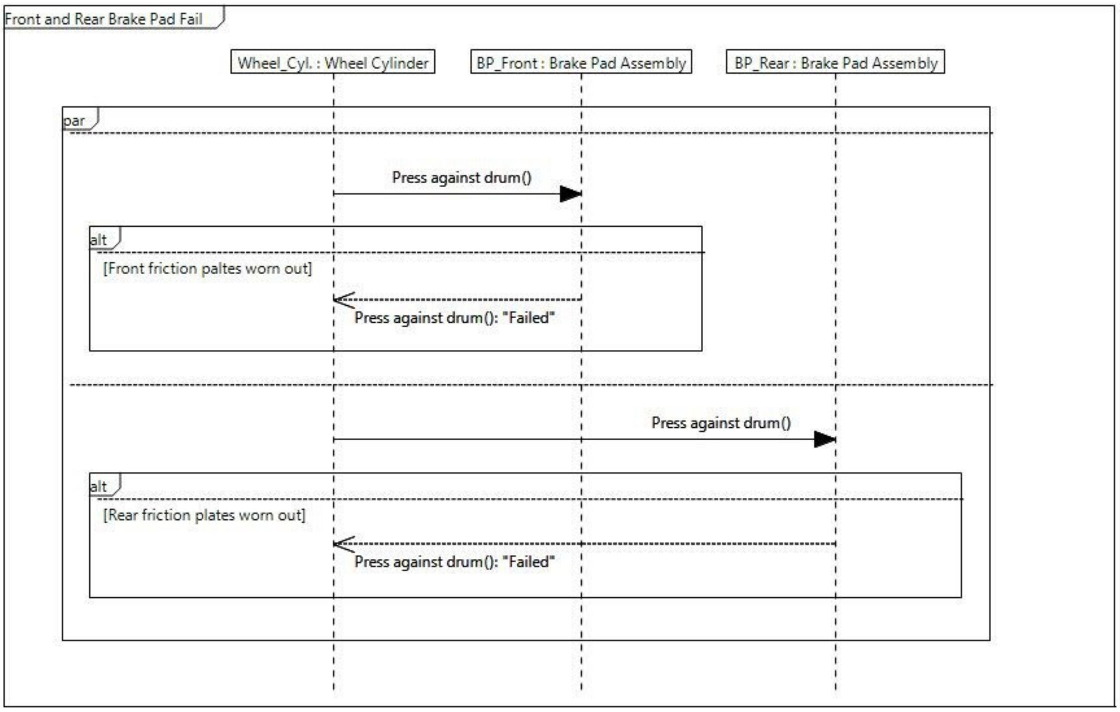

Front and Rear Brake Pad Fail
Wheel_Cyl. : Wheel Cylinder
BP_Front : Brake Pad Assembly
BP_Rear : Brake Pad Assembly
par
Press against drum()
alt
[Front friction paltes worn out]
Press against drum(): "Failed"
Press against drum()
alt
[Rear friction plates worn out]
Press against drum(): "Failed"

Fig. 27: Front and rear brake pads failure example

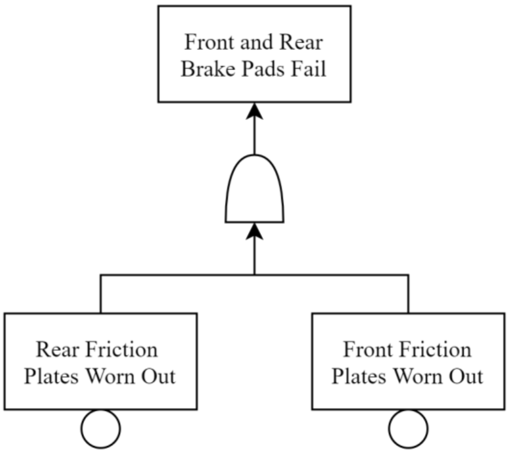

Front and Rear Brake Pads Fail
Rear Friction Plates Worn Out
Front Friction Plates Worn Out

Fig. 28: FTA of front and rear brake pads failure

In this model-based approach every fault described in the *Interaction* lead to the top event. Each *operand* of a *combined fragment* (containing a *reply message* describing a fault occurrence) represents an event connected to a logical gate (defined by the *interaction operator*). In case of nested *combined fragments*, their resulting events must be related to the faults defined in the *operand* to which they belong.

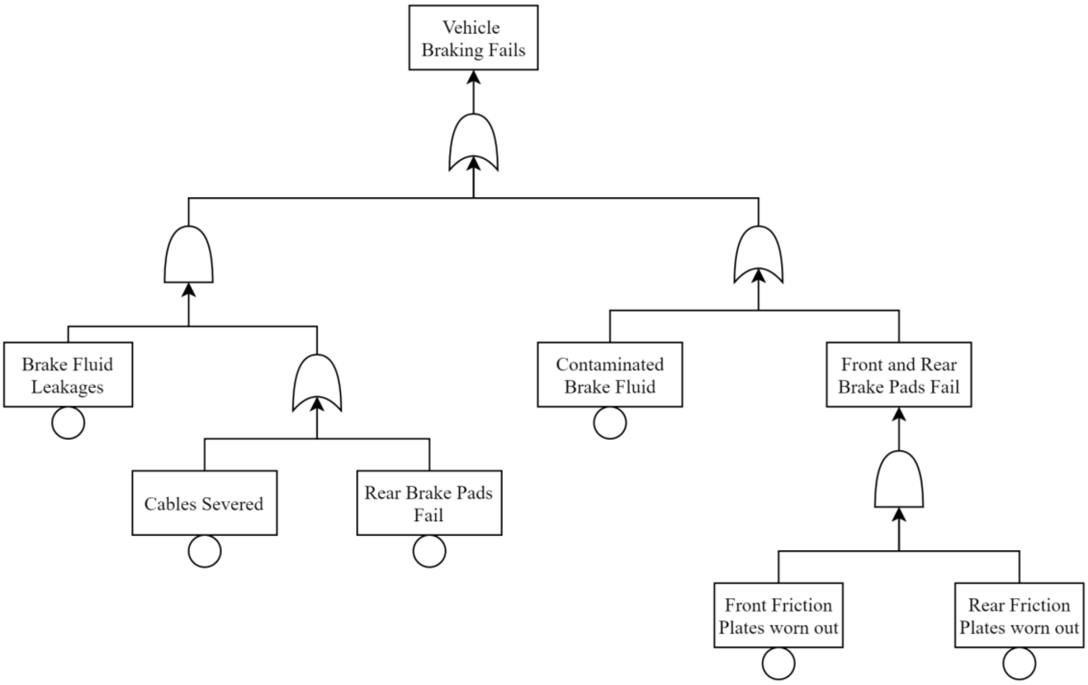


Fig. 29: Simple braking system document-based FTA

It is easy to notice that the fault tree model shown in Fig. 29 results being simpler than Sequence Diagram. However, it only shows information about faults and their consequences. Its relative SysML diagram, instead, describes the entire system behavior, considering both intended functions and the possible fault occurrences. In this way it is easier to understand which events may occur and to which undesired events they can lead.

3.3. Model-based FMEA

Before performing an FMEA using a model-based approach it is necessary to define as much as possible all the information about the system under analysis using models. Specifically, the system architecture shall be represented using the BDD and IBD diagrams; whereas the Activity Diagram, State Machine Diagram and Sequence Diagram shall be used to represent the system behavior, with more focus on what it is intended to do. If these data are not available, the model-based FMEA can be performed anyway; however, there might be some gaps and it might result inaccurate. The steps necessary to perform an FMEA following a model-based approach can be summarized in:

6. Define a *State Machine*
7. Develop a State Machine Diagram
8. Define a new *State Machine* to represent a failure
9. Identify the failure *state* using a *terminate pseudostate*
10. Detail the *transitions* that involve the failure occurrence
11. Describe failure mode and effects with *do behavior* and *exit behavior*

These steps allow defining different State Machine Diagrams, which describe different failure modes, their causes, and their effects. Each diagram contains the fundamental information necessary to compile a generic FMEA worksheet.

3.3.1. Define a *State Machine*

To perform an FMEA with a model-based approach it is first necessary to define a *State Machine*. This kind of SysML element is used to describe the behavior of a *Block* in terms of *states* in which it can be and *transitions* that may bring from one *state* to another. When the *Block* is in a *state*, it can perform different sets of actions.

3.3.2. Develop a State Machine Diagram

The next step is to develop a State Machine Diagram which can represent the *State Machine* defined before. The diagram shall depict the different *states* which the considered *Block* can assume and the *transitions* that may cause its *state* changing. An example of State Machine Diagram is reported in Fig. 30, concerning a brake pad assembly.

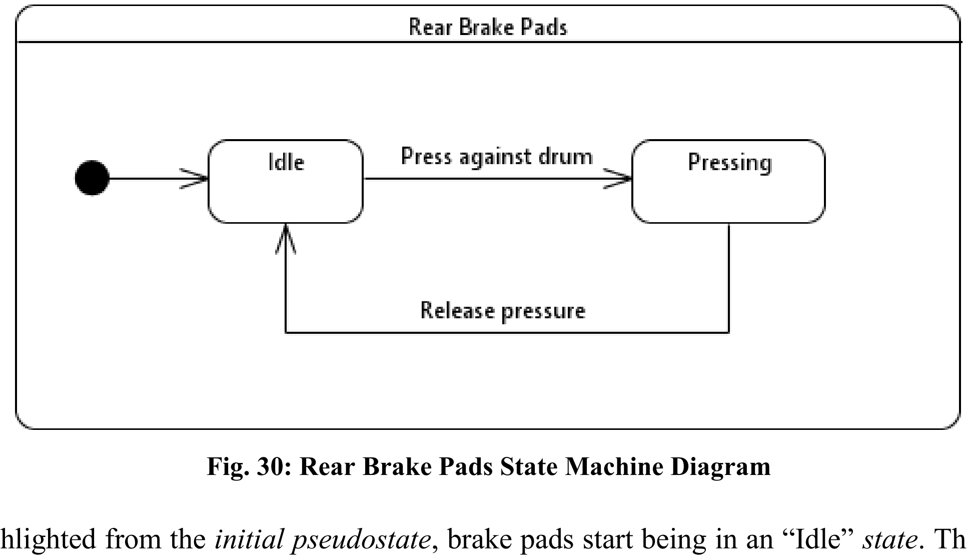


Fig. 30: Rear Brake Pads State Machine Diagram

As highlighted from the *initial pseudostate*, brake pads start being in an "Idle" *state*. Therefore, they are not being used. However, a *transition* changes their *state* in "Pressing" after receiving the request of pressing against the drum. Their *state* then goes back to "Idle" when is requested to release the pressure.

3.3.3. Define a new *State Machine* to represent a failure

Starting from the *State Machine* defined before, it is necessary to create another one. This new element shall have the purpose of representing the *Block* behavior in case of a failure. Therefore, it shall contain all the *states* and *transitions* shown in the previous State Machine Diagram. In addition, it shall contain a new *state* which represents a possible failure.

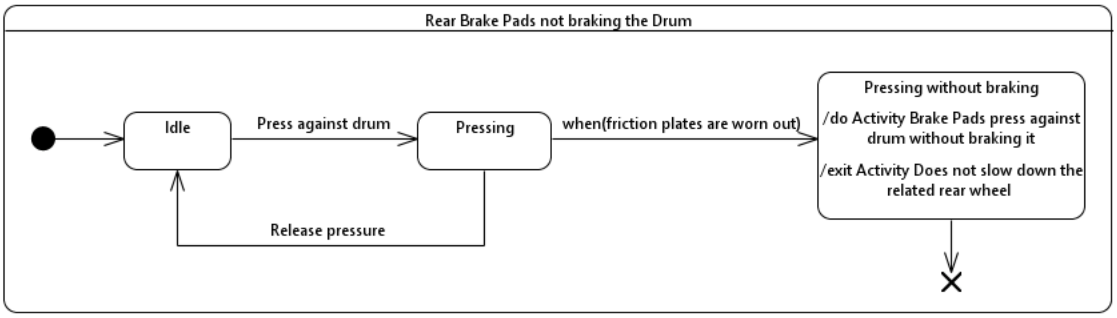

Rear Brake Pads not braking the Drum
Idle
Press against drum
Pressing
when(friction plates are worn out)
Pressing without braking
/do Activity Brake Pads press against drum without braking it
/exit Activity Does not slow down the related rear wheel
Release pressure

Fig. 31: Rear Brake Pads failure example

The State Machine Diagram in Fig. 31 shows a behavior similar to one presented in Fig. 30. However, in this case, a new *state* which represents a failure is depicted. Indeed, if brake pads friction plates result being worn out, it is not possible to brake the related wheel.

3.3.4. Identify the failure *state* with a *Terminate pseudostate*

To identify the *state* that represents the failure, it is necessary to use a *Terminate pseudostate*. When this entity is reached the behavior of the entire *State Machine* terminates. Its use allows even depicting the influence of failure on the *Block* behavior, since it will not be able to behave as intended anymore.

3.3.5. Detail the *transitions* that involve the failure occurrence

The *transitions* that involve the failure state shall be modeled and represented in the State Machine Diagram. Those entities specify when a change of state occurs. They are characterized by:

- *Trigger*: define the events that cause the transition.
- *Guard*: define the constraint to respect to make the transition occur.
- *Effect*: define the behavior executed during the transition.

The first one is essential to perform a model-based FMEA since it defines the failure causes. For this reason, *triggers* shall be described for each *transition* that involves a *state* which represents a failure. *Guards*, instead, can be useful to add more information about the constraints necessary to make the failure happen. The example in Fig. 31 depicts only the *trigger*. It is characterized by a *change event,* which highlights that the transition occurs when the friction plates result being worn out. No *guard* and *effect* have been defined in this case.

3.3.6. Describe failure mode and effects with *do behavior* and *exit behavior*

While being in a state, a *Block* can perform different actions. Each *state* can describe three different kinds of behavior:

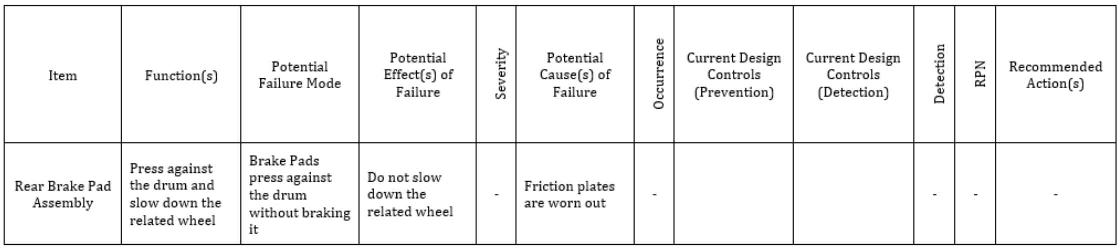
- *Entry behavior*: is performed whenever the *state* is entered
- *Do behavior*: is performed after the *entry behavior* and until it completes (or the *state* is exited)
- *Exit behavior*: is performed whenever the *state* is exited

Specifically, the last two are useful to define other details about the failure. The *do behavior* can be used to describe the failure mode, whereas the *exit behavior* can be used to describe the failure effects. In Fig. 31, is detailed that the brake pads are not able to brake the drum, even if they are pressing on it. Consequently, they are not able to slow down their related wheel.

A State Machine Diagram developed in this way contains information about the considered failure, its cause, and effects. Therefore, each diagram can be used to fill a specific row of a FMEA worksheet. However, some data – which the State Machine Diagram cannot provide – can be detailed in other diagrams. All the necessary information which can be gained with a model-based approach to perform a document-based FMEA, can be listed as follows:

a. **Item**: is a *Block* related to the considered *State Machine*; it can be represented in a BDD.
b. **Function**: can be an *Action*, *Activity*, *Operation* or *Reception* related to the item defined before; it can be represented in an Activity Diagram or in a BDD as an *Owned Behavior* or *Nested Classifier* of the considered *Block*.
c. **Failure Mode**: is the *do behavior* defined in the *state* which represents the failure.
d. **Failure Causes**: are the combination of *triggers* and *guards* that characterize the *transitions* which involve the failure *state*.
e. **Failure Effect/Corrective Action**: is the *exit behavior* defined in the *state* which represents the failure.

Unfortunately, the proposed model-based technique cannot provide all the information defined in a FMEA worksheet. In Fig. 32 is shown an example of a document-based FMEA that can be obtained after using a model-based approach.



Item	Function(s)	Potential Failure Mode	Potential Effect(s) of Failure	Severity	Potential Cause(s) of Failure	Occurrence	Current Design Controls (Prevention)	Current Design Controls (Detection)	Detection	RPN	Recommended Action(s)
Rear Brake Pad Assembly	Press against the drum and slow down the related wheel	Brake Pads press against the drum without braking it	Do not slow down the related wheel	-	Friction plates are worn out	-			-	-	-

Fig. 32: Rear Brake Pads document-based FMEA

The presented model-based approach to perform an FMEA has principally two main goals. The first is to gather most of the information about failures in a single model and represent them with different diagrams. This makes that data always available and easier to understand. The latter is to give the possibility to perform anyway a document-based FMEA, but more intuitively thanks to all the information provided by the model.

3.4. Model-based RBD

The RBD is an inductive model which represents distinct elements by means of blocks, combined according to different success pathways. One of its characteristics is that relies on the interrelationships between elements instead of focusing on possible failures that may occur. For this reason, it is necessary to highlight how the components that characterize system interact one with another. SysML language does not provide a diagram capable of representing the RBD as defined by document-based approach. However, it is possible to depict the connections between system components by means of an Internal Block Diagram (IBD). The interactions described in the IBD can be used to define an RBD capable of representing the system reliability.
The different steps necessary to represent an RBD following a model-based approach can be summarized in:

1. Define a BDD to represent the system under analysis, its components and other elements which interact with it.
2. Define an IBD to represent the interactions between system components.
3. Define the connections among components using *connectors.*
4. Define the *multiplicity* of *part properties* and *connectors* ends.
5. Use the IBD to identify RBD blocks and success pathways.

Following these steps, it is possible to develop an IBD capable of providing the necessary information to understand the system functioning and consequently to easily define an RBD.

3.4.1. Define a BDD to represent the system under analysis

The first step is the development of a BDD, which is useful to represent the structure of system under analysis. In this diagram the system itself and its components can be depicted by means of *Blocks*, which are then connected using different kind of *Associations*. These elements of SysML language are essential to define the system hierarchy in a BDD. The diagram in Fig. 33 represents an example of BDD. It focuses on a simple braking system, which have been modeled using a *Block*. Instead, *Composite Associations* have been used to define the relationships present between the system and its components. This specific kind of *Association* is characterized by a line adorned with a black diamond and an open arrow on its ends. The first describes the pointed element as whole, whereas the latter describes the pointed element as a part. As an example, in Fig. 33 the "Braking System" is the whole and the "Master Cylinder" is one of its parts. Another feature of *Composite Associations* are the names and multiplicities shown on their ends. Specifically, the ones on the ends adorned with the open arrow indicate the *part properties* of the element connected on the other end. Those *properties* are essential since they will be used in the IBD to represent the relationships among parts.

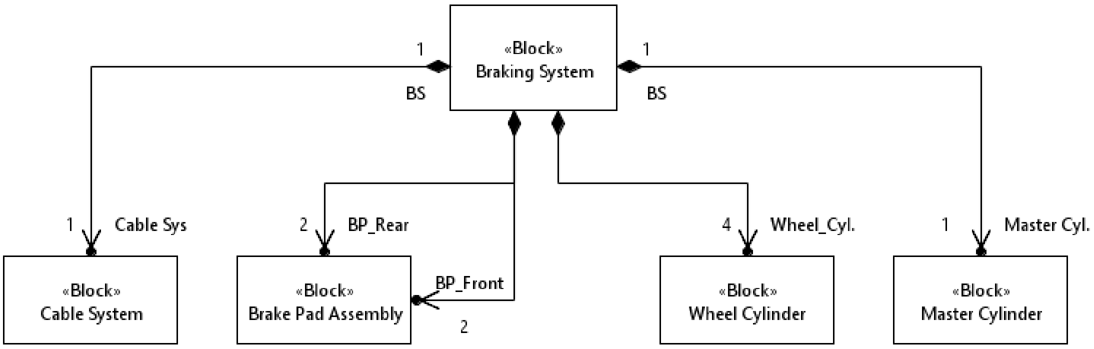


Fig. 33: Simple braking system BDD

An example in Fig. 33 is the "Brake Pad Assembly," which is represented with a *Block*. This element is connected to the "Braking System" by means of two *Composite Associations*, that specify two different *part properties*: the front brake pads and the rear ones. The multiplicity depicted near their name indicates the number of *instances*, which in this case is "2" for both front and rear brake pads.

3.4.2. Define an IBD to represent the interactions between components

After defining the system hierarchy by means of a BDD, it is necessary to identify and represent the interactions between the components that characterize that system. This procedure can be done using an IBD. The example in Fig. 34 shows how the different *part properties* that belong to the "Braking System" *Block* are connected one with another.

3.4.3. Define the connections among components using *connectors*

SysML language makes available different instruments to represent the connections between parts of a *Block*. In most of the cases it is also possible to provide information about what the *part properties* exchange one with another. However, to define a diagram capable of providing the information necessary for developing an RBD, only *connectors* shall be used. In this way, the resulting IBD will be simpler, highlighting the interactions between the *Block* parts. In case of complex systems, the only use of *connectors* might be not enough to understand how to define an RBD. That is the reason why it is possible to create another IBD containing more elements (such as *flow ports*, *full ports*, etc.) to support the RBD development.

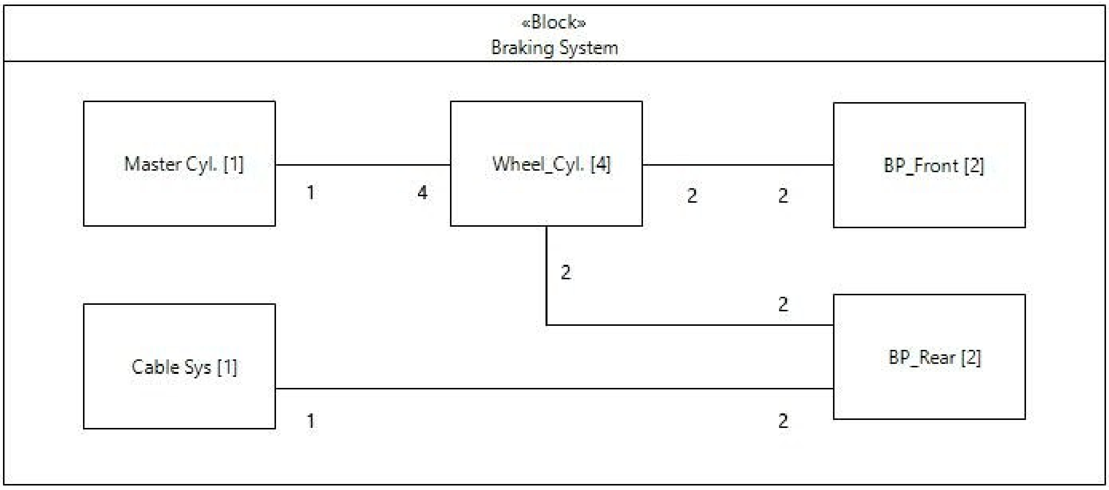


Fig. 34: Simple braking system IBD

3.4.4. Define the multiplicity of part properties and connectors ends

Once the connections between the different *part properties* have been created, it is necessary to define their *multiplicity*. This parameter is usually defined after the employment of a *Composite Association* in a BDD and indicates the number of *instances* of a *part property*. *Multiplicity* shall be defined also for *connectors*. The number that adorns each one of their ends indicates how many *instances* can be connected by links described by the *connector*. The example in Fig. 34 shows that there is only one *connector* between the cable system and the rear brake pads. However, the *multiplicities* on its ends indicate that the only existing cable system *instance* is linked with two separate rear brake pads *instances* (which may be the right and the left rear brake pads). The same can be said about the wheel cylinder, which has a *multiplicity* of four. Two of its *instances* are linked to the front brake pads ones and the other two are linked to the rear brake pads ones.

3.4.5. Use the IBD to identify RBD blocks and success pathways

After defining the IBD it can be used to develop an RBD which may allow to calculate the considered system reliability. The *part properties*, *reference properties* and their relative *instances* can be used to define the RBD blocks. Whereas, *connectors* and *multiplicities* on their ends can be used to identify the success pathways. The diagram in Fig. 35 depicts an RBD created starting from the IBD shown in Fig. 34. It is possible to notice that the wheel cylinder and brake pads *instances* have been considered to represent different blocks. Indeed, the RBD consists of four wheel cylinders, two front brake pads and two rear brake pads. The success pathways, instead, have been defined starting from the *connectors* and the *multiplicities* on their ends; so, each wheel cylinder results being linked to a specific brake pad.

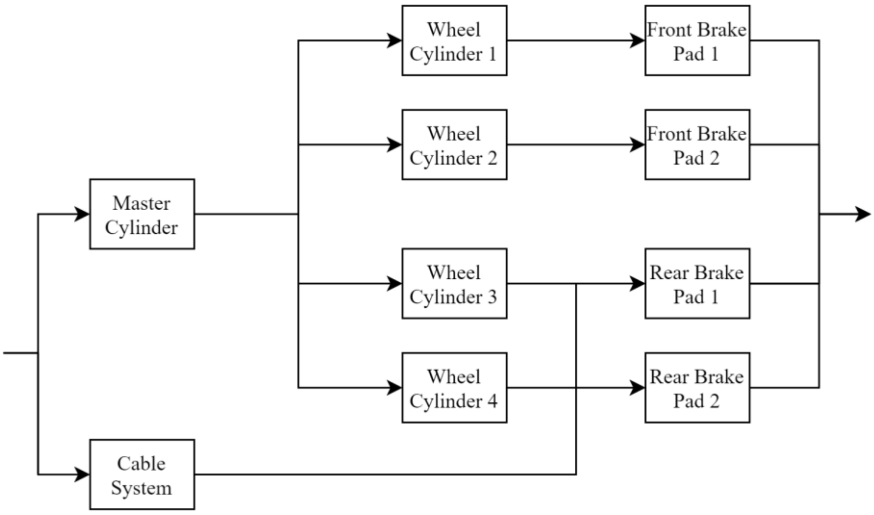


Fig. 35: Simple braking system document-based RBD

One lack in SysML language is the inability to specify which *instances* are linked together in an IBD, especially when using *connectors* with *multiplicities* greater than "1" on their ends. As an example, the IBD in Fig. 34 does not specify which wheel cylinder *instance* is connected to front or rear brake pads *instances*. If it is necessary to represent some specific connections between components, it is preferable to define specific *part properties*. An example are the brake pads, which have been modeled as two *parts* to highlight the interaction of the rear ones with both wheel cylinders and cable system.

Once the RBD has been define it is possible to evaluate the reliability of system under analysis. A change in its employed components or in its architecture may involve a different reliability value. This is an important aspect to take in account during the development of innovative system architectures. Indeed, a change in reliability may influence maintenance hours, maintenance costs and operating costs. In addition, if taken in account during the architecture development, it may also bring variations to some design parameters (such as MTOW, MEW, Fuel consumption, etc.). For this reason, reliability calculation shall be integrated in an MDO environment, which takes in account different disciplines simultaneously to design new on-board systems architectures. However, it will be necessary to develop a tool capable of defining reliability from an RBD and capable of considering the modifications involved by architecture changes. Chapter 4 focuses on the description of this tool.

4. Integration with the MDO environment

The model-based RAMS analyses presented in chapter 3 enable to efficiently evaluate both safety and reliability of conventional and innovative systems. Their integration into an MDO environment would also allow evaluating the impact of these systems on performance and costs of the aircraft on which they are employed. Therefore, it would be necessary to develop a tool capable of analyzing systems architectures and defining their dependability. Moreover, it shall also be able to cooperate with the other tools that compose the MDO environment. This kind of instrument already exists and allows defining systems reliability, maintenance hours, maintenance costs and operative costs. However, it relies on the statistical model described in section 2.3 and follows a top-down approach. This makes quite difficult its use for innovative on-board systems since statistical data about them are not available. Moreover, a bottom-up approach would be more suitable due to the architecture changes applied. Therefore, it has been necessary to develop a new tool, capable of defining on-board systems dependability while taking in account modifications in their architectures.

4.1. Tool inputs

The tool aim is to evaluate systems reliability value, starting from an RBD which represents them. To do this, it requires different input data. First, it is necessary to define the reliability of all the blocks that compose the RBD. Then, the connections between those blocks shall be also defined.

4.1.1. Components Reliability

The reliability of the components that constitute the RBD can be evaluated starting from their failure rate function (usually represented with λ). Their relationship can be expressed as follows:

$$R(t) = exp\left[-\int_0^t \lambda(t')\, dt'\right] \tag{4.1}$$

The failure rate function is characteristic for each component. It indicates how many failures may occur during its lifetime and its trend depends on the type of failures experienced by the component during its use (e.g. early failures, random failures, wearout failures, etc.). One of the most important form that failure rate function can assume is the *bathtub curve*, shown in Fig. 36. It is widely used in aerospace field and is characterized by the combination of three different parts:

1. **Burn-in**: the failure rate has an initial a high value, which then decreases over time. This behavior is typical of new-born components that result being unreliable until they

are tested. The failures that occur during this time span are early failures (also called "Infant Mortality" failures).

2. **Useful Life**: the failure rate has a low and constant value. In this part of the curve the component is usually used to perform the functions for which has been designed. The failures that occur during this period are random failures.
3. **Wearout**: the failure rate has an initial low value, which then increases over time. This behavior is typical of components which result being worn out after long period of usage. The failures that occur during this time span are called wear-out failures.

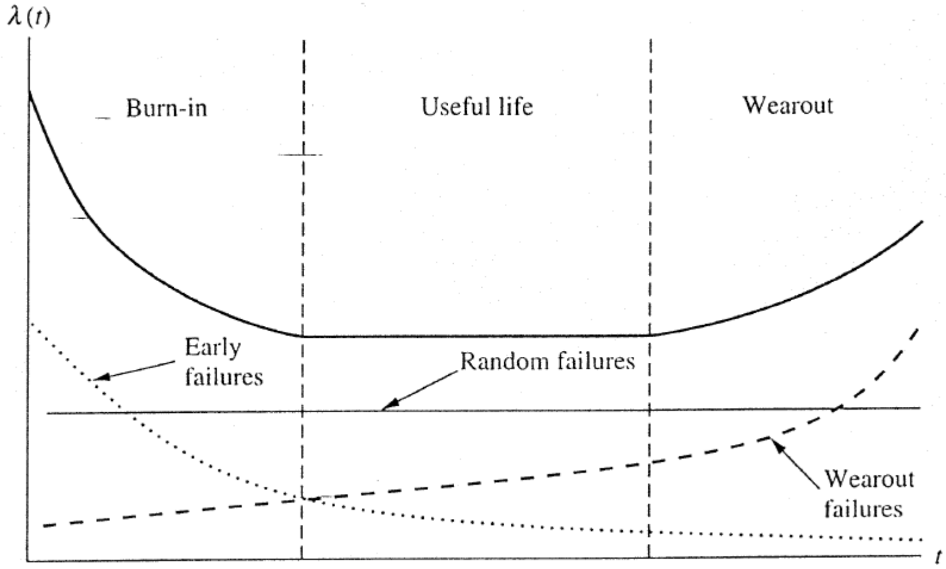


Fig. 36: Bathtub curve [44]

The bathtub curve has been chosen to represent the components failure rate over time. However, its mathematical expression shall be defined to make it available as an input data for the tool. To do this, the Weibull distribution can be used. It is one of the most useful probability distribution, which can be used to model increasing, decreasing and constant failure rates. The mathematical expression of failure rate characterized by a Weibull distribution is:

$$\lambda(t) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1} \tag{4.2}$$

In this equation t represents the time, β is referred to as shape parameter and θ is a scale parameter. The last two must be always greater than zero, whereas time must be greater or equal to zero. Depending on the shape parameter value, the failure rate distribution can be modified as follows:

- $\beta < 1$: the failure rate distribution follows a decreasing trend.
- $\beta = 1$: the failure rate distribution remains constant.
- $\beta > 1$: the failure rate distribution follows an increasing trend.

The scale parameter, instead, influences both the mean and the spread of the distribution. It is then possible to define the reliability mathematical expression by substituting Eq. 4.2 in Eq. 4.1:

$$R(t) = e^{-\left({t}/{\theta}\right)^{\beta}} \tag{4.3}$$

If the shape parameter is set to unitary value, the reliability equation becomes an exponential function. Moreover, the failure rate distribution results being constant and inversely proportional to the scale parameter. Therefore, Eq. 4.2 and Eq. 4.3 becomes respectively:

$$\lambda(t) = {1}/{\theta} \tag{4.4}$$

$$R(t) = e^{-\lambda t} \tag{4.5}$$

To model a failure rate distribution capable of following the bathtub curve trend it is necessary to combine three different Weibull distribution functions. Those functions can be defined by providing time span, shape parameter and scale parameter for each one of them. Therefore, these data must be provided as an input to the tool, so that it can define the failure rate distribution for each component considered in the RBD. The input necessary to make the tool work must be composed by three different vectors:

- $t = \{t_1, t_2, t_3\}$: shall contain respectively the times – defined in Flight Hours (FH) – at which decreasing, constant and increasing distributions of failure rate end. Whereas, times in which they begin are automatically defined in the tool.
- $\beta = \{\beta_1, \beta_2, \beta_3\}$: shall contain respectively the shape parameters which describe decreasing, constant and increasing distributions of failure rate (consequently, β_1 shall be minor than one, β_2 shall have unitary value and β_3 shall be greater than one).
- $\theta = \{\theta_1, \theta_2, \theta_3\}$: shall contain respectively the scale parameters which concern the decreasing, constant and increasing distributions of failure rate.

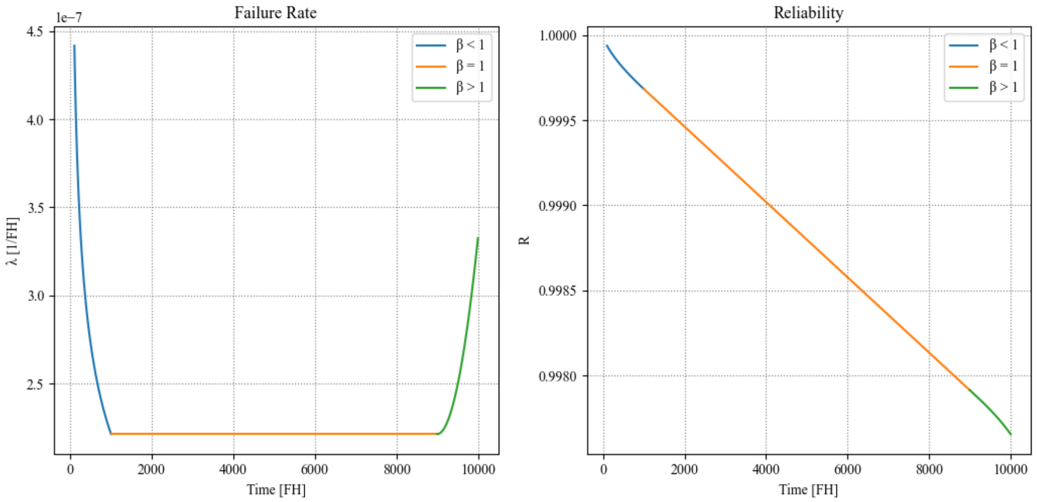


Fig. 37: Failure rate and reliability distribution over time example

After receiving these input data, the tool can define and combine the three different functions which allow to represent the failure rate distribution with the bathtub curve trend. Consequently, it is also capable of defining the reliability trend over time. An example of the resulting distributions is shown in Fig. 37. It can be noticed that reliability always decreases over time, even if its curve form changes. In addition, the elements that compose vectors t, β and θ can be changed to model the failure rate distribution and give it shapes also different from the one of the bathtub curve.

4.1.2. System Architecture

After defining the reliability of the components that constitute the RBD, it is necessary to analyze their connections. A simple and efficient way to represent those connections is using a matrix, which allows to make them available as an input for the tool. The mentioned matrix shall be square, whereas its number of rows shall be equal to the number of blocks depicted in the RBD. Each row represents one of these blocks and the same is valid for columns. Each element of a row represents a possible connection with the component of the respective column. Specifically, the matrix is composed only by ones and zeros. The presence of a "1" indicates that there is a connection between the block represented by the row and the one represented by the column. The presence of a "0" instead, indicates that there is not a connection. An example of this kind of matrix is shown in Fig. 38. It represents the connections between the blocks of the RBD depicted in Fig. 35, concerning a simple braking system.

$$
\begin{array}{r|cccccccccc|}
 & \text{Master Cylinder} & \text{Cable System} & \text{Wheel Cylinder 1} & \text{Wheel Cylinder 2} & \text{Wheel Cylinder 3} & \text{Wheel Cylinder 4} & \text{Front Brake Pad 1} & \text{Front Brake Pad 2} & \text{Rear Brake Pad 1} & \text{Rear Brake Pad 2} \\
\text{Master Cylinder} & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\text{Cable System} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\text{Wheel Cylinder 1} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\text{Wheel Cylinder 2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\text{Wheel Cylinder 3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\text{Wheel Cylinder 4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\text{Front Brake Pad 1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{Front Brake Pad 2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{Rear Brake Pad 1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{Rear Brake Pad 2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

Fig. 38: Index matrix defined from the simple braking system RBD

It is important to highlight that the numbers “1” present on rows shall indicate only connections that go from the left to the right side of the diagram. As an example, the first row in the matrix indicates the four connections of the master cylinder with the respective wheel cylinders. The rows which represent the wheel cylinders, instead, indicate only their connections with brake pads and not the one with the master cylinder. This way of modeling the matrix makes it less chaotic and more readable. Moreover, the connections that go from the right to the left side of the RBD results being already defined. Indeed, considering the numbers “1” on the columns that represent the wheel cylinders, they indicate the connection with master cylinder.

The use of a matrix to represent and analyze an RBD diagram has been already taken in account in [45]. In this case the matrix has been developed starting from different IBDs. Afterward, it has been used to define the corresponding RBD following a process explained in the same document. However, a tool capable of analyze that matrix to calculate the overall reliability has not been created.

4.2. Tool operation

After defining vectors t, β and θ for each block that compose the RBD and the matrix that represents its architecture, it necessary to save these data in a CPACS file. This specific kind of XML file has been developed by DLR and enables tools integrated into an MDO environment to exchange information [46]. When the CPACS file has been updated with the new input data, the tool can be operated. Its functioning has been represented using a flowchart, which is shown in Fig. 39. First, the tool opens and reads the CPACS file to get the information it requires. One of these information is the *specified time*, that indicates the time span to use for integrating the failure rate of each component and consequently calculating its reliability. Afterwards, it gets the number of systems and starts analyzing them one by one. For each system, the tool gets the number of its components and starts acquiring the vectors t, β and θ relative to each of them. Then, by means of *specified time*, it calculates the reliability of all the components and saves them in a specific vector. After doing this, the tool shall acquire the matrix which describes the RBD configuration. However, it necessary to consider that reliability can be calculated for a specific mission, which is composed of different phases. During each one of them the RBD configuration may change due to some failures or some changes in system functioning. Consequently, it may be necessary to consider different matrices that represent the RBDs of each mission phase. Therefore, the tool can define the number of architectures that need to be analyzed. For each of them, it acquires the associated matrix and uses it to evaluate the mission phase reliability. All the results are then saved in a vector. Finally, the product of its elements gives back the overall mission reliability. This procedure is carried out for each system that the tool analyzes.

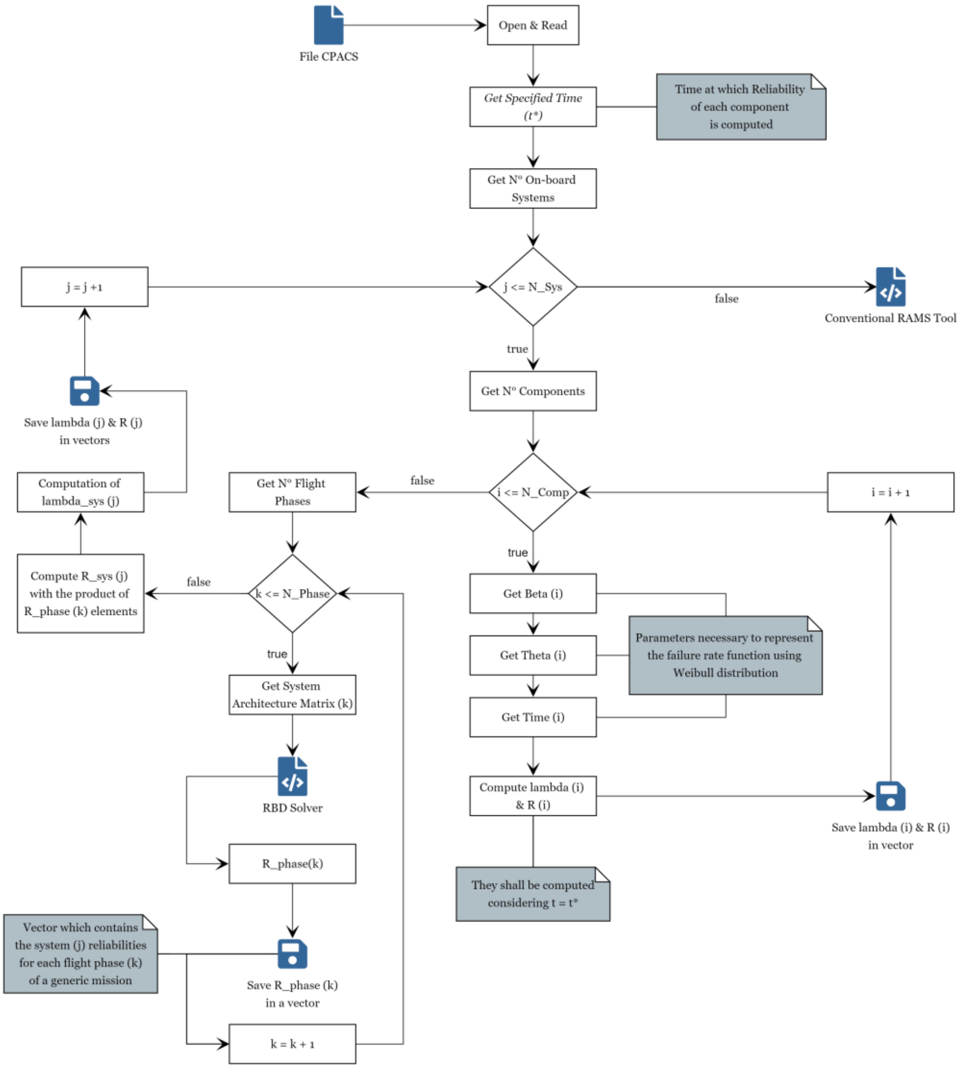

File CPACS
Open & Read
Get Specified Time (t*)
Time at which Reliability of each component is computed
Get N° On-board Systems
j = j +1
j <= N_Sys
false
Conventional RAMS Tool
true
Get N° Components
Save lambda (j) & R (j) in vectors
Computation of lambda_sys (j)
Get N° Flight Phases
false
i <= N_Comp
i = i + 1
true
Compute R_sys (j) with the product of R_phase (k) elements
false
k <= N_Phase
Get Beta (i)
Get Theta (i)
Parameters necessary to represent the failure rate function using Weibull distribution
Get Time (i)
true
Get System Architecture Matrix (k)
RBD Solver
Compute lambda (i) & R (i)
Save lambda (i) & R (i) in vector
R_phase(k)
They shall be computed considering t = t*
Vector which contains the system (j) reliabilities for each flight phase (k) of a generic mission
Save R_phase (k) in a vector
k = k + 1

Fig. 39: Flowchart describing the tool operation

The reliability evaluation in the tool is performed with a specialized function called *RBD Solver*. It requires both the matrix representing the RBD configuration and the vector containing reliability values of blocks that compose it. With these two inputs it can analyze the provided architecture and calculate its resulting reliability. However, this function is limited since it is still not able to analyze complex configurations. Therefore, the reliability evaluation concerning architectures such as the one shown in Fig. 35 will result in an error. A simple way to avoid this problem is to modify the RBD so that it can be analyzed by the tool. The diagram in Fig. 40 is the modified version of the one in Fig. 35. It is characterized only by series and parallel configurations and is appropriate for the reliability calculation with tool. Obviously, the resulting reliability value will be different since the modified RBD does not consider the effect of common cause failures. But, in a first approximation, it can effectively represent how much reliable the considered system architecture can be.

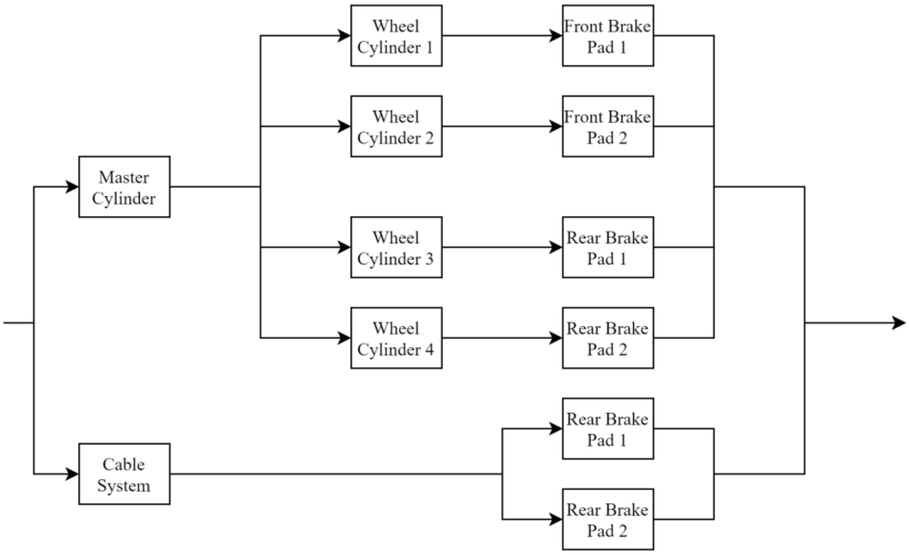


Fig. 40: Modified simple braking system RBD

After calculating the reliability of all the systems listed in the CPCAS file, the tool evaluates their failure rate using the inverse of Eq. 4.5. Both vectors containing reliabilities and failure rates are then saved in a specific section of the CPACS file. Furthermore, the second one also given as an input to another tool. This one has been developed in DLR during another project development and follows the statistical model described in Section 2.3 to evaluate reliability, safety, maintenance costs and operative costs of conventional systems architectures. In addition, it can also evaluate the overall aircraft failure rate after considering the introduction of new technologies (such as laminar flow wing, composite structures or EHAs). However, it does not take in account the possible changes in systems architectures and the introduction of innovative ones. The tool described in this chapter, instead, considers them. Therefore, the combination of its results with the data provided by the conventional RAMS tool enables the evaluation of dependability even for aircraft with innovate on-board systems architectures.

5. Test Case

The methodology described in Section 3 is now employed to define and improve the dependability of an FCS, considering both conventional and innovative architectures. The reference aircraft – on which the selected on-board system is implemented – is the Airbus A320. A representative 3D model is shown in Fig. 41. The main requirements and specifications concerning the reference aircraft are collected in Table 1. The FCS under analysis is characterized by a conventional architecture which refers to the one implemented on the Airbus A320 family. Starting from this, two innovative architectures have been defined and then analyzed using a model-based approach.

Table 1: Baseline aircraft main design parameters

Aircraft performance and capacity		
Range	[km]	6200
Maximum Operating Altitude	[m]	11918
Take-off (ISA, Seal Level, MTOW)	[m]	2100
Landing (ISA, Seal Level, MTOW)	[m]	1500
Passengers	[-]	150 – 186
Maximum Payload Weight	[kg]	19900
Aircraft masses		
Maximum Take-off Weight (MTOM)	[kg]	73500
Maximum Landing Weight (MLW)	[kg]	64500
Maximum Zero Fuel Weight (MZFW)	[kg]	60500
Operative Empty Weight (OEM)	[kg]	42600
Fuselage geometrical data		
Fuselage length	[m]	37.57
Fuselage width	[m]	4.14
Cabin width	[m]	3.63
Wing geometrical data		
Wingspan	[m]	34.1
Wing area	[m^2]	124
Wing aspect ratio	[-]	10.3
Wing sweep	[deg]	25
Propulsion system data		
Engines Thrust	[kN]	90 – 120

Fig. 41: Reference aircraft 3D model

5.1. Flight Control System – Overview

The Flight Control System (FCS) is an essential part of any aircraft since it allows to control both translational and rotational motion. By means of control surfaces it can modify the aerodynamic forces acting on the aircraft and generate aerodynamic torques which change its attitude. Typical flight control surfaces implemented on commercial airliners are:

- **Ailerons**: generate torque around the aircraft longitudinal axis (they act jointly and anti-symmetrically).
- **Elevators**: generate torque around the aircraft lateral axis.
- **Rudders**: generate torque around the aircraft vertical axis.
- **Flaps**: greatly increase lift for a given speed during take-off and landing phases.
- **Slats**: greatly increase lift for a given speed during take-off and landing phases.
- **Spoilers**: increase drag and reduce lift; they can be used symmetrically to reduce aircraft speed and brake or asymmetrically (only on one side of the wing) to complement the ailerons function.
- **Trimmable Horizontal Stabilizer (THS)**: maintains horizontal static equilibrium and stabilize the aircraft in the pitch axis if used in combination with the elevators.

An example of these flight control surfaces and their arrangement on a commercial aircraft is shown in Fig. 42. The first three are called Primary Flight Controls and provide respectively roll, pitch and yaw control. The other surfaces, instead, are called Secondary Flight Controls and modify the aircraft macroscopic aerodynamic characteristics, providing high lift generation and drag increase. Moreover, some of those (such as Spoilers and THS) can be used to complement some Primary Flight Controls functions.

The FCS conventional architecture is based on traditional hydro-mechanical systems, which implement hydraulic piston actuators to move the control surfaces. Pilot commands are electrically transmitted to the hydraulic actuators through wires (Fly-By-Wire system). Flight Control Computers (FCCs) determine how to move the control surfaces and control the hydraulic actuator, in order to accomplish the commanded movements [47].

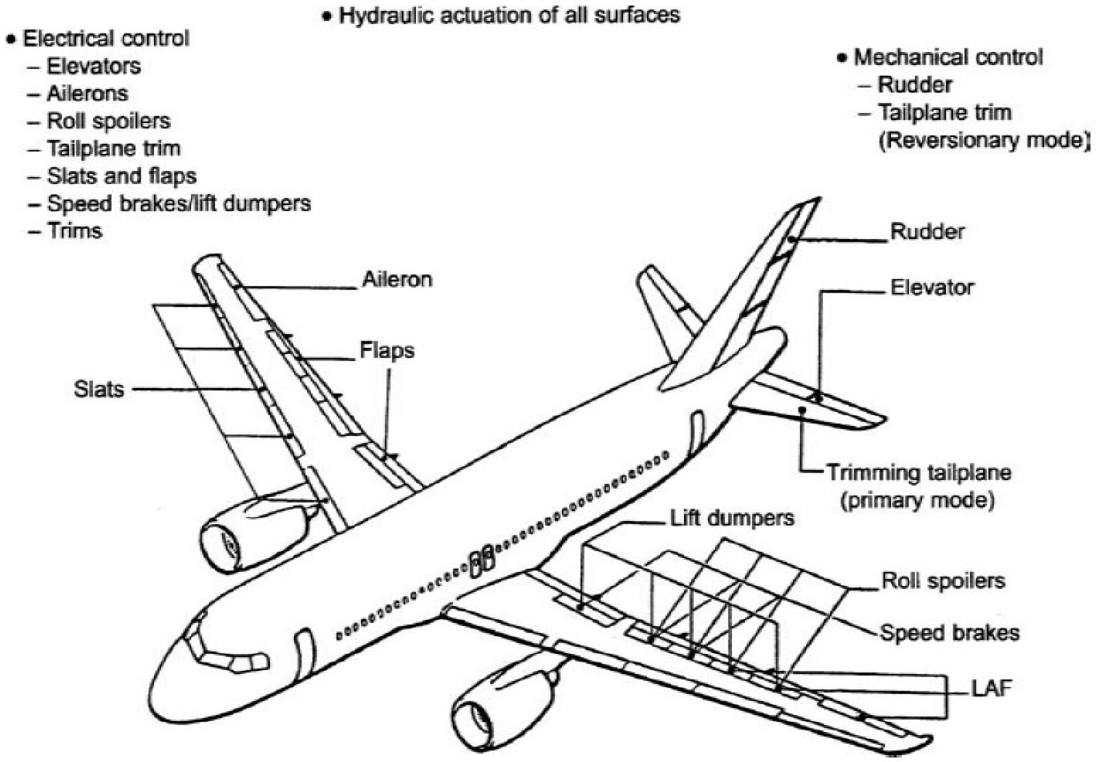


Fig. 42: Example of a commercial airliner control surfaces [48]

However, new trends are moving towards the electrification of the FCS to gain advantages in terms of masses, efficiency and maintainability. The hydraulic power used in the conventional architecture to move control surfaces is replaced by the electrical one in the More Electric architecture. To do that it is necessary to take in account new technologies, such as the EHAs. This kind of actuator uses three-phase AC power to supply the power drive electronics and consequently a variable speed motor, which in turn drives a constant displacement hydraulic pump [48]. A more efficient form of actuation is then accomplished by means of the only electrical power, involving a marginal use of the hydraulic one.

5.2. Flight Control System – Architectures

There are different possible kinds of FCS architectures, depending on the aircraft on which it operates and the manufacturer company. Moreover, the architecture itself may change in case new technologies or innovative components are introduced. The A320 FCS is characterized by different components which enable its operation. Among these there are: flight control surfaces (both primary and secondary), flight control computers (such as ELAC, SEC, FAC, and FCDC), cockpit controls (e.g. pilot and copilot sidesticks, pedals, speed brake control lever, etc.), actuators and autopilot. Furthermore, must be taken in account also other systems which are not part of FCS but contribute to its functioning (such as the electrical, the avionic and the hydraulic one).

The use of SysML language enables the capability to model and represent those architectures, highlighting their dissimilarities. Specifically, the BDD and IBD can be used to depict respectively the FCS structural hierarchy and the interactions between its components (such as the connections between them, the kind of matter, energy or signals they can exchange, etc.). Different diagrams can be developed to represent as many architectures, which can then be compared to identify their respective peculiarities[2].

5.2.1. A320 FCS – Conventional Architecture

The conventional FCS architecture implemented on A320 family aircraft is depicted in Fig. 43. Primary and secondary control surfaces displacements are actuated by means of hydraulic actuators. Three different hydraulic power sources are used to supply them: blue, green and yellow. They are represented with the first letter of their names capitalized.

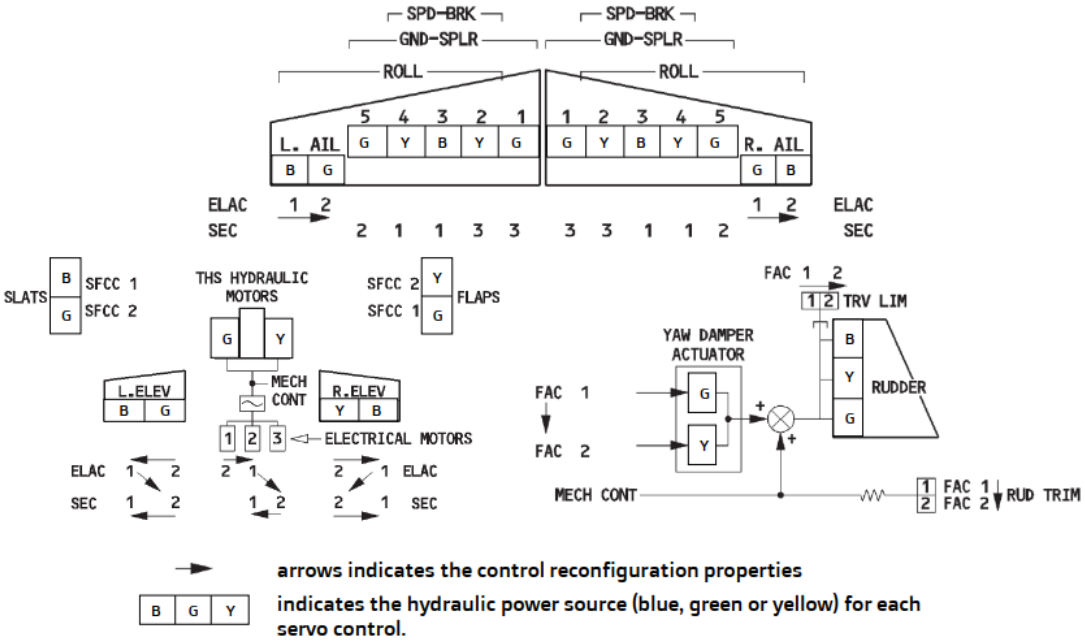


Fig. 43: A320 FCS conventional architecture [49]

[2] In this chapter, all the terms highlighted with the *Italic* font represent a characteristic element of SysML language.

Some control surfaces (especially the primary ones) can be moved using more than one actuator, each of which supplied by a different hydraulic power source. This allows to control them even in a case of a failure. As an example, each aileron can be moved by two different actuators: one is supplied by the blue hydraulic power source whereas the other by the green one.

Flight control surfaces movements are determined by FCCs. Both ELACs are used together with the three SECs to manage respectively ailerons and spoilers. Elevators, instead, are managed only by ELAC 2 during normal operations and only by ELAC 1 in case of failures. If them both result being not available, pitch control shifts to SEC 1 and SEC 2. The rudder is the only surface which can be directly controlled by pilots using pedals. However, yaw damping and turn coordination functions are automatically performed by both FACs in cooperation with ELACs.

The architecture shown in Fig. 43 can be depicted more in detail using SysML diagrams. A BDD can be used to represent the kind of relationships between the FCS, its components and the other systems with which it cooperates. As an example, the diagram shown in Fig. 44 depicts different *Associations* that relate the FCS *Block* with the others. Specifically, *Composite Associations* are used to identify its *part properties*, whereas *Reference Associations* are implemented to identify its *reference properties*. Therefore, *Blocks* used to represent components (such as flight control surfaces, flight control computers, actuators, etc.) are associated as *parts* of the FCS. Instead, *Blocks* implemented to represent external systems (such as the electrical, avionic or hydraulic systems) are associated as *references*.

The BDD can depict the structural hierarchy of the system under analysis, but it is not able to show how the internal components interact one with another. To fulfill this task, it is necessary to use an IBD. The diagram shown in Fig. 45 depicts an excerpt of FCS conventional architecture focused on roll control. *Flow ports* and *connectors* are used to define the type of matter, energy or information exchanged between the *parts* represented in the IBD. It is shown how pilot and copilot commands are sent to both ELACs and SECs through their respective sidesticks. The same can be done from the autopilot, which directly sends commands to FCCs. Both the ELACs communicate with the three SECs sending them the roll orders to apply. Furthermore, each flight control computer is powered by the electrical system and receives the necessary information to work (such as air data, aircraft inertial data, aircraft attitude data, etc.) from the avionic system. Each FCC sends electrical signals to the actuators it can command. Those signals are then receipt from servo valves, which change the hydraulic fluid flow and pressure in their respective actuators. This enables the piston rod of each actuator to move as ordered. To verify that the applied displacement is the same as the one commanded, a feedback position is sent back from the actuator to its relative FCC. Flight control surfaces can be moved thanks to the connections between their control horns and the piston rods of the actuators.

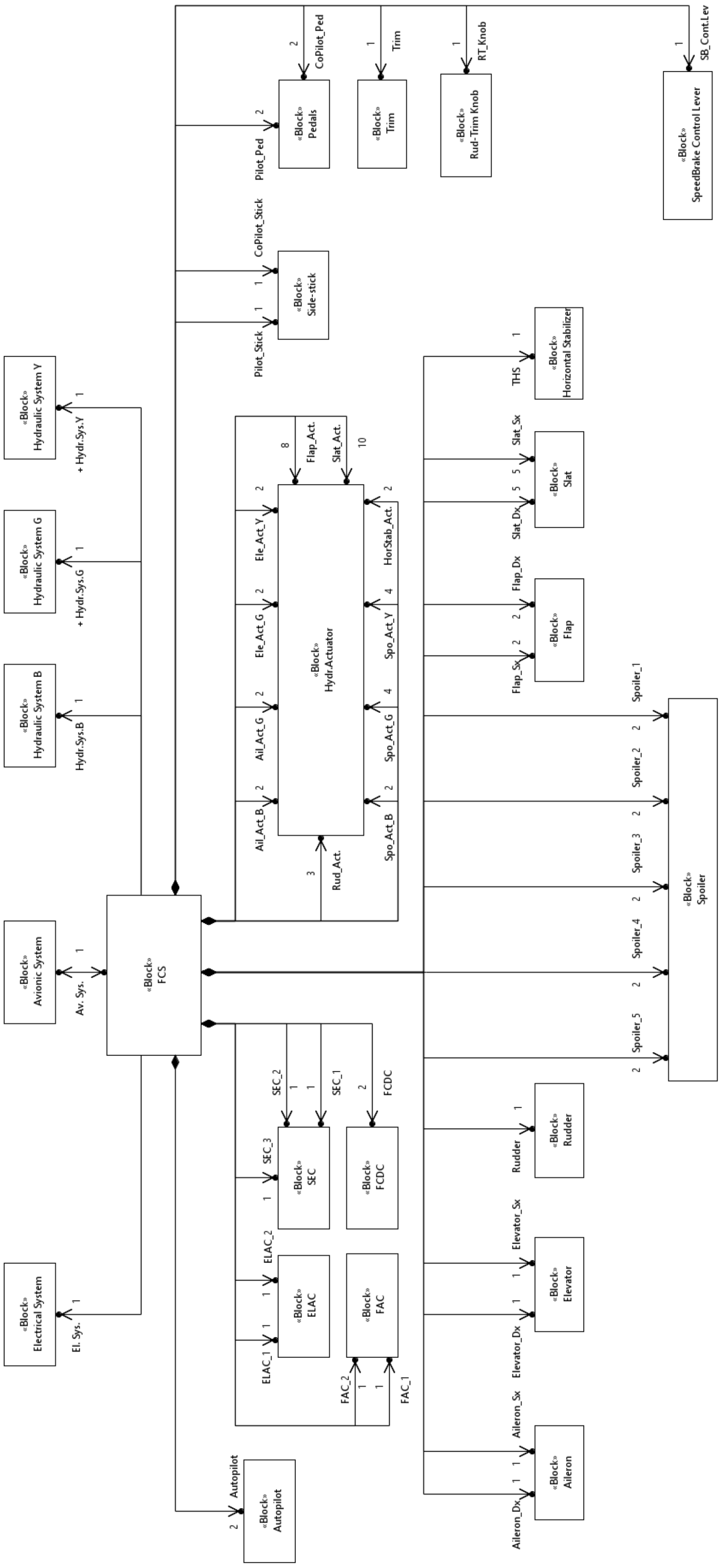


Fig. 44: Conventional A320 FCS BDD

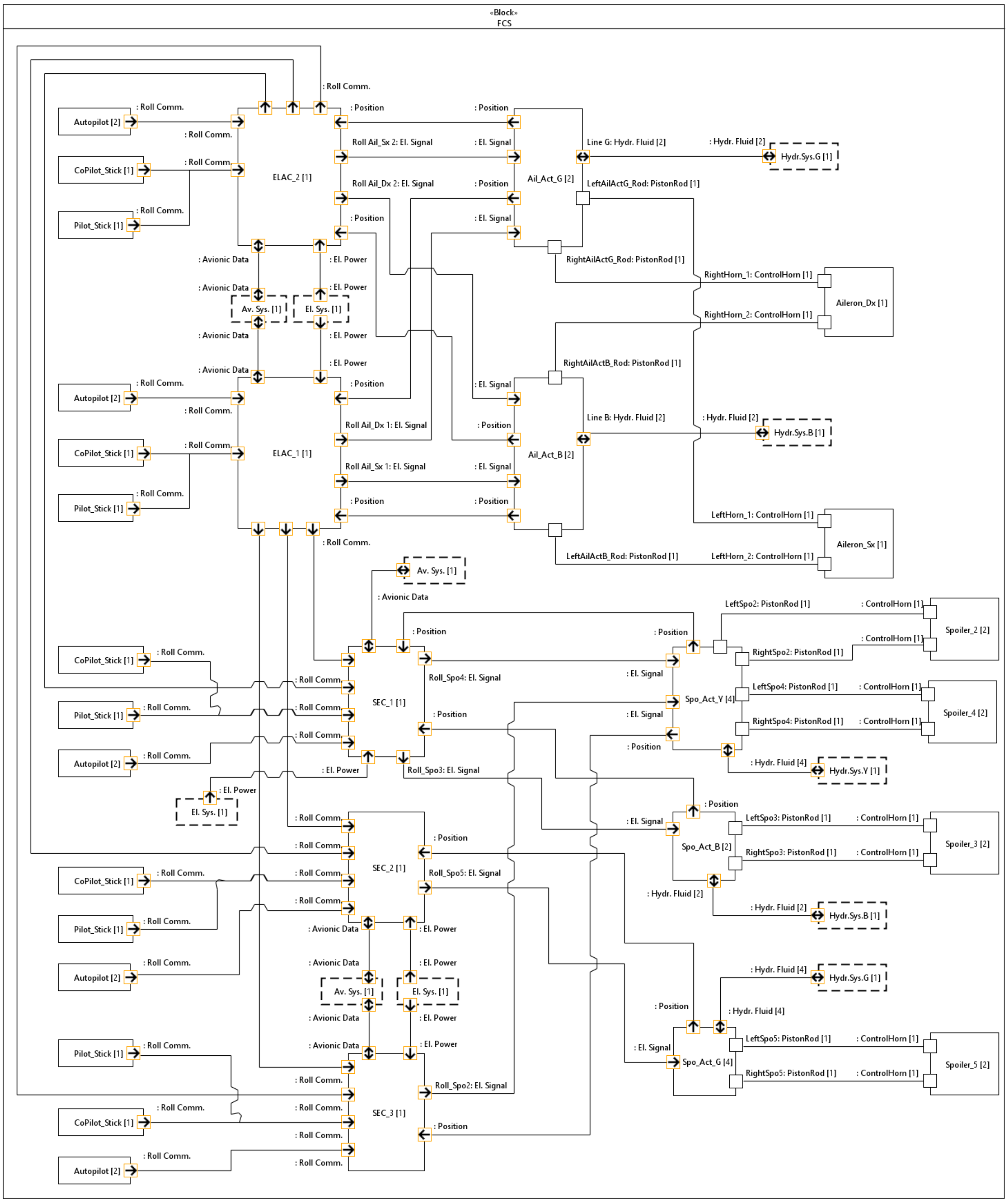
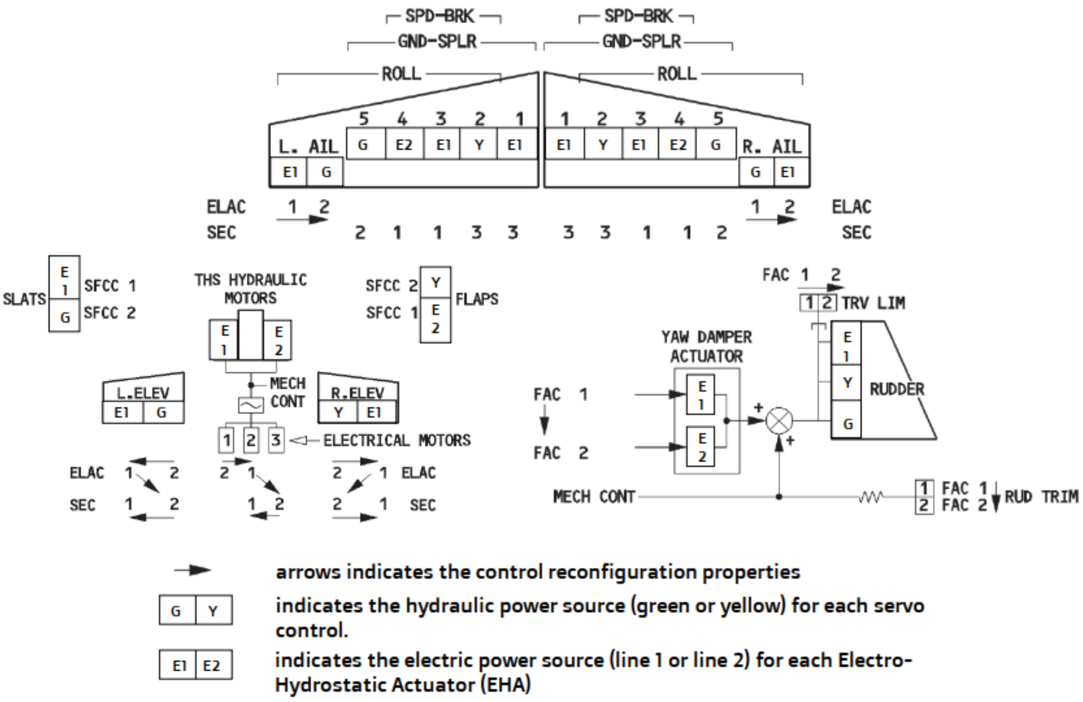


Fig. 45: Conventional A320 FCS IBD – focused on Roll Control

In Fig. 45 the *flow ports* highlight the *type* of the *items* which flow through them. They also specify the flow direction, which can be: **in**, **out** or **in-out**. As an example, the *port* named "Roll Ail_Dx 1" placed on the ELAC 1 *part property* specifies that the *item* flows out of it and "El. Signal" is its *type*. Therefore, the considered *item* will have all the same *properties* belonging to the element that describes its *type*. Instead, *full ports* are used to represent piston rods and control horns, which result being respectively *parts* of the hydraulic actuators and flight control surfaces. *Reference properties* has been used to represent external systems (such as the electrical, the avionic and the hydraulic one). In contrast to *part properties*, they are depicted by means of a box with a dashed boundary.

5.2.2. A320 FCS – More Electric Architecture

The conventional FCS architecture uses three different hydraulic power sources to supply the actuators. This involves an increase weight and a reduction of dependability, since hydraulic systems can frequently experience different kind of failures (such as oil leakages, cavitation, etc.). For this reason, there is the necessity to introduce some new technologies which allows to reduce the FCS dependency from hydraulic power sources. The more electric architecture shown in Fig. 46 makes it possible. Indeed, different flight control surfaces (especially the primary ones) are provided of at least one EHA. This innovative kind of actuator uses a variable speed electric motor to drive an internal fixed displacement hydraulic pump, which in turn moves the piston. Therefore, EHAs only need to be supplied by electrical power to work. The developed more electric architecture relies on two hydraulic power sources instead of the three used in the conventional A320 FCS. In addition, there are two electrical power sources necessary to make the EHAs work. This will result in an increase of weight, but also in an increase of system dependability.



arrows indicates the control reconfiguration properties

G Y indicates the hydraulic power source (green or yellow) for each servo control.

E1 E2 indicates the electric power source (line 1 or line 2) for each Electro-Hydrostatic Actuator (EHA)

Fig. 46: A320 FCS More Electric architecture

The BDD that represents the more electric architecture is shown in Fig. 70, Appendix C. It resembles the one depicted in Fig. 44, but there are some differences between them. First, the more electric is made of only two hydraulic systems: the green and the yellow one. The other difference is the use of two *Blocks* to represent actuators. One is used to identify the hydraulic actuators, whereas the other identifies the EHAs. The different actuators related to each flight control surface are defined using *Composite Associations*. It can be notice that the electrical power sources necessary to supply the EHAs are not shown. That is because those sources are defined as *parts* of the electrical system. Therefore, it would be better to represent them in another specific diagram, designed to depict the electrical system.

The IBD representing the More Electric architecture is shown in Fig. 71, Appendix C. Although its similarities with the conventional FCS, there are some differences concerning the implemented actuators. Both the ailerons are moved by means of at least one EHA. Each of them in turn, is powered by an electrical line, devoted to supply the FCS. The same is done for spoilers 3 and 4 (both left and right), which are moved with EHAs. In this case two different electrical channels power the actuators: the electrical line 1 – also used for ailerons – is implemented for those connected to spoilers 3, whereas the electrical line 2 is adopted for actuators that handle spoilers 4. This type of more electric architecture is named “2H/2E”, since it implements two different hydraulic power sources and two different electrical power sources. Even if the electrical line are *parts* of the electrical system, they result being *reference properties* of FCS. Therefore, in the IBD they are represented by means of a box with dashed boundaries.

5.2.3. A320 FCS – All Electric Architecture

The FCS More Electric architecture can involve different benefits in terms of weight reduction and dependability. That is due to the implementation of electrical power sources and EHAs, which increase the system reliability and reduce the dependence on hydraulic power sources. Therefore, the All Electric architecture would further increase these advantages. An example of this kind of architecture applied to the A320 FCS is depicted in Fig. 47.

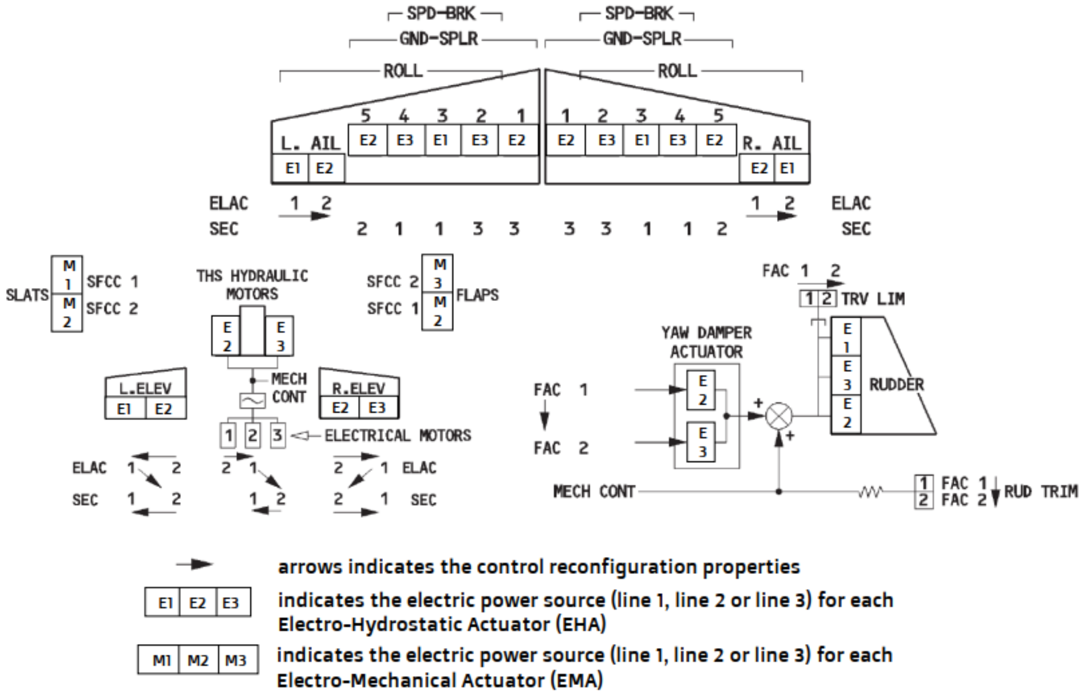


Fig. 47: A320 FCS All Electric architecture

The hydraulic power sources have been replaced with electrical power sources. Most of the implemented actuators are EHAs. Moreover, flaps and slats are moved using Electro-Mechanical Actuators (EMAs). In this way, only the electrical system is needed to move flight control surfaces.

The BDD representing the All Electric architecture of A320 FCS is shown in Fig. 72, Appendix C. It can be noticed that there are no *Blocks* concerning hydraulic systems. Moreover, most of the implemented actuators result being EHAs; whereas, the ones used to move flaps and slats are defined as EMAs. The marginal use of these kinds of actuators is related to their low reliability since they can easily get jammed. However, EMAs do not need a hydraulic pump to move their piston rods. Therefore, their weight result being reduced compared to EHAs. This is the reason that enables to use EMAs only to move secondary flight control surfaces.

The interactions between the components that characterize the All Electric architecture of FCS are represented in Fig. 73, Appendix C. The IBD is alike the others developed to depict Conventional and More Electric architectures. The only difference is in the implemented actuators. Since the diagram is focused on roll control, only EHAs are shown. Each of them is power by its relative electrical line, which in turn is represented as a *reference property*. The major difference with the other IBDs is the absence of hydraulic power sources. Indeed, this FCS architecture relies only on electrical power sources to work correctly.

In the next sections the model based RAMS analyses previously described will be applied to define safety and reliability of these three different architectures. The numerical results will be also introduced in an MDO environment to identify their effectiveness in dependability and performances.

5.3. FCS – Functional Hazard Analysis

The first analysis performed was the FHA. The aim has been to define the safety requirements at system level that must be respected. This analysis shall be performed before the design phases. However, the considered FCS has already been designed and cannot be changed. Therefore, the FHA has been performed only to explain its model-based application. In addition, it has been used to define the safety requirement that must be compared with the results gained from the other analyses. Before performing the FHA, it has been necessary to define how the considered system is intended to work. The *Activity Diagram* shown in Fig. 48 has been used to fulfill this function. It is focused on roll control and describes how the pilot and copilot inputs are transformed into a torque which can turn the aircraft around its longitudinal axis. Essentially, each input is converted into electrical signals that can be processed by FCCs. Afterward, the deflection of each flight control surface is computed taking in account both pilots commands and position feedbacks. Ailerons and spoilers are then moved following the computer orders.

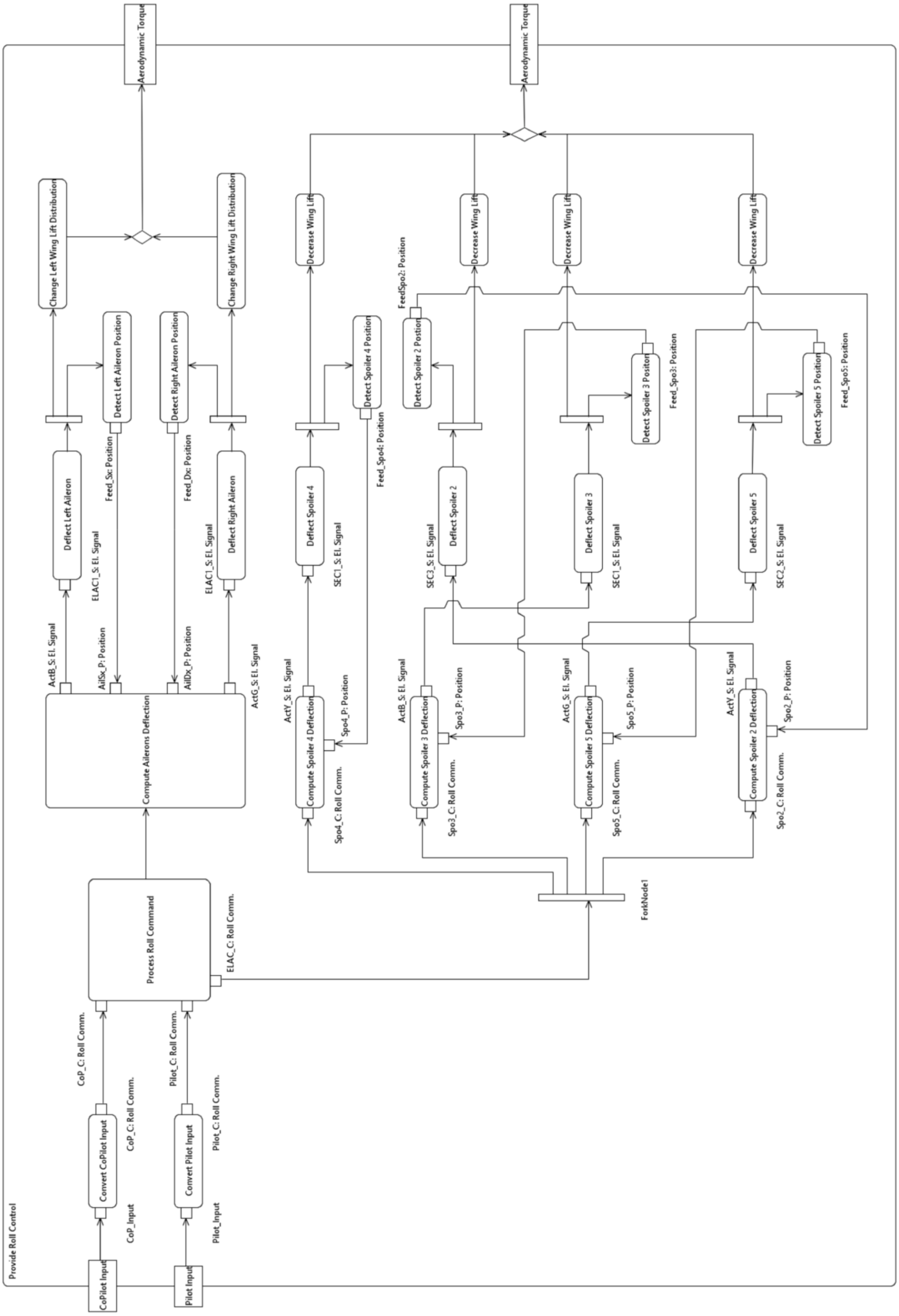


Fig. 48: FCS Activity Diagram – Provide Roll Control

The deflection of ailerons and spoilers involves two different effects on wings aerodynamic flow. The first can increase or decrease the lift distribution and can be applied anti-symmetrically on both wings to amplify its effect. The latter, instead, can only decrease the lift distribution and increase drag. Therefore, deflection of spoilers shall be applied asymmetrically to reduce lift only on one wing. Both flight surfaces movement involves a torque around aircraft longitudinal axis, even if they act in a different way on the aerodynamic flow.

Starting from the diagram shown in Fig. 48 it has been possible to define and represent other *Activities* which describe the possible functional failures that may occur. Considering the roll control, one of these failures is the inability to control ailerons. The Activity Diagram excerpt depicted in Fig. 49 shows that if it is not possible to deflect both the ailerons their control fails. If the FCS is not able to perform this function, the result is its inability to provide roll control with those control surfaces. Another possible functional failure concerns the control of spoilers. Indeed, if FCS is not able to move them, it can only provide partial roll control, as depicted in Fig. 50.

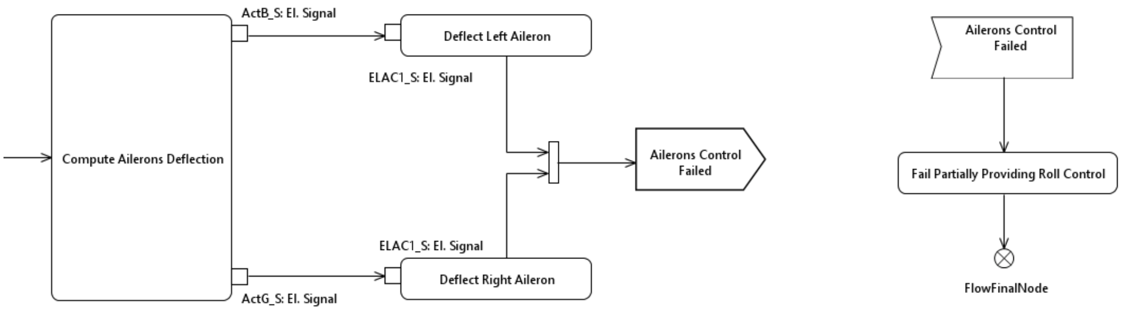


Fig. 49: Activity Diagram excerpt – Ailerons control failure

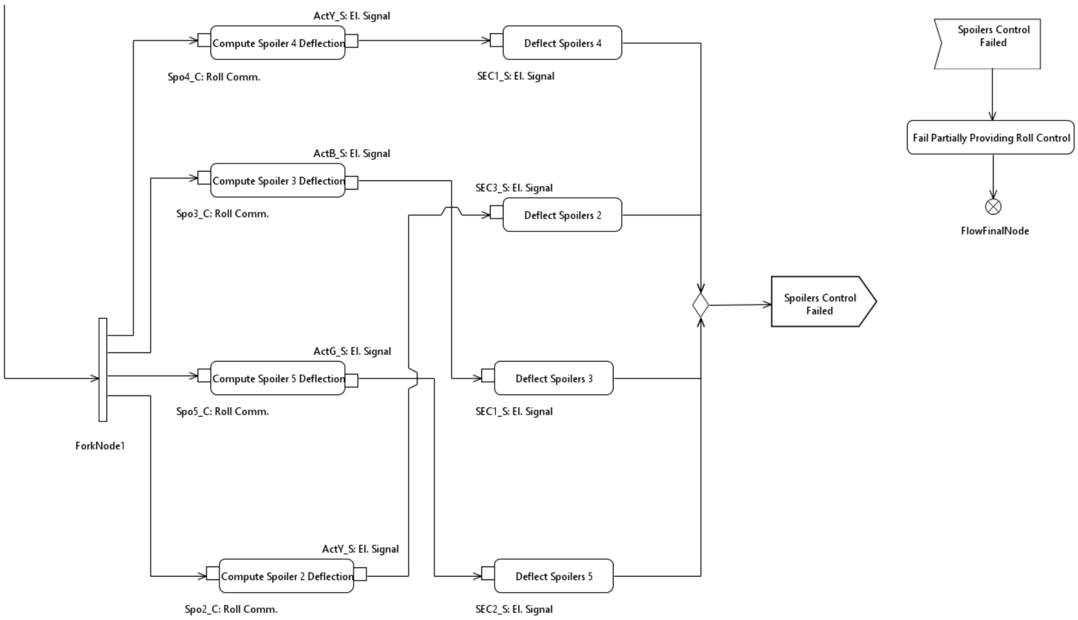


Fig. 50: Activity Diagram excerpt – Spoilers control failure

The occurrence of the two functional failures described before is represented by means of *signals*. The attributes of these elements can be used to define their respective hazard severity class and probability. *Signals* depicted in Fig. 51 show that those parameters are respectively “Critical” and “Remote” for both the functional failures. It means that they can involve severe injuries for passengers and crew; therefore, they shall be unlikely, but they can possibly occur during the considered system life. If only one of them takes place, it results being still possible to manage the aircraft around its longitudinal axis. Their simultaneous occurrences, instead, would lead to the complete inability to provide roll control. This functional hazard is classified as “Catastrophic” and can lead to death or sever injuries. Therefore, it shall be so unlikely to be never experienced during the FCS life.

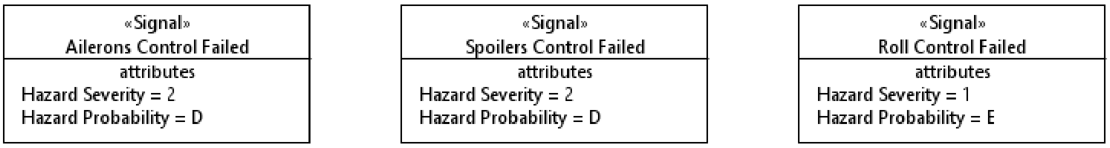
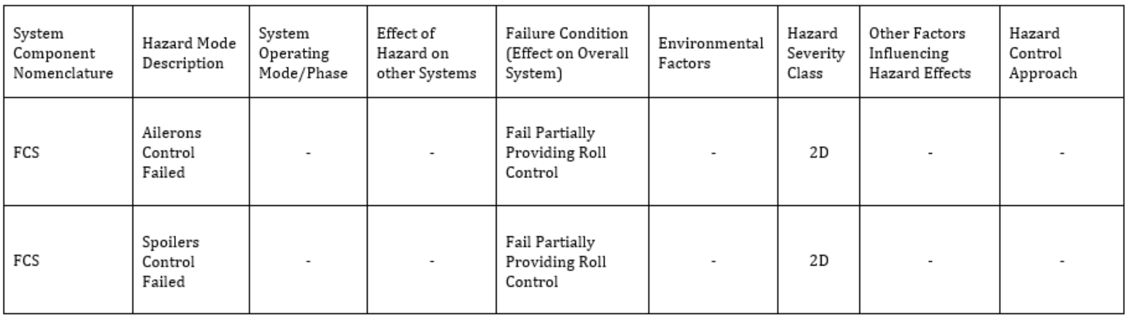


Fig. 51: Model-based hazard severity class and probability

The complete Activity Diagrams concerning the functional failures described before are shown respectively in Fig. 74 and Fig. 75, Appendix C. The information contained in these diagrams can be also used to partially fill the worksheet used during the execution of a document -based FHA.



System Component Nomenclature	Hazard Mode Description	System Operating Mode/Phase	Effect of Hazard on other Systems	Failure Condition (Effect on Overall System)	Environmental Factors	Hazard Severity Class	Other Factors Influencing Hazard Effects	Hazard Control Approach
FCS	Ailerons Control Failed	-	-	Fail Partially Providing Roll Control	-	2D	-	-
FCS	Spoilers Control Failed	-	-	Fail Partially Providing Roll Control	-	2D	-	-

Fig. 52: FCS Document Based FHA

The presented model-based FHA can be applied to all the FCS architectures described in Section 5.2. Their differences in terms of components and power sources do not change the functions they must provide. Therefore, all the three architectures can experience the presented functional failures. PSSA and SSA shall be performed to assure that they can handle these hazards and that their probability to happen is less than probability defined with FHA.

The advantage of using this model-based approach consists in the possibility to represent and analyze the behavior of the system under analysis. In this way it is easier to understand which failures can reasonably occur and the effects they can involve.

5.4. FCS – Fault Tree Analysis

The definition of safety requirements with the FHA is usually followed by execution of PSSA and SSA, which aim respectively to demonstrate and verify that the system under analysis will meet those requirements. The FTA is one of the most common technique adopted to perform these analyses. The model-based approach has been used to determine qualitatively the safety of FCS Conventional, More Electric and All Electric architectures. Specifically, the FTAs has been focused on ailerons control and will be compared to determine which one can offer more safety.

5.4.1. A320 FCS – Conventional Architecture FTA

Before starting the model-based FTA it is better to represent the system behavior to analyze, without considering fault events. The Sequence Diagram in Fig. 53 shows how different *parts* of FCS interact to provide ailerons control. In normal conditions ELAC 1 processes the command received by pilot. Then it requests the actuators powered with blue and green hydraulic power sources to deflect respectively the left and the right ailerons. Finally, it gets the feedbacks concerning their respective positions and processes them. The ***loop*** *interaction operator* specifies that this behavior is repeated until the actuators reach the commanded position. Moreover, the ***par*** indicates that the control of both ailerons occurs simultaneously. Once the system behavior to analyze has been defined it results being easier to identify the fault events that may occur and their consequences.

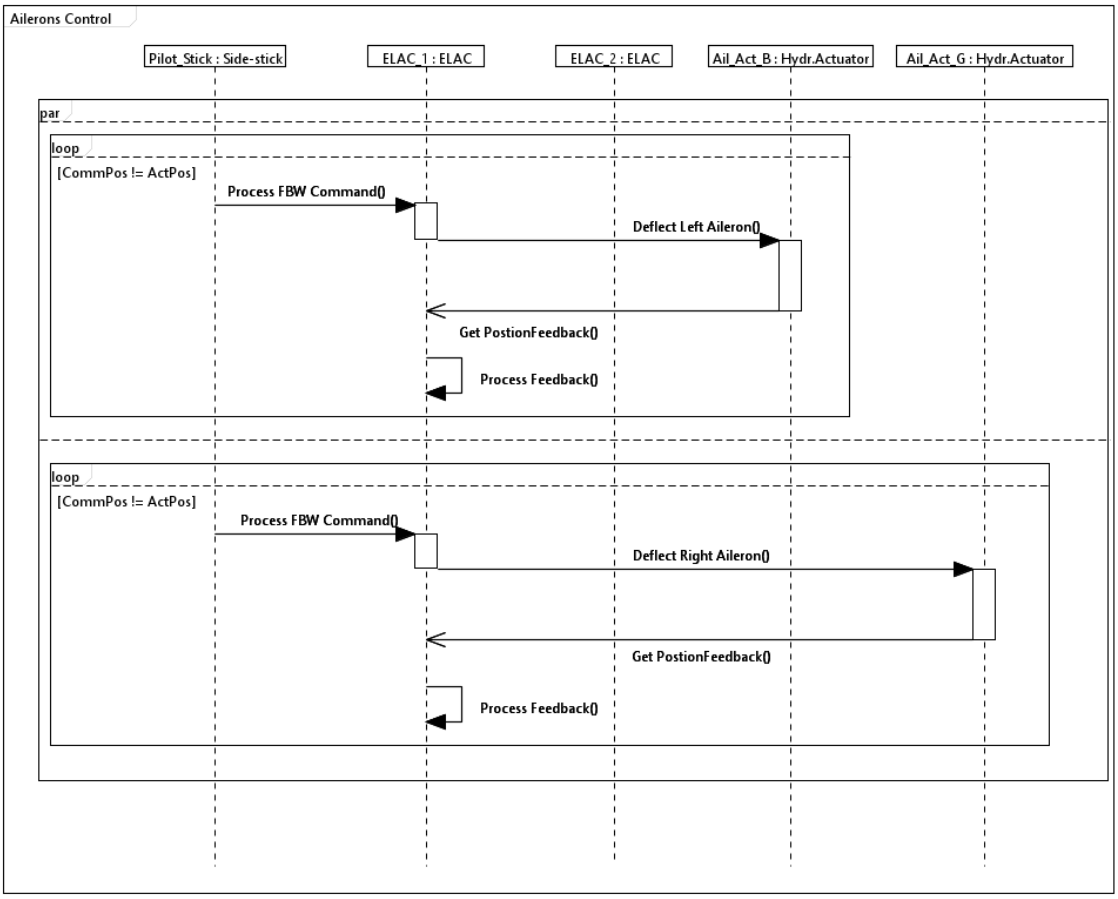


Fig. 53: Ailerons Control on Conventional FCS – Sequence Diagram

The model-based FTA performed on FCS Conventional architecture is shown in Fig. 54 and Fig. 55. The Sequence Diagram is only one, but it has been split to make it more comprehensible. The first step has been the **top event** definition, which in this case consists in the inability of controlling ailerons. Following the model-based approach described in Section 3.2, it shall coincide with the name of the *Interaction* used to represent the FTA. That name is represented on the top left of the diagram in Fig. 54. The next step has been the identification of fault events that bring to the **top event** and the **logic gates** that relates them.

The FTA starts considering an ELAC 1 failure consequently to the request of processing the pilot command. The *reply message* containing *synchronous message* name and the reply "Failed" identifies the failure occurrence. Whereas, the *guard* that characterize the first *operand* of the ***alt*** represents the fault event name. Therefore, the ELAC 1 inability of processing pilot orders involves the sending of the same request to the ELAC 2, which should perform its same functions. If also this computer fails processing commands, the result is that is not possible to control ailerons.

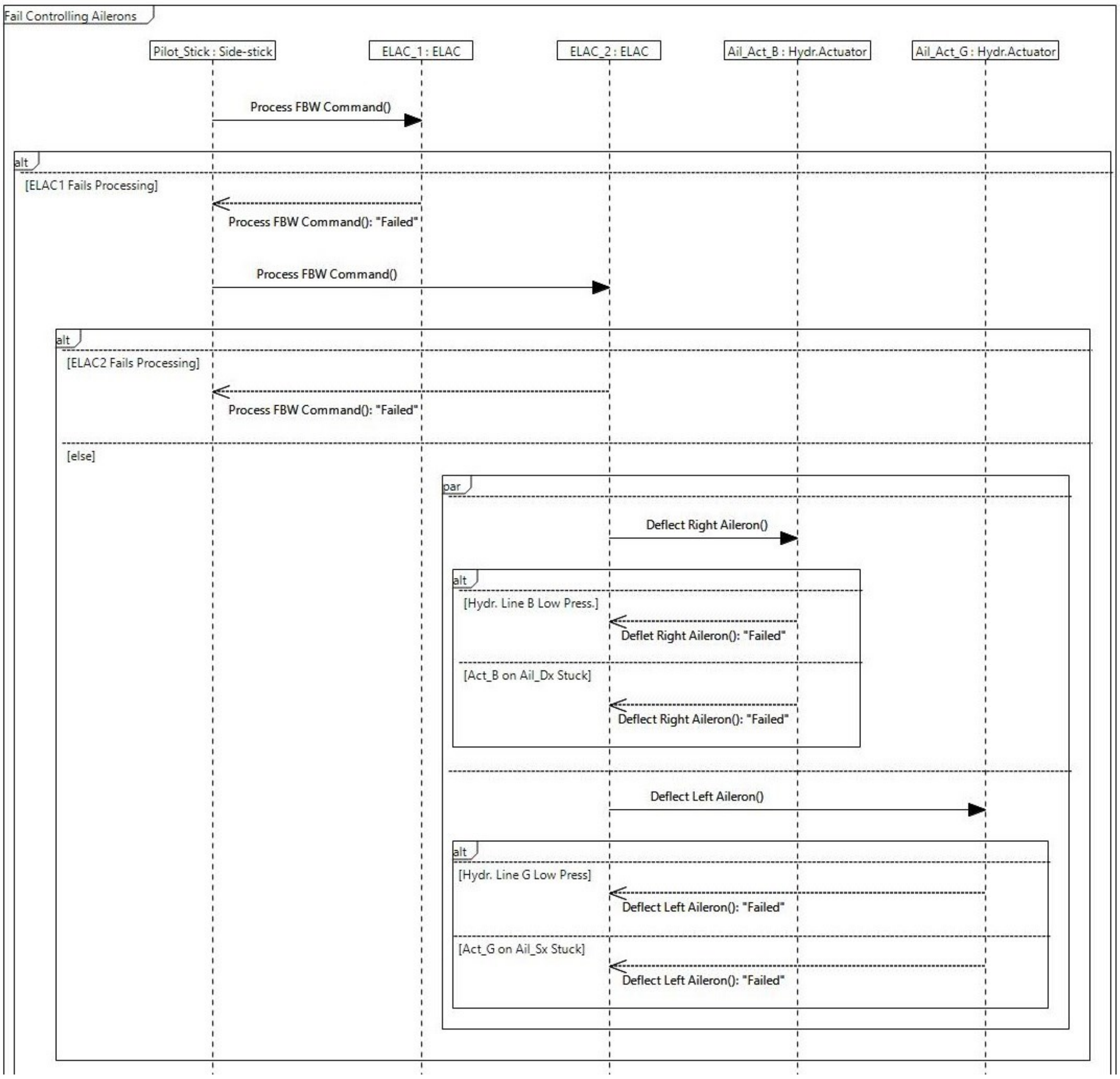

Fail Controlling Ailerons
Pilot_Stick : Side-stick
ELAC_1 : ELAC
ELAC_2 : ELAC
Ail_Act_B : Hydr.Actuator
Ail_Act_G : Hydr.Actuator
Process FBW Command()
alt
[ELAC1 Fails Processing]
Process FBW Command(): "Failed"
Process FBW Command()
alt
[ELAC2 Fails Processing]
Process FBW Command(): "Failed"
[else]
par
Deflect Right Aileron()
alt
[Hydr. Line B Low Press.]
Deflet Right Aileron(): "Failed"
[Act_B on Ail_Dx Stuck]
Deflect Right Aileron(): "Failed"
Deflect Left Aileron()
alt
[Hydr. Line G Low Press]
Deflect Left Aileron(): "Failed"
[Act_G on Ail_Sx Stuck]
Deflect Left Aileron(): "Failed"

Fig. 54: Failed ailerons control on Conventional FCS – Sequence Diagram Part 1

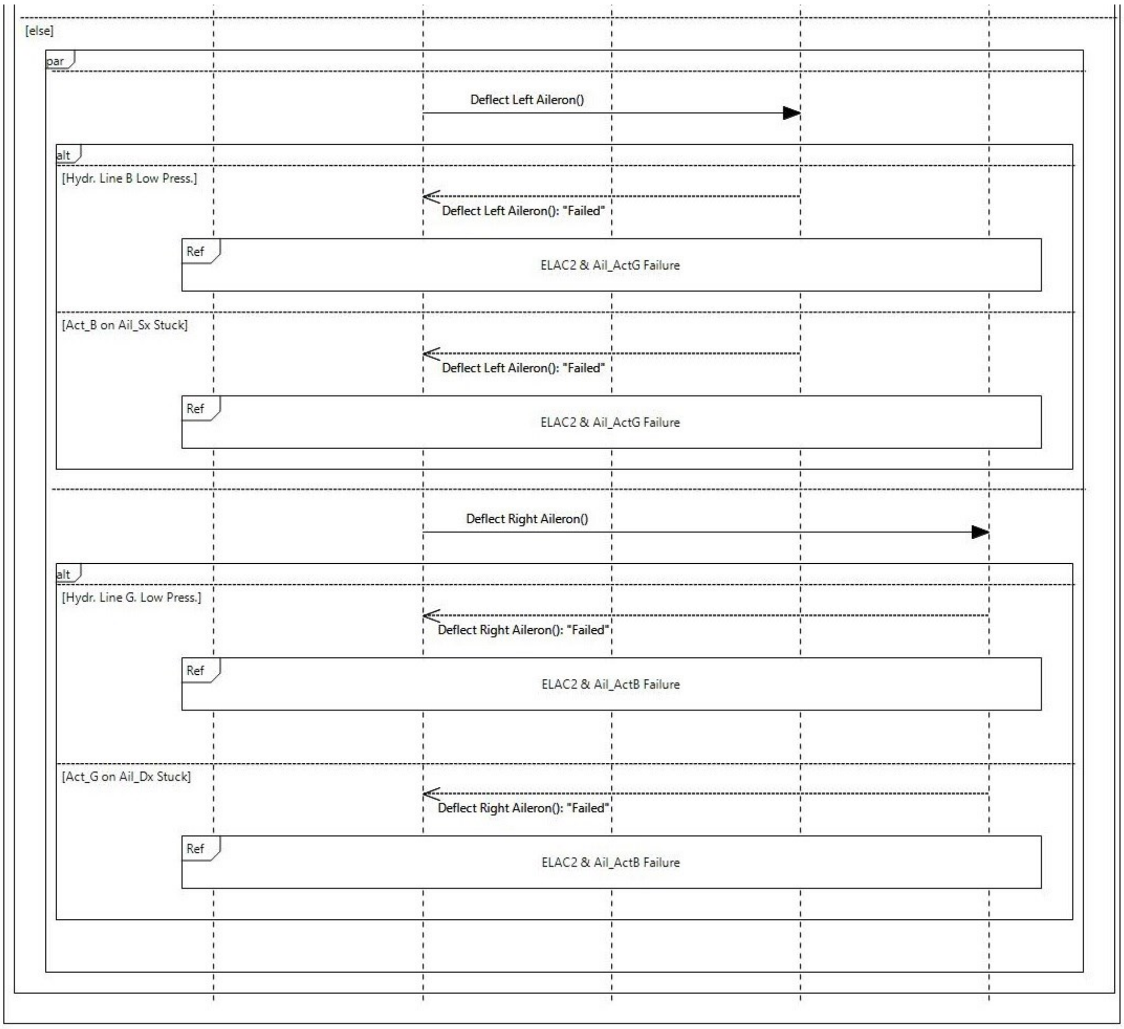


Fig. 55: Failed ailerons control on Conventional FCS – Sequence Diagram Part 2

Both ELAC 1 and ELAC 2 faults lead to the top event. This is highlighted from the absence of other *messages* in the *operand* which contains the reply coming from ELAC 2. Moreover, the use of nested ***alt*** indicates that both the events must occur to reach the top one. On a fault tree this can be represented with an AND gate.

The other fault events defined in the FTA concern the loss of hydraulic power sources pressure and the jamming of actuators. These both involve the inability to move the ailerons as requested by ELACs. Part of the FTA (especially the one shown in Fig. 55) have been represented using the *interaction use*, which refers to an existing *interaction* depicted in another Sequence Diagram. This solution has been implemented to reduce the diagram dimensions and decrease its complexity. Nevertheless, each *interaction* at which an *interaction use* refers follows the rules necessary to represent a model-based FTA. The ones shown in Fig. 55 are depicted in Fig. 76 and Fig. 77, Appendix C.

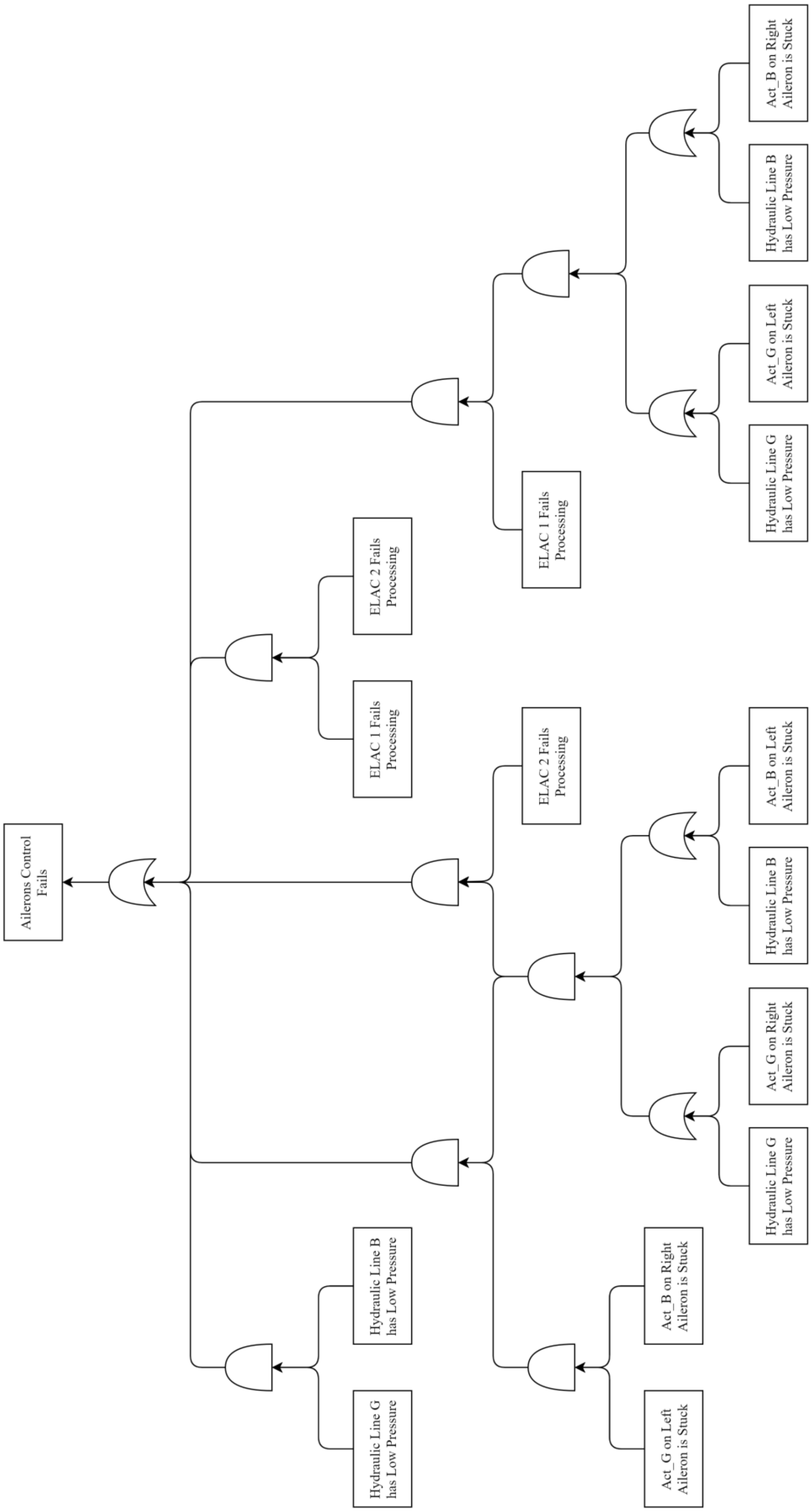

Ailerons Control Fails
Hydraulic Line G has Low Pressure
Hydraulic Line B has Low Pressure
Act_G on Left Aileron is Stuck
Act_B on Right Aileron is Stuck
Hydraulic Line G has Low Pressure
Act_G on Right Aileron is Stuck
Hydraulic Line B has Low Pressure
Act_B on Left Aileron is Stuck
ELAC 2 Fails Processing
ELAC 1 Fails Processing
ELAC 2 Fails Processing
ELAC 1 Fails Processing
Hydraulic Line G has Low Pressure
Act_G on Left Aileron is Stuck
Hydraulic Line B has Low Pressure
Act_B on Right Aileron is Stuck

Fig. 56: FCS Conventional architecture – document-based FTA

The combination of faults described before makes possible different paths which can lead to the top event. The fault tree in Fig. 56 represents more clearly the interaction between these faults. Therefore, the document-based approach gives a simpler and clearer way to depict the FTA. However, in contrast to the model-based approach, it does not allow to define fault events starting from the system nominal behavior. This involves a greater difficulty in performing the analysis.

5.4.2. A320 FCS – More Electric Architecture FTA

Performing the FTA on the FCS More Electric architecture has given results alike the one shown in section 5.4.1, but with some peculiar differences. Indeed, the use of EHA and electrical lines instead of hydraulic actuators and hydraulic power sources changed some fault events. The Sequence Diagram depicted in Fig. 78, Appendix C, shows that in case of an internal failure, EHAs follow the ailerons movements instead of controlling them. The same happens for the electrical line 1, which can fail providing power to the variable speed motor of each EHA. These faults combined with the hydraulic actuators jamming and low pressure in hydraulic systems lead to the inability to control ailerons. Compared to the Conventional architecture, the More Electric one has less probability to reach the top event. The reason is the implementation of components with different technologies, which can fail in different ways. This makes less probable the combination of faults that lead to top event.

5.4.3. A320 FCS – All Electric Architecture FTA

The execution of FTA on the FCS All Electric architecture has given results further different from the ones gained before. The implementation of only EHAs and electrical lines involves the realization of the top event with few different type of faults. This is in contrast with the results gained from the More Electric architecture and is more alike what has been observed in the Conventional one. Therefore, the All Electric FCS appears less safe than More Electric in controlling ailerons. However, if faults concerning EHAs and electrical lines result having less probability to occur than the ones regarding hydraulic actuators and hydraulic power sources, the All Electric would appear the safest among the three architectures.

5.5. FCS – Failure Mode and Effects Analysis

The FMEA is another technique used in SSA, usually performed after designing the aircraft and its systems. It allows to describe more in detail failure occurrences and their way of acting. It also specifies their causes and their effects on the system. FMEA can be performed to analyze the possible failures that may occur to system components and determine their effect on safety. This can be also connected to the FTA with the aim of investigating which faults have a greater impact and which one result being more probable to occur.

The model-based FMEA has been performed taking in account three different components: a hydraulic actuator, an EHA and an EMA. Each of them characterizes at least one of the three FCS architectures under analysis. Therefore, the obtained results will provide a detailed description of their failures impact on safety.

5.5.1. A320 FCS – Conventional Architecture FMEA

The FMEA performed on FCS Conventional architecture has been focused on one of its hydraulic actuators. Before starting with the analysis, it has been developed a State Machine Diagram capable of describing the considered component behavior. Indeed, in Fig. 57 is represented a *State Machine* composed by different *regions*. Each of them contains *states* in which the considered component can be. In this case, the actuator is simultaneously in three different *states*: it is waiting for an electrical demand, monitoring the piston displacement and keeping the piston in a fixed position. After acquiring the electrical demand, it starts regulating the hydraulic fluid flow rate. At the same time, it starts moving the piston because of the increasing pressure difference acting on it. When this difference becomes again null the actuator returns waiting for another electrical demand and keeping the piston position fixed. Moreover, while passing from one *state* to another it continues monitoring the piston displacement. To describe more in detail the component functioning, have been defined *Do behavior* and *Exit Behavior* of some *states*. As an example, it is specified that while monitoring the piston displacement, the actuator detects the its position. Instead, when exiting form that *state* it sends position data to the Actuator Control Electronics (ACE), which is integrated in FCCs in the A320 FCS architectures.

Starting from the diagram in Fig. 57, another *State Machine* has been defined representing the component behavior in case of a failure. Specifically, in Fig. 58 is shown that the actuator becomes stuck in case of its servo-valve seizure. When in this *state*, it is specified that the considered component remains in a fixed position regardless the demand. The effect is the inability to move the control surface related to the actuator, since it results being blocked. The obtained results allow defining the failure influence on component behavior and the severity of its effects on the FCS. Moreover, the identification of its cause allows to further investigate methods to reduce or avoid the failure occurrence. In this case, the actuator jamming makes it unable to perform its functions and the effects on FCS can become severe or catastrophic. However, the servo-valve seizure results being unlikely, making the failure occurrence likewise improbable.

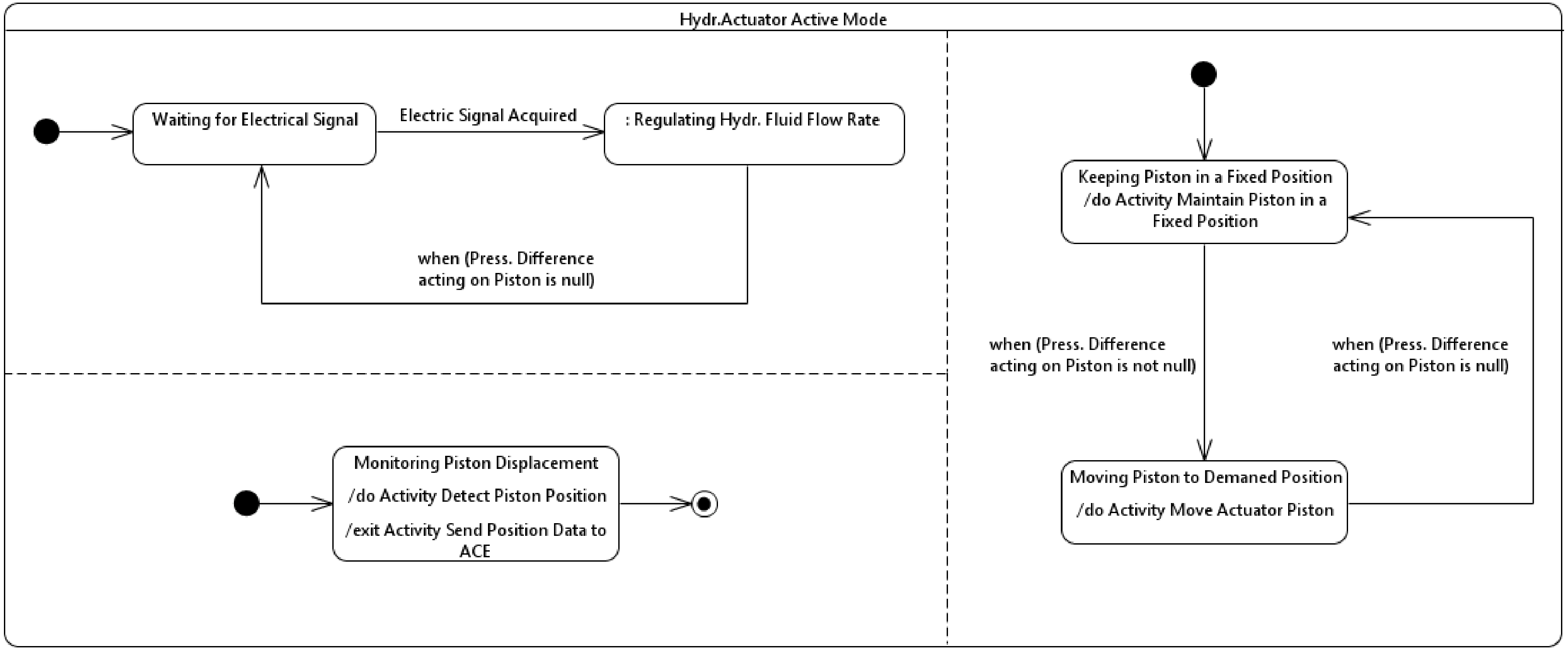

Hydr.Actuator Active Mode
Waiting for Electrical Signal
Electric Signal Acquired
: Regulating Hydr. Fluid Flow Rate
when (Press. Difference acting on Piston is null)
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE
Keeping Piston in a Fixed Position /do Activity Maintain Piston in a Fixed Position
when (Press. Difference acting on Piston is not null)
when (Press. Difference acting on Piston is null)
Moving Piston to Demaned Position
/do Activity Move Actuator Piston

Fig. 57: Hydraulic Actuator active mode – State Machine Diagram

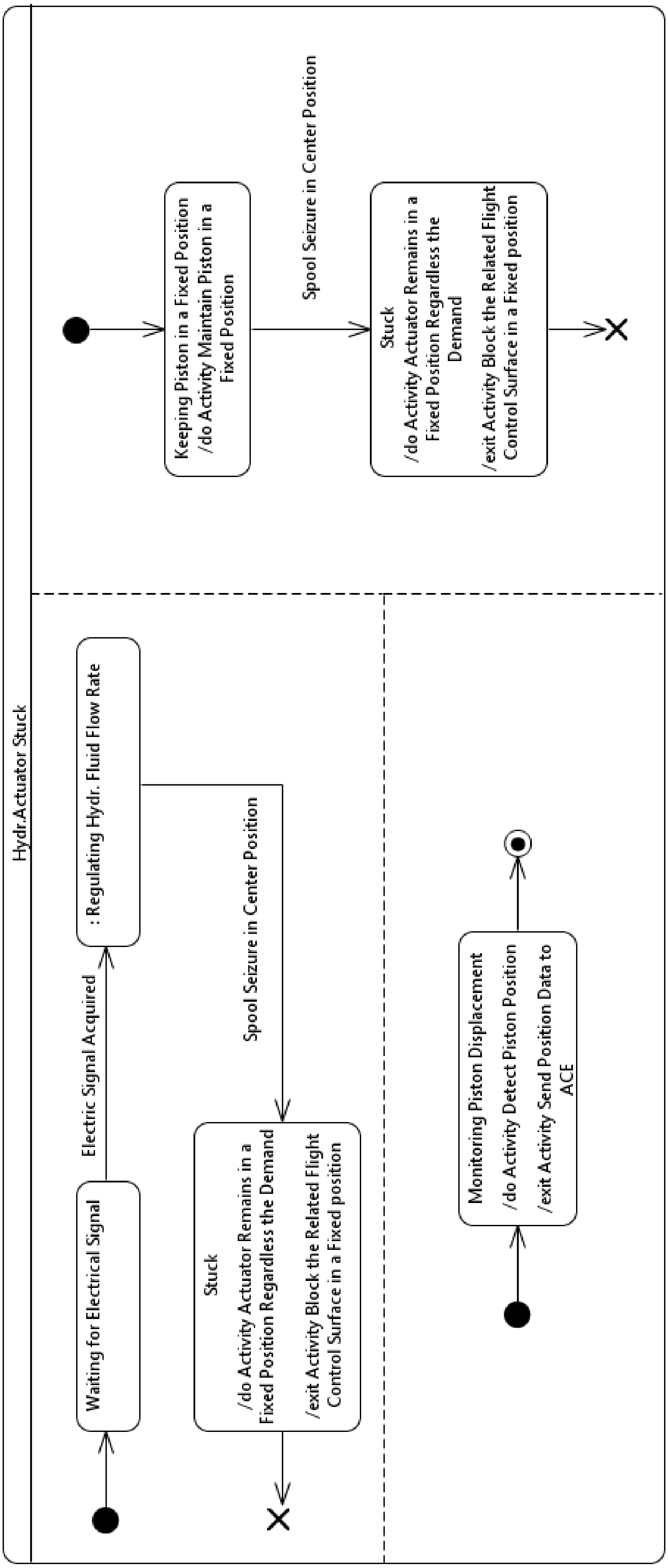

Hydr.Actuator Stuck
Waiting for Electrical Signal
Electric Signal Acquired
: Regulating Hydr. Fluid Flow Rate
Spool Seizure in Center Position
Stuck
/do Activity Actuator Remains in a Fixed Position Regardless the Demand
/exit Activity Block the Related Flight Control Surface in a Fixed position
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE
Keeping Piston in a Fixed Position
/do Activity Maintain Piston in a Fixed Position
Spool Seizure in Center Position
Stuck
/do Activity Actuator Remains in a Fixed Position Regardless the Demand
/exit Activity Block the Related Flight Control Surface in a Fixed position

Fig. 58: Stuck Hydraulic actuator – State Machine Diagram

The model-based approaches used to perform FMEA and FTA also enables the connection of their results. Indeed, some fault events in Fig. 54 and Fig. 55 concern the ailerons actuators jamming. Therefore, the diagram in Fig. 58 can be used to obtain more information about that failure, giving the possibility to analyze more in depth the system safety. This does not mean that the FTA can be derived from FMEA and conversely. It only means that the execution of both the analyses following a model-based approach contributes to a better and well-structured safety analysis.

5.5.2. A320 FCS – More Electric Architecture FMEA

The FCS More Electric architecture is characterized by both hydraulic actuators and EHAs. Since the first has been already considered for the FMEA performed on the Conventional architecture, the latter will be taken in account. The EHA behaves in a similar way to the hydraulic actuator. The principal difference – as depicted in Fig. 80, Appendix C – consists in the use of an internal hydraulic pump instead of external hydraulic power sources and servo-valves to regulate the fluid flow. Therefore, after acquiring the FCC demand, the EHA uses electrical power to drive a variable speed motor, which in turn moves the fixed displacement pump. The hydraulic fluid regulation involves the generation of a pressure difference that acts on piston and makes it move as ordered.

The difference between hydraulic actuator and EHA makes possible to take in account and analyze another kind of failure. Indeed, the model-based FMEA depicted in Fig. 81 shows that the EHA follows its relative flight surface movements instead of moving its piston to the demanded position. This is caused by pump performance degradation, which does not allow to apply the right pressure difference to the piston and leaves it at mercy of external forces. The effect of this failure is the inability to control the surface and move it as ordered by pilot.

Even in this case, the adopted model-based approach allows to connect the results obtained from FMEA and FTA. Moreover, it has been possible to integrate the analysis performed on the hydraulic actuator in the More Electric architecture model. In this way, that model has been enhanced using information already defined in the Conventional FCS model.

5.5.3. A320 FCS – All Electric Architecture FMEA

The FMEA performed on All Electric FCS architecture is focused on the EMA, since the EHA has already been considered during the More Electric architecture analysis. The EMA behavior resemble the ones of hydraulic actuator and EHA, as shown in Fig. 82. The significant difference is the use of mechanical power to move the piston rod, instead of the hydraulic one. This involves the implementation of an electric motor, which is driven after the EMA reception of FCC electrical signal. The motor in turn applies rotary motion to the screw jack, which moves the piston rod until it reaches the demanded position.

The failure analyzed with the model-based FMEA consists in the EMA jamming, caused by the deformation of a screw jack ball. As depicted by the State Machine diagram in Fig. 83, this failure does not allow to move the screwjack anymore. Therefore, the effect results being the inability to move the flight control surface related to the EMA, which remains stuck in a fixed position.

5.6. FCS – Reliability Block Diagram

Once FTA and FMEA has been performed to assess safety of the considered FCS architectures, the model-based approach has been used to outline their respective RBDs. In this way, it has been possible to evaluate and compare the reliability of the three architectures. Moreover, the obtained results have been taken in account for being integrated into an MDO environment. Their impact on the aircraft parameters (MTOW, operating costs, maintenance costs) will be discussed in the next chapter.

5.6.1. A320 FCS – Conventional Architecture RBD

The conventional flight control system RBD has been defined starting from the IBD that represents its architecture. It has been already depicted in Fig. 45 and specifies the interaction between the different components that compose the FCS, focusing on roll control. However, that diagram is characterized by a large amount of information which can make difficult outline the RBD. Therefore, following the steps described in chapter 3, a simpler IBD has been defined, which is depicted in Fig. 59. It only contains *parts* and *connections* and focuses on representing the links between the different *instances*. This made easier to understand how to relate the considered components in the RBD. Moreover, to reduce the diagram complexity, both avionics and autopilot have not been taken in account.

Both pilot and autopilot result being connected to all the FCCs implemented for roll control[3]. Since they carry out the same function, they must be considered in parallel. Moreover, each FCC is powered by the electric system. Therefore, it must be in series with them all. The IBD shows no connections between FCCs, even if ELACs exchange data with SECs. For the purposes of RBD outline these connections are not relevant since a failure concerning them does not involve FCCs malfunctions. Therefore, the relationship between ELACs and SECs in the RBD is a parallel. Afterwards, also the connections between FCCs and their related actuators have been analyzed. The IBD highlights the number of actuators that each computer controls using *multiplicities* on *connectors*. This allowed to specify the number of links present between a

[3] It is important to highlight that even if they have been depicted multiple times in the diagram, there are only one pilot and one copilot sidestick; this way of representing has been adopted to make the diagram easier to understand.

computer *instance* and the actuator *instances*. As an example, there is only one connector that joins SEC 3 to the “Spo_Act_Y” *part property*. But the *multiplicities* shown on its ends specify that the *instance* of SEC 3 is linked to two of the four *instances* of the spoiler actuators. When *multiplicities* are not shown on *connectors* ends, their value is one. The relationship between each FCC and the actuators it controls must be a series in the RBD since the roll control cannot be provided if one of them fails. Then, each actuator has been also related to its relative flight control surface and to its hydraulic power source. Obviously, the relationship with each hydraulic power source in the RBD must be a series, since they provide the power needed form actuator to work. After analyzing the IBD, it has been possible to outline an RBD capable of defining the FCS reliability in providing roll control. The resulting diagram is shown in Fig. 60.

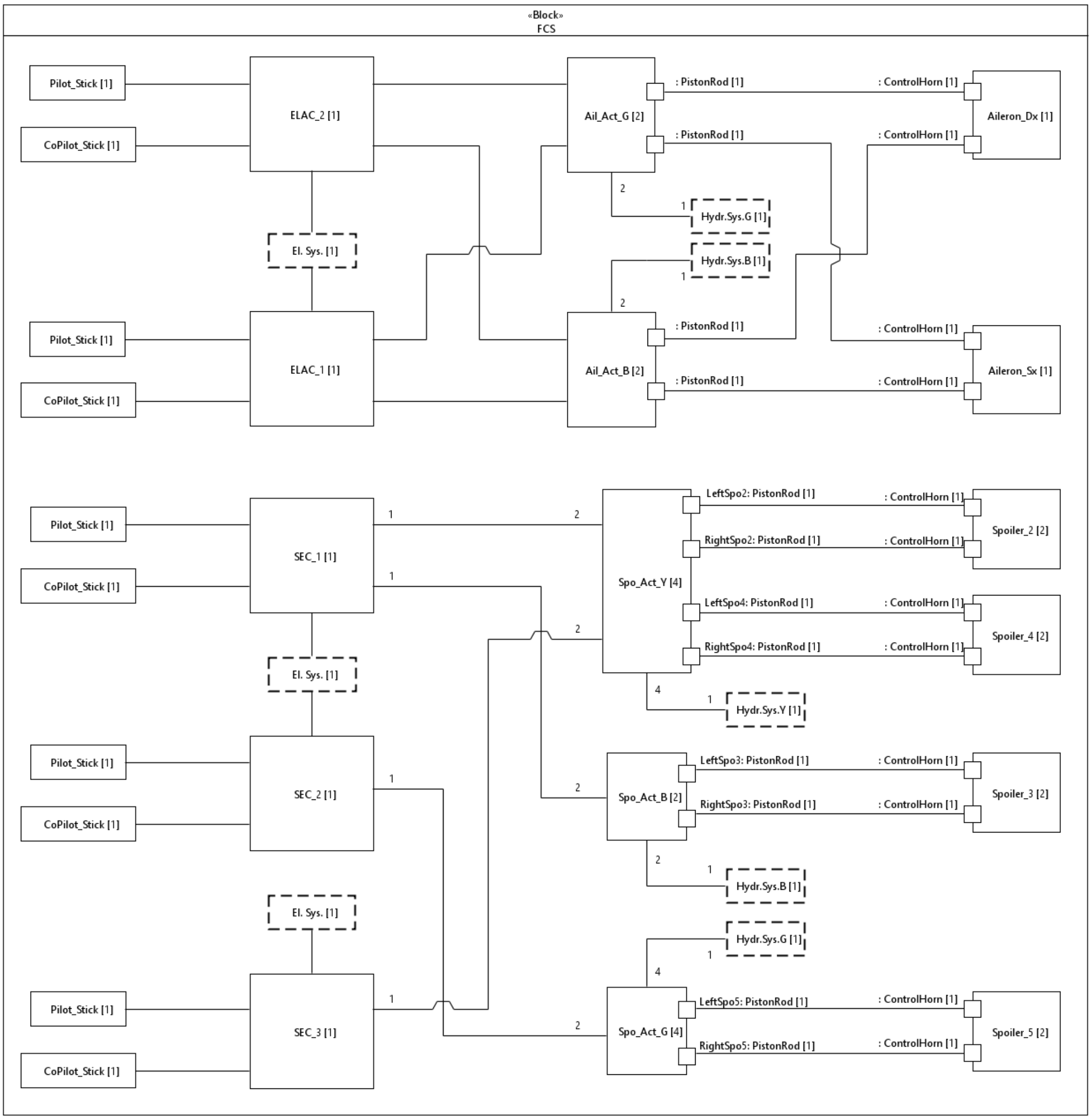


Fig. 59: FCS Conventional architecture – IBD Simplified focused on Roll Control

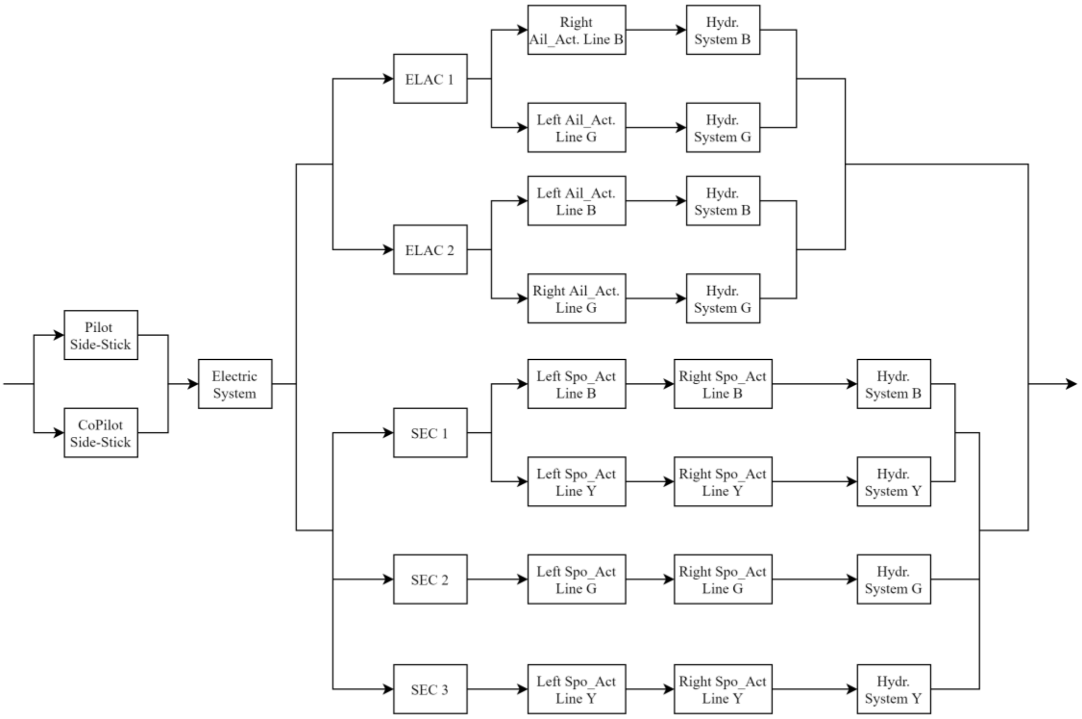


Fig. 60: FCS Conventional Architecture RBD – Focused on Roll Control

Even if flight control surfaces have been represented in the IBD, the have not been taken in account in the RBD since they do not contribute to the reliability definition. The reason of this is attributable to the FCS, whose aim is the control of those surfaces. Therefore, its reliability can be defined only taking in account the components that make it achieve that purpose. Another difference in the RBD is in the spoiler actuators. The relationship between the ones placed symmetrically on the two semi-wings is a series. This is in contrast with the relationship defined for aileron actuators. The reason is that FCCs automatically retract a spoiler if they detect a fault on another spoiler placed symmetrically on the other semi-wing. Therefore, the failure of the right spoiler actuator involves the inability to use the left spoiler and conversely.

The hydraulic power sources are represented with multiple blocks in the RBD, even if they should have been only three. However, this would have made the diagram configuration complex and therefore unsolvable by the tool described in chapter 4. So, it has been decided to adopt the method described in section 3.4 to make the RBD solvable. Obviously, it has been taken in account that the FCS resulting reliability will not take in account common cause failures.

5.6.2. A320 FCS – More Electric Architecture RBD

Even the RBD concerning the FCS More Electric architecture has been defined starting from a simpler version of its IBD. As depicted in Fig. 84 the diagram is alike the one representing the FCS Conventional architecture. The only difference is the implementation of EHAs instead of some hydraulic actuators and the use of electrical lines to power them. This similarity has affected also the RBD, which results having the same configuration but with some different blocks.

Therefore, it is expected that reliability value obtained from the RBD analysis will be near to the one obtained for the Conventional FCS architecture. That is because the only parameters that can affect that value are the failure rates of EHAs and electric lines, which should be different form the ones of hydraulic actuators and hydraulic systems.

5.6.3. A320 FCS – All Electric Architecture RBD

The All Electric FCS architecture is characterized by a further electrification respect with the More Electric one. This results in the only use of EHAs and electrical lines to control flight control surfaces, which has been fully depicted with the IBD in Fig. 85. Therefore, the RBD representing the All Electric FCS has the same configuration of the ones used to represent both the Conventional and More Electric architectures. The only difference is the complete absence of hydraulic actuators and hydraulic systems. In this case, the implementation of only EHAs and electrical power sources can affect the reliability result more than in the More Electric architecture.

The calculation of the three architectures reliabilities and their comparison will be further discussed in the next chapter. It will be also highlighted the impact of these values on some aircraft parameters after integrating them into an MDO environment.

6. MDO Integration Results

The safety provided by the three FCS architecture under analysis has been qualitatively defined by means of model-based FTA and FMEA and has been discussed in the previous chapter. An RBD for each one of them has been also defined. It can be analyzed with the tool described in chapter 4 to evaluate quantitatively their reliability. These results have been then integrated into an MDO environment, to estimate the impact of each architecture on the overall aircraft parameters.

6.1. Reliability Results

The RBDs representing the Conventional, More Electric and All Electric architectures (focused on roll control) have been defined starting from their IBDs, depicted respectively in Fig. 59, Fig. 84 and Fig. 85 (the last two are shown in Appendix C). Before proceeding, it has been necessary to define the reliability of each one of their components. Usually these values are not provided by aircraft companies. However, their failure rates have been found in some books [50] and research works [51] in which FCS safety and reliability were analyzed.

From these failure rates it has been possible to calculate the Weibull scale parameter necessary for the tool to define the reliability of each component. The shape parameter, instead, has been set to unitary value for components which are usually subject to random failures (such as flight control computers). Whereas, for components that tend wearing out it has been set equal to one in most of their first hours of life and greater than one in the last hours, when failures caused by wear out are more likely to occur. Both failure rates and Weibull parameters are respectively shown in Table 2 and Table 3.

Once defined the Weibull parameters and the lifetime of each component, it has been necessary to develop the architectures RBDs in form of the index matrix described in section 4.1.2. This process may induce to a great amount workload – especially when defining matrices with large dimensions – and may involve different errors. Therefore, the tool contains a function which can automatically create the index matrix. To make it work it is necessary to specify the name of the components which results being connected in the RBD. Specifically, the only connections taken in account must be the ones that go from the left to the right of the diagram. Each one of them shall be defined in the CPACS file following this format:

- The first is the name of the component taken in account, followed by semicolon.
- Then shall be inserted the name of the components connected to the considered one. Each one of them shall be separated by semicolon.

An example can be done considering the electrical system, which is connected to five FCCs. This relationship shall be defined in the CPACS file as:

- Electrical System; ELAC1; ELAC 2; SEC 1; SEC 2; SEC 3.

By using this format, the tool will be able to produce an index matrix in which the electrical system results being connected to all the five flight control computers. Note that the name of each component must coincide with the ones defined in the previous part of the CPACS file (where the Weibull parameters have been assigned) to make the tool work correctly.

Table 2: FCS components failure rates [50][51]

FCS Component	Failure Rate [1/FH]
Side-Stick (Pilot/ Copilot)	$2 \cdot 10^{-6}$
Electrical System	$4.9 \cdot 10^{-10}$
ELACs/ SECs	$1 \cdot 10^{-4}$
Hydraulic Actuators	$2.5 \cdot 10^{-4}$
EHAs	$7.37 \cdot 10^{-5}$
EMAs	$1.37 \cdot 10^{-4}$
Hydraulic Systems (B/ G/ Y)	$5 \cdot 10^{-5}$
Electrical Lines (1/ 2)	$4.9 \cdot 10^{-7}$

Table 3: FCS components Weibull parameters

FCS Component	$\boldsymbol{\theta}$	$\boldsymbol{\beta_1}$	$\boldsymbol{\beta_2}$	$\boldsymbol{\beta_3}$
Side-Stick (Pilot/ Copilot)	$5 \cdot 10^{5}$	-	1	2.9
Electrical System	$2.04 \cdot 10^{9}$	-	1	2.5
ELACs/ SECs	$1 \cdot 10^{4}$	-	1	-
Hydraulic Actuators	$4 \cdot 10^{3}$	-	1	3
EHAs	$1.36 \cdot 10^{4}$	-	1	3
EMAs	$7.3 \cdot 10^{3}$	-	1	3.5
Hydraulic Systems (B/ G/ Y)	$2 \cdot 10^{4}$	-	1	3
Electrical Lines (1/ 2)	$2.04 \cdot 10^{6}$	-	1	2.5

The RBD of each architecture has been implemented to evaluate the reliability of three different flight phases: take-off, cruise and landing. No architectural changes have been taken in account during the entire mission. Therefore, the implemented RBDs are the same for each of the considered flight phases. The product of their reliabilities gives as result the mission reliability.

Table 4: FCS mission reliability results

FCS Architecture	Mission Reliability	Mission Failure Rate [1/FH]
Conventional	0.9999999923470577	$1.5305884618347298 \cdot 10^{-9}$
More Electric	0.9999999923470577	$1.5305884618347298 \cdot 10^{-9}$
All Electric	0.999999992347061	$1.5305877957009096 \cdot 10^{-9}$

The results shown in Table 4 have been defined considering a specified time (t^*) of five flight hours, which is a reasonable time for typical missions carried out by A320. The obtained failure rates result being greater than 10^{-9} due to a limit in the implementation of the tool. Indeed, it uses the same specified time to evaluate the reliability of each flight mission. Instead, it should consider different timespans and then evaluate the mission failure rate using their sum. In this way, the failure rate of each architecture would be less than 10^{-9}, as prescribed by certifying agencies for safety critical systems.

It is easy to notice that the obtained results are nearly the same for the three architectures under analysis. Specifically, reliabilities of Conventional and More Electric architectures results being the same. The reason is attributable principally to the RBDs configuration. Indeed, the different redundancy lines disposed in parallel increase the architectures reliability. But this growth is so high that the numbers which highlight the difference between the architectures are truncated or rounded due to the machine limits. To avoid this problem the RBDs have been modified, considering only the components necessary to control ailerons. The reduction of redundancy lines allowed to highlight the differences among the three architectures.

Table 5: FCS mission reliability results – ailerons control only

FCS Architecture	**Mission Reliability**	**Mission Failure Rate [1/FH]**
Conventional	0.999999235972191	$1.5280562017773832 \cdot 10^{-7}$
More Electric	0.9999992410585871	$1.5178834018667545 \cdot 10^{-7}$
All Electric	0.9999992423116318	$1.5153773104082516 \cdot 10^{-7}$

The results in Table 5 show that the implementation of EHAs and electrical lines to control the flight control surfaces involves an increase in reliability. Indeed, the More Electric architecture results being more reliable than the Conventional one, with a lower probability of failure occurring during a mission. The same can be said about the All Electric FCS, which is even more reliable than the More Electric.

Table 6: FCS mission reliability results – spoiler control only

FCS Architecture	**Mission Reliability**	**Mission Failure Rate [1/FH]**
Conventional	0.9999999763205695	$4.7358861615717856 \cdot 10^{-9}$
More Electric	0.9999999765414848	$4.6917031033175605 \cdot 10^{-9}$
All Electric	0.9999999900499632	$1.9900073687988613 \cdot 10^{-9}$

Even considering only the components necessary to provide spoiler control, the result is the same. As shown in Table 6, the more the FCS relies on electrical components (such as EHAs or EMAs), the more its reliability increases.

6.2. Overall Aircraft Results

After defining and comparing the mission reliability results, those ones have been integrated into an MDO environment. The aim has been to analyze the impact of each FCS architecture on the overall aircraft performance, such as masses, maintenance costs and operating costs. To better investigate these effects, the analyses performed in the MDO environment have taken in account five different instances:

- The A320 with Conventional FCS architecture.
- The A320 with More Electric FCS architecture.
- The A320 with All Electric FCS architecture and landing gear actuated hydraulically.
- The A320 with All Electric FCS architecture and landing gear actuated electrically.
- The A320 with All Electric FCS architecture, implementing EMAs instead of EHAs to actuate all the flight control surfaces.

The results shown in Table 7 highlight an increase of the A320 MTOW whit the implementation of More Electric and All Electric architectures. The principal reason of that is the increase of electrical system weight, which is greater than the reduction of the hydraulic system weight. Moreover, also the increase of the flight controls weight played an important role. It is caused by the implantation of EHAs and EMAs, whose weight is higher than the hydraulic actuators one. The complete absence of the hydraulic system in the AEA with landing gear electrically actuated enabled to reduce its MTOW, bringing it at the same level of the MEA. The implementation of EMAs in the last instance, instead, involved an increase in FCS weight so high that it could not be balanced by the absence of hydraulic system. The use of this one in the AEA with landing gear actuated hydraulically made its weight being higher than the one of MEA and of its counterpart with electrically actuated landing gear.

Table 7: Overall aircraft – Masses results

	Conv. Arch.	MEA	AEA (Hydr. Landing Gear)	AEA (Elect. Landing Gear)	AEA (Only EMAs)
MTOW [kg]	72098	74061	74345	74061	74830
Systems Mass [kg]	6404	7240	7406	7274	7638
Flight Controls Mass [kg]	447	661	910	910	1270
Hydr. Generation[kg]	103	67	20	0	0
Hydr. Distribution [kg]	584	377	113	0	0
Electric Generation [kg]	112	630	699	699	699
Electric Distribution [kg]	810	1150	1307	1309	1309

It can be noticed that the electrical system weight results being nearly the same in all the instances concerning the AEA. Moreover, the flight controls mass is the same in the AEA with landing gear actuated hydraulically and in the one with landing gear actuated electrically since both implement only EHAs. The data in Table 7 highlight that the most convenient among the innovative architectures result being the MEA and the AEA with landing gear actuated electrically since they would entail the lowest MTOW. This observation is valid even if the lowest weight of systems is granted by the MEA, which implements less EHAs.

The results in Table 8 show a decrease in terms of reliability and maintainability with the implementation of MEA and AEA. Indeed, compared to the Conventional architecture, they generate a growth of both aircraft failure rate and maintenance hours per flight hour. As already described in section 6.1, the FCS failure rate results being nearly the same, even if implementing different architectures. The hydraulic and the electrical systems failure rates, instead, change. Conversely to the FCS, their values have been evaluated using a statistical approach, which is described in section 2.3. Therefore, their failure rates change on varying of their respective weights and of the aircraft OEW. This relationship also influences the results concerning the aircraft failure rate and maintenance hours per flight hour. Consequently, the most reliable and maintainable architecture would result being the Conventional one. But this is in contrast with what has been discussed in section 6.1, which highlighted an increase of reliability with the implementation of MEA and AEA. Therefore, it can be concluded that the results shown in Table 8 are not completely trustworthy, since they are strongly affected by systems weight and by the aircraft OEW. However, they may allow to better understand which might be the order of magnitude of failure rate and maintenance hours concerning systems implemented on innovative architectures.

Table 8: Overall aircraft – RAMS results

	Conv. Arch.	MEA	AEA (Hydr. Landing Gear)	AEA (Elect. Landing Gear)	AEA (Only EMAs)
FCS Failure Rate $[1/10^9 \mathrm{FH}]$	1.5306	1.5306	1.5306	1.5306	1.5306
Hydr. System Failure Rate $[1/10^9 \mathrm{FH}]$	3.8425	4.0032	4.0311	0.0000	0.0000
Elect. System Failure Rate $[1/10^9 \mathrm{FH}]$	3.5469	3.6953	3.7210	5.9664	6.0658
Aircraft Failure Rate $[1/10^9 \mathrm{FH}]$	94.7365	98.6340	99.3111	98.6674	100.2858
Aircraft Maintenance Man Hours [MMH/FH]	0.8616	0.8700	0.8714	0.8697	0.8731

The fact that the FCS failure rate is not dependent from weights makes the RAMS analysis more trustworthy. Indeed, the capability of a system to perform correctly its duties should depend principally on the reliability of its components and on the current interactions among them and with other systems. Therefore, a change in weights should not affect systems dependability that much.

Finally, the results in Table 9 indicate how maintenance and operating costs vary depending on the implemented FCS architecture. The MEA, for example, produces an increase of costs if compared to the Conventional architecture, because of its significant use of both hydraulic and electrical systems. The AEA produces a similar effect. The use of electrically actuated landing gear enables a reduction of maintenance costs. Moreover, the reduction of the MTOW also reduced the fuel costs, with a consequent decrease of operating costs. Instead, the employment of a hydraulically actuated landing gear and of EMAs involved and increase of both maintenance and fuel costs. Therefore, the most convenient innovative architecture is AEA with landing gear actuated electrically. Moreover, the benefits that it can provide in terms of safety and reliability makes it the most valid alternative to the Conventional architecture.

Table 9: Overall aircraft – Costs results

	Conv. Arch.	MEA	AEA (Hydr. Landing Gear)	AEA (Elect. Landing Gear)	AEA (Only EMAs)
Direct Maintenance Costs [$/FH]	702	711	713	705	708
Maintenance Burden Costs [$/FH]	468	474	475	470	472
Total Maintenance Costs [$/FH]	1171	1186	1188	1174	1180
Fuel dollars [$/FH]	2710	2821	2829	2815	2852
Crew Costs [$/FH]	340	340	340	340	340
Operating Costs[$/FH]	4221	4346	4356	4329	4372

7. Conclusions

The research work described in this thesis aimed to define a methodology which enabled to perform RAMS analyses following a model-based approach. Those analyses could be then employed to evaluate the dependability of innovative on-board systems. Finally, the obtained results could be integrated into an MDO environment to estimate the effects of innovative architectures on the overall aircraft performance and operating costs.

Four different RAMS analyses have been taken in account. The FHA has been employed to evaluate safety requirements; the FTA and FMEA have been used to assess the system safety; the RBD has been implemented to calculate the system reliability. Each one of them has been accurately analyzed and some guidelines have been developed to enable performing them following a model-based approach. Those guidelines specify how to use the elements of SysML language to represent the essential information of the considered RAMS analyses. In addition, they also specify how to extract these data and relate them to the respective documents.

The model-based RAMS analyses have been then applied to three different FCS architectures: Conventional, More Electric and All Electric. Safety and reliability that they can respectively provide have been evaluated and it has been possible to highlight the greater dependability of the last two architectures. Specifically, the More Electric one results being the safer. The reason is in the implementation of both hydraulic and electrical systems, which allows to move the flight control surfaces with a less probability of totally losing their control.

Afterwards, safety and reliability results concerning the three architectures have been integrated into an MDO environment. In this way, it has been possible to evaluate and compare their impact on the aircraft performance and costs. It has come to light that the more safety offered by the MEA also involved an increase of the overall weights and costs. The AEA, instead, if combined with landing gear actuated electrically, involved a reduction of weights and operating costs.

In conclusion, the SysML turned out being a valid instrument, which also enabled to integrate the RAMS analyses into already existing models. In this way, it has been possible to represent the system architecture and the analyses performed in only one model. In addition, the SysML offered the possibility to create new models starting from the ones that have already been created. Therefore, it made easier the development and the analysis of innovative on-board systems.

Despite the results gained from this research work, different improvements can be developed:

- Develop new model-based approaches which could enable to perform also other RAMS analyses (such as Markov Analysis, Common Cause Analysis, Zonal Safety Analysis, etc.)
- Define a method to connect the SysML elements used to perform the RAMS analyses, so that the model could result easier to understand and to manage.

- Find a way to implement the Foundational UML subset with the aim of improving the model-based RAMS analyses through the execution of models.
- Investigate how to develop more accurate RBDs, taking in account also the different functions that each system shall perform and their impact on the overall system reliability.
- Improve tool integrated into the MDO environment, making it capable of taking in account different timespans for each considered flight phase.
- Improve tool integrated into the MDO environment, making it also capable of solving complex RBD configurations.

The potential future works concerning these improvements may enable to employ in a better way the model-based approach and to obtain more accurate results from the integration with the MDO environment.

A part of this research works describe has been submitted to the American Institute of Aeronautics and Astronautics (AIAA) and has been published as a meeting paper in the session of Model-Based Systems Engineering (MBSE) Integration with MDO I [52].

8. Acknowledgments

As conclusion to this thesis work, I want to thank properly all the people that contributed to make me achieve this goal. I thank the Prof. Fioriti for giving me the possibility to take part in this research work and for guiding me during its development. I thank Luca and Pina for their help, their unlimited availability and the useful advices they gave during the period I spent in Hamburg. Also, I thank all the other DLR employees that helped me enjoy the permanence in the company. A special thank goes to all the Europa Center guys, which allowed me to experience a lot of funny moments, both during working and free time. I will never forget the time spent together and I hope to have fun with you again (I also will never forget Reeperbahn, thanks to a certain person). I want to thank my parents for all the emotional and economical support they gave me, despite the countless difficulties that showed up. Thanks for all the advices you gave me along this troubled path. I thank my sister for taking care of my room as if it has been hers, during all these years away from home. I also thank my grandmother and my uncles for all the Apulian support they gave me (including the great food and the long procrastination times), as no one else could do. I want to really thank everyone for helping me achieving this important goal.

Last, but not least, I want to do a very special thanks to the person that pushed me following this path form the very beginning. From now on, I address directly to you in first person. I want to thank you for telling me that I was wasted in the place where we lived and that I could do more. I still do not know if you were right or not, but that thought was enough for me to ignore the others way of thinking and make me give a try. Therefore, thank you for walking two thousand kilometers together. Thank you for passing an entire night completely lost in Turin. Thank you for making me wait hours in the queues at the EXPO, just for taking a little box of salt and taste the new lime coke. Thank you for all the pizzas that we prepared and shared. Thank you for taking good care of me when I was sick and for supporting me during the stressful periods caused by university. Thank you for helping me when I did not have a place where to stay and when I was scared in Germany. Thanks for sharing with me all your drawing and for listening every strange song that I tried to arrange. Thank you for all the time spent playing together as two kids. Thank you for thinking that my way of being different is not a flaw. Thank you for all the adventures we had together during these six years. I do not know what will happen in the future, but I hope we can grow together and have more of them.

9. References

[1] Patrick W. Wheeler, Jon C. Clare, Andrew Trentin and Serhiy Bozhko, "An overview of the more electrical aircraft," *Journal of Aerospace Engineering*, Vol. 227, No. 4, Apr. 2013, pp. 578-585.
DOI: 10.1109/DASC.1993.283509

[2] Vincenzo Madonna, Paolo Giangrande, Michael Galea, "Electrical Power Generation in Aircraft: Review, Challenges, and Opportunities," *IEEE Transactions on Transportation Electrification*, Vol. 4, No. 3, Sept. 2019, pp. 646-659
DOI: 10.1109/TTE.2018.2834142

[3] J.A. Weimer, "Electrical Power Technology for the More Electric Aircraft," *AIAA/IEEE Digital Avionic System Conference*, IEEE, Fort Worth, TX, USA, 1993.
DOI: 10.1109/DASC.1993.283509

[4] J.A. Rosero, J.A. Ortega, E. Aldabs, L. Romeral, "Moving Towards a More Electric Aircraft," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 22, No. 3, Mar. 2007, pp. 3-9.
DOI: 10.1109/MAES.2007.340500

[5] Bulent Sarlioglu, Casey T. Morris, "More Electric Aircraft: review, Challenges, and Opportunities for Commercial Transport Aircraft," *IEEE Transactions on Transportation Electrification*, Vol. 1, No. 1, Jun. 2015, pp. 54-64.
DOI: 10.1109/TTE.2015.2426499

[6] Luca Boggero, "Design techniques to support aircraft systems development in a collaborative MDO environment," Ph.D. Thesis, *Graduate School of Polytechnic of Turin (ScuDo)*, Polytechnic University of Turin, Turin, Italy, Jul. 2018.

[7] Marco Fioriti, Luca Boggero, Sabrina Corpino, Prajwal Shiva Prakasha, Pier Davide Ciampa, Björn Nagel, "The effect of sub-systems design parameters on preliminary aircraft design in a multidisciplinary design environment," *Transportation Research Procedia*, Vol. 29, 2018, pp. 135-145.

[8] Mike Sinnett, "787 No-Bleed Systems: Saving Fuel and Enhancing Operational Efficiencies," *AERO Magazine*, QTR_04 07, published online Oct. 2007, pp. 6-11.

[9] Dominique van den Bossche, "The A380 Flight Control Electro-Hydrostatic Actuators, Achievements and Lesson Learnt," *25th International Congress of the Aeronautical Sciences*, ICAS, Hamburg, Germany, 2006.

[10] Joaquim R. R. A. Martins, Andrew B. Lambe, "Multidisciplinary Design Optimization: A Survey of Architectures," *AIAA Journal*, Vol. 51, No. 9, Sept. 2013, pp. 2049-2075.
DOI: 10.2514/1.J051895

[11] Dr. John C. Conlon, Mr. Walter A. Lilius, Lt. Col. Frank H. Tubbesing, Jr., USAF, *Test Evaluation of System Reliability, Availability and Maintainability*, 3rd ed., Office of the Director Defense Test and Evaluation Under Secretary of Defense for Research and Engineering, 1982.

[12] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, "Fundamental Concepts of Dependability," Report no. CS-TR-739, Department of Computer Science, University of Newcastle upon Tyne, 2001.

[13] U.S. Department of Defense, *DoD Guide for Achieving Reliability, Availability and Maintainability*, Washington, DC, USA, 3 Aug. 2005.

[14] Gary A. Pryor, "Methodology for Estimation of Operational Availability as Applied to Military Systems," *ITEA Journal*, Vol. 29, 2008, pp. 420-428.

[15] U.S. Military Standard, *MIL-STD-882E*, "System Safety," *Department of Defense*, Washington, D.C., 2012.

[16] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, Carl Landwehr, "Basic Concepts and Taxonomy of dependable and Secure Computing," *IEEE transactions on dependable and secure computing*, Vol. 1, No. 1, 2004, pp. 11-33.
DOI: 10.1109/TDSC.2004.2

[17] Enrico ZIO, Mengfei FAN, Zhiguo ZENG, Rui KANG, "Application of reliability technologies in civil aviation: lessons learnt and perspectives," *Chinese Journal of Aeronautics*, Vol. 32, No. 1, Jan. 2019, pp. 143-158.
DOI: 10.1016/j.cja.2018.05.014

[18] Aerospace Standard, *ARP4761*, "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," *SAE International*, 1996.

[19] U.S. Military Standard, *MIL-HDBK-217F*, "Reliability Prediction of Electronic Equipment," *Department of Defense*, Washington, D.C., Dec. 1991.

[20] Guru Prasad PANDIAN, Diganta DAS, Chuan LI, Enrico ZIO, Michael PECHT, "A critique to reliability prediction techniques for avionics applications," *Chinese Journal of Aeronautics*, Vol. 31, No. 1, Jan. 2018, pp. 10-20.
DOI: 10.1016/j.cja.2017.11.004

[21] Sergio Chiesa, *Affidabilità, sicurezza e manutenzione nel progetto di sistemi*, 2nd ed., CLUT, Turin, 2008.

[22] Ana Luísa Ramos, José Vasconcelos Ferreira, Jaume Barceló, "Model-Based Systems Engineering: An Emerging Approach for Modern Systems," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, Vol. 42, No. 1, Jan. 2012, pp. 101-111.
DOI: 10.1109/TSMCC.2011.2106495

[23] Sanford Friedenthal, Alan Moore, Rick Steiner, *A Practical Guide to SysML*, 2nd ed., Morgan Kaufmann, 2011.

[24] *OMG Systems Modeling Language (OMG SysML)*, OMG Std., June 2012.

[25] Delligatti Lenny, *SysML Distilled: A Brief Guide to the Systems Modeling Language*, Addison-Wesley, 2013.

[26] Aerospace Standard, *ARP4754A*, "Guidelines for Development of Civil Aircraft and Systems," *SAE International*, 2010.
DOI: 10.4271/ARP4754A

[27] Adam Scharl, Kevin Stottlar, Rani Kady, "Functional Hazard Analysis (FHA) Methodology Tutorial," *International System Safety Training Symposium*, St. Louis, Missouri, 2014.

[28] Jeffrey W. Vincoli, *Basic Guide to System Safety*, John Wiley and Sons, 2006.

[29] P. J. Wilkinson, T. P. Kelley, "Functional Hazard Analysis for Highly Integrated Aerospace Systems," 1998.

[30] Dr. Michael Stamatelatos, Dr. William Vesely, Dr. Joanne Dugan, Mr. Joseph Fragola, Mr. Joseph Minarick III, Mr. Jan Railsback, *Fault Tree Handbook with Aerospace Applications*, NASA Office of Safety and Mission Assurance, Washington, D.C., Aug. 2002.

[31] W. E. Vesely, et al., *Fault tree handbook*, U.S. Nuclear Regulatory Commission, Washington, D.C., 1981.

[32] U.S. Military Standard, *MIL-STD-1629A*, "Procedures for Performing a Failure Mode, Effect and Criticality Analysis," *Department of Defense,* Washington, D.C., 1980.

[33] Carl S. Carlson, *Effective FMEAs: Achieving Safe, Reliable, and Economical Products and Processes using Failure Mode and Effect Analysis*, John Wiley and Sons, 2012.

[34] Apollo Program, *RA-006-013-1A*, "Procedure for Failure Mode, Effects and Criticality Analysis (FMECA)", *NASA*, Washington, D.C., Aug. 1966.

[35] Paul O. Chelson, R. Eric Eckstein, *Reliability Computation Form Reliability Block Diagrams*, Technical Report no. 32-1543, NASA Jet Propulsion Laboratory, Pasadena, California, Dec. 1971.

[36] Masha Aghaei, Ali Zenial Hamadani, Mostafa Abouei Ardakan, "Redundancy allocation problem for *k*-out-of-*n* systems with a choice of redundancy strategies," *Journal of Industrial Engineering International*, Vol. 13, No. 1, Mar. 2017, pp. 81-92.

[37] Joshi, Anjali, Mats P.E. Heimdahl, "Model-based Safety Analysis of Simulink Models Using SCADE Design Verifier," *Department of Computer Science and Engineering*, University of Minnesota, Minneapolis, 2005.

[38] P. David, V. Idasiak and F. Kratz, "Reliability study of complex physical systems using SysML," *Reliability Engineering & System Safety,* vol. 95, no. 4, pp. 431-450, 2010.

[39] M. Izygon, H. Wagner, S. Okon, L. Wang, M. Sargusingh, J. Evans, "Facilitating R&M in spaceflight systems with MBSE," *Annual Reliability and Maintainability Symposium (RAMS),* pp. 1-6, 2016.

[40] E. Brusa, D. Ferretto, C. Stigliani and C. Pessa, "A model based approach to design for reliability and safety of critical aeronautic systems," in *Proceedings of INCOSE Conference on System Engineering*, Turin (IT), 2016.

[41] F. Mhenni, J. Y. Choley, N. Nguyen, "Extended mechatronic systems architecture modeling with SysML for enhanced safety analysis," in *IEEE International Systems Conference Proceedings*, 2014.

[42] A. H. de Andrade Melani and G. F. de Souza, "Obtaining Fault Trees Through SysML Diagrams: A MBSE Approach for Reliability Analysis," 2002.

[43] Daniela Angelica Giovingo, "RAMS and Maintenance cost assessment in a Multidisciplinary Design Optimization environment," M.Sc. Thesis, *Department of Mechanical and Aerospace Engineering*, Polytechnic University of Turin, Italy, 2019.

[44] Charles E. Ebeling, *An Introduction to Reliability and Maintainability Engineering*, 2nd ed., McGraw-Hill, 2003.

[45] Xiao Liu, Yi Ren, Zili Wang, Linlin Liu, "Modelling Method of SysML-based Reliability Block Diagram," *International Conference on Mechatronic Sciences, Electronic, Engineering and Computer (MEC)*, Shenyang, China, Dec. 2013, pp. 206-209.

[46] DLR, Institute of System Architectures in Aeronautics , "CPACS - A Common Language for Aircraft Design," [Online]. Available: http://cpacs.de. [Accessed 26 04 2020]

[47] Atul Garg, Rezawana Islam Linda, Tonoy Chowdhury, "Evolution of Aircraft Flight Control System and Fly-By-Light Flight Control System," *International Journal of Emerging Technology and Advanced Engineering*, Vol. 3, No. 12, Dec 2013, pp. 60-64.

[48] Ian Moir, and Allan Seabridge, *Aircraft systems: Mechanical, electrical, and avionics subsystems integration*, 3rd ed., John Wiley & Sons, 2008.

[49] "Airbus A319-320-321 [Flight Controls]," *Smart Cockpit* [online document]. Available: http://www.smartcockpit.com/docs/A319-320-321-Flight_Controls.pdf [Accessed on: Oct. 2019].

[50] Ian Moir, Allan Seabridge, Malcom Juckes, *Civil Avionics Systems*, 2nd ed., John Wiley and Sons, 2013.

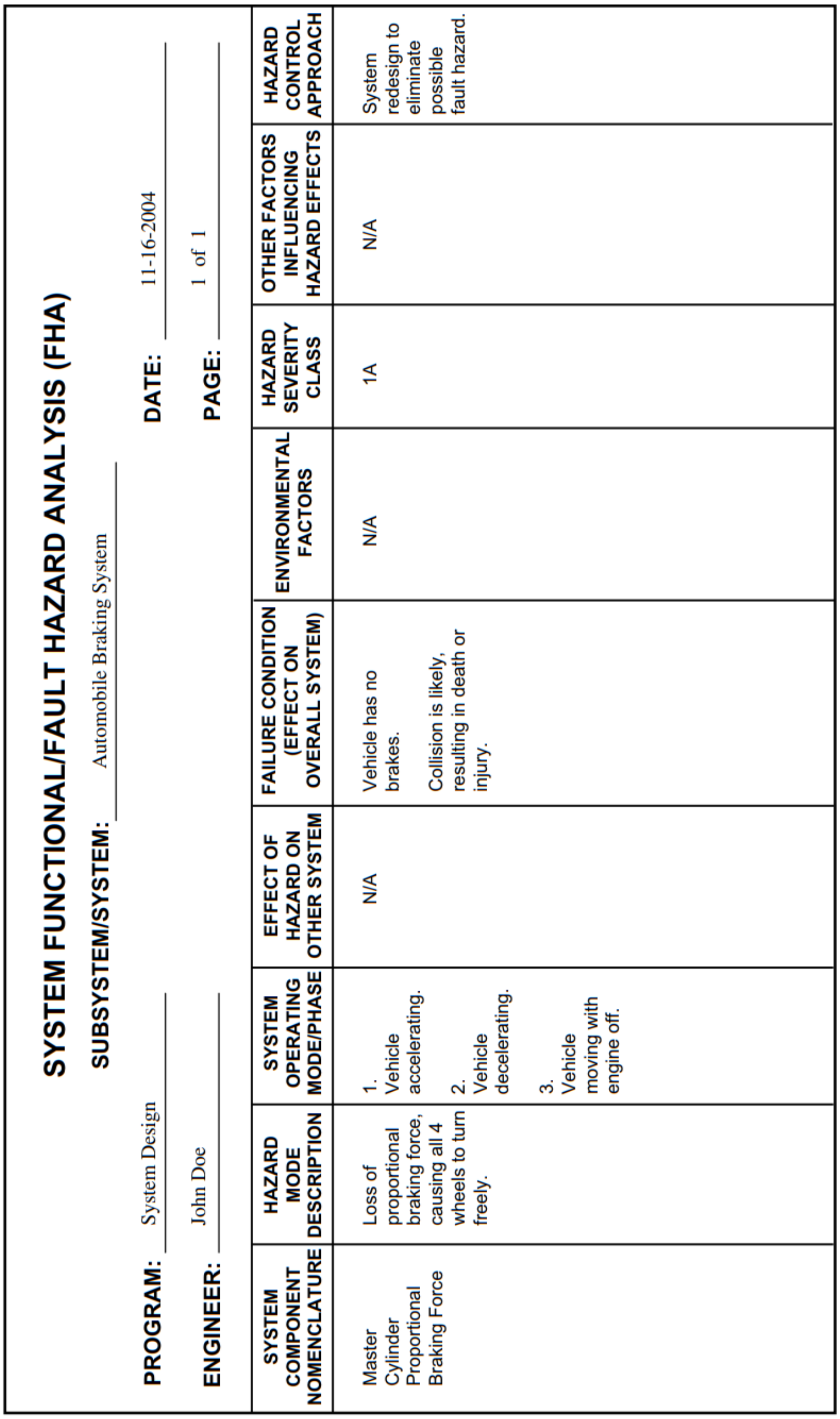
[51] Xue Longxian, "Actuation Technology for Flight Control System on Civil Aircraft," M.Sc. Thesis, Cranfield Institute of Technology, England, 2009.

[52] Francesco Bruno, Marco Fioriti, Giuseppa Donelli, Luca Boggero, Pier Davide Ciampa, Björn Nagel, "A Model-Based RAMS Estimation Methodology for Innovative Aircraft on-board Systems supporting MDO Applications," *AIAA 2020 Aviation Forum*, June 2020. DOI: 10.2514/6.2020-3151

Appendix A

Appendix A contains examples of instruments necessary to perform RAMS analyses following a document-based approach (such as FMEA and FHA worksheets).

A.1. FHA Worksheet Examples



SYSTEM FUNCTIONAL/FAULT HAZARD ANALYSIS (FHA)

SUBSYSTEM/SYSTEM: Automobile Braking System

PROGRAM: System Design

ENGINEER: John Doe

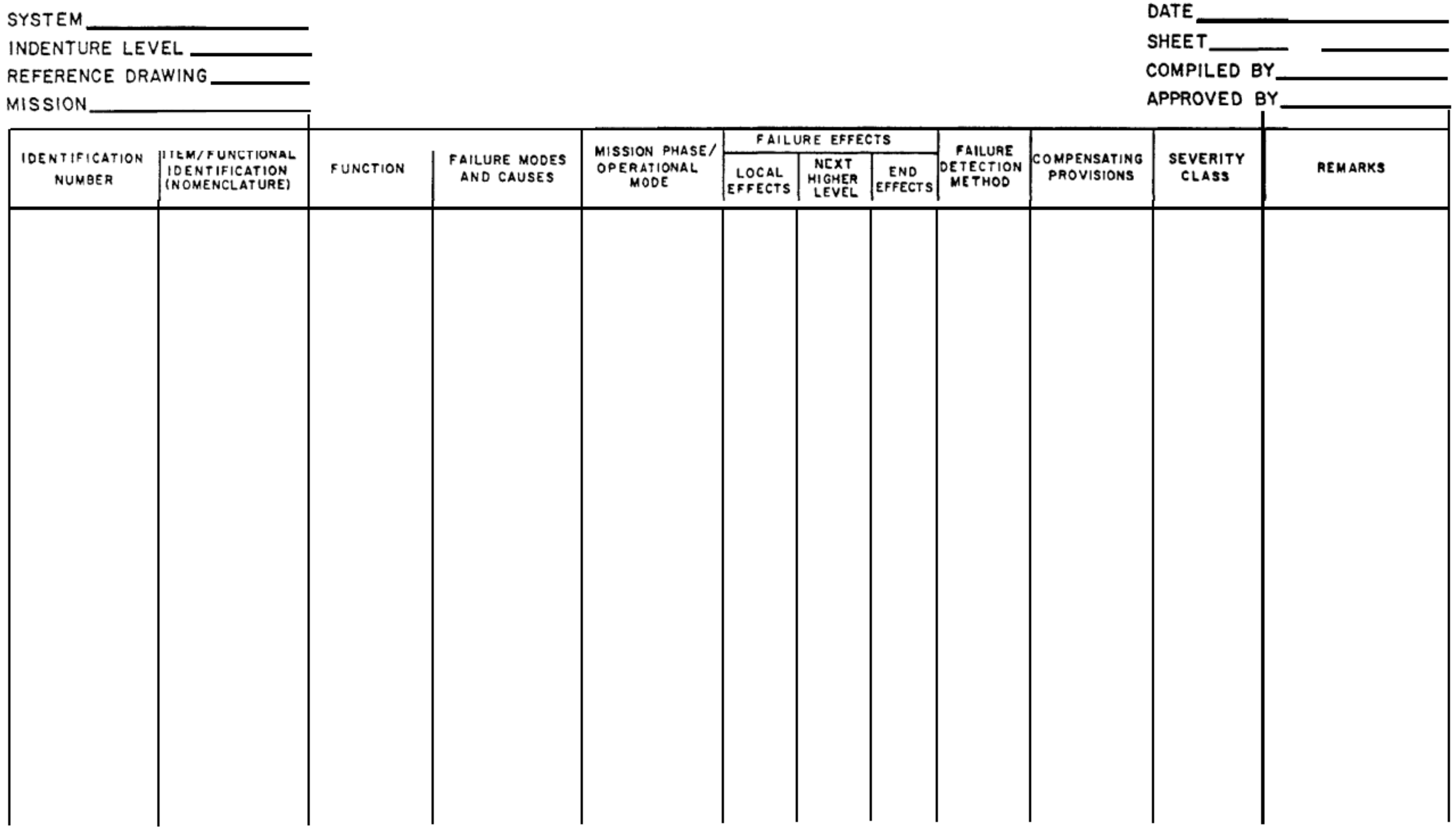
DATE: 11-16-2004

PAGE: 1 of 1

SYSTEM COMPONENT NOMENCLATURE	HAZARD MODE DESCRIPTION	SYSTEM OPERATING MODE/PHASE	EFFECT OF HAZARD ON OTHER SYSTEM	FAILURE CONDITION (EFFECT ON OVERALL SYSTEM)	ENVIRONMENTAL FACTORS	HAZARD SEVERITY CLASS	OTHER FACTORS INFLUENCING HAZARD EFFECTS	HAZARD CONTROL APPROACH
Master Cylinder Proportional Braking Force	Loss of proportional braking force, causing all 4 wheels to turn freely.	1. Vehicle accelerating. 2. Vehicle decelerating. 3. Vehicle moving with engine off.	N/A	Vehicle has no brakes. Collision is likely, resulting in death or injury.	N/A	1A	N/A	System redesign to eliminate possible fault hazard.

Fig. 61: FHA Worksheet example [28]

A.2. FMEA Worksheet Examples



SYSTEM__________
INDENTURE LEVEL__________
REFERENCE DRAWING__________
MISSION__________

DATE__________
SHEET_____ _____
COMPILED BY__________
APPROVED BY__________

IDENTIFICATION NUMBER	ITEM/FUNCTIONAL IDENTIFICATION (NOMENCLATURE)	FUNCTION	FAILURE MODES AND CAUSES	MISSION PHASE/ OPERATIONAL MODE	FAILURE EFFECTS			FAILURE DETECTION METHOD	COMPENSATING PROVISIONS	SEVERITY CLASS	REMARKS
					LOCAL EFFECTS	NEXT HIGHER LEVEL	END EFFECTS				

Fig. 62: FMEA worksheet format example [32]

Item	Function(s)	Potential Failure Mode	Potential Effect(s) of Failure	Severity	Potential Cause(s) of Failure	Occurrence	Current Design Controls (Prevention)	Current Design Controls (Detection)	Detection	RPN	Recommended Action(s)	Responsible Person	Actions Taken	Revised Rankings			
												Target Completion Date	Effective Completion Date	Severity	Occurrence	Detection	RPN

Fig. 63: Design FMEA worksheet format example [33]

Appendix B

B.1. Reliability Calculation

Reliability can be mathematically expressed with a function:

$$R(t) = Pr\{T \geq t\} \tag{9.1}$$

It is called *Reliability function* and represents the probability that time to failure T is greater than or equal to a given time t. Its complementary is the *Cumulative Distribution Function* (CDF), which can be expressed as:

$$F(t) = 1 - R(t) = Pr\{T < t\} \tag{9.2}$$

The CDF represents the probability that a failure occurs before a given time t. Those two functions can be linked to a third one, called *Probability Density Function* (PDF):

$$f(t) = \frac{dF(t)}{dt} = -\frac{dR(t)}{dt} \tag{9.3}$$

This one describes the shape of the failure distribution. It is important since it is always greater than or equal to zero and the area beneath the curve that describes is always equal to one. Both Reliability function and CDF are affected by these properties, due to their relationship with the PDF:

$$F(t) = \int_0^t f(t')dt' \tag{9.4}$$

$$R(t) = \int_t^\infty f(t')dt' \tag{9.5}$$

Both relationships represent two areas under the curve described by the PDF and their sum must be equal to one. That means that their value cannot be less than zero nor more than one:

$$0 \leq F(t) \leq 1 \tag{9.6}$$

$$0 \leq R(t) \leq 1 \tag{9.7}$$

Distribution of Reliability function, CDF and PDF are respectively represented in Fig. 64, Fig. 65, and Fig. 66:

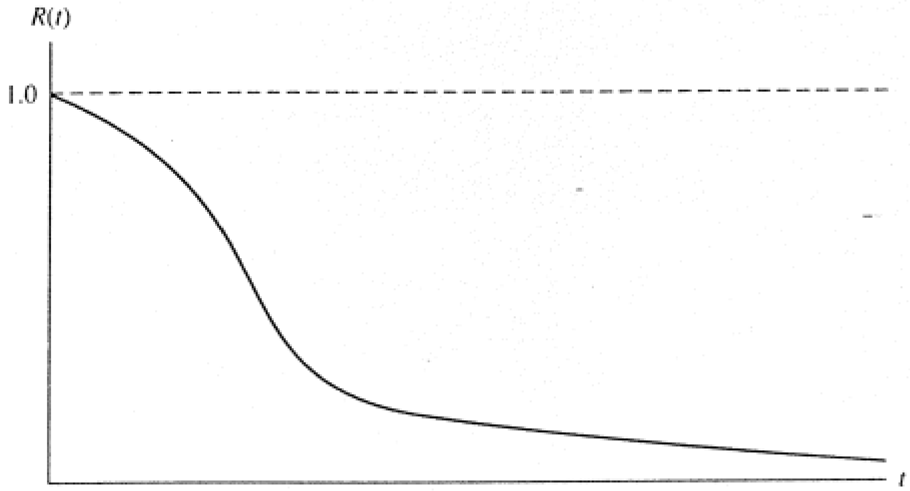


Fig. 64: Reliability Function [44]

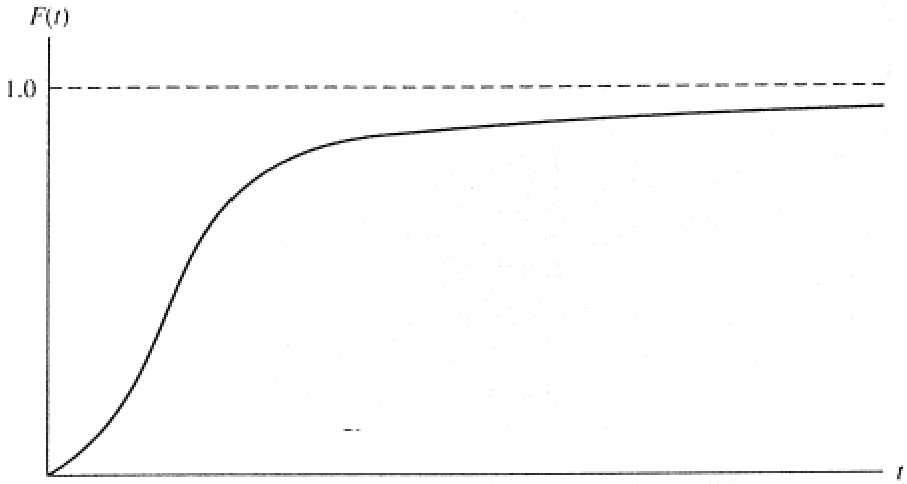


Fig. 65: Cumulative Distribution Function (CDF) [44]

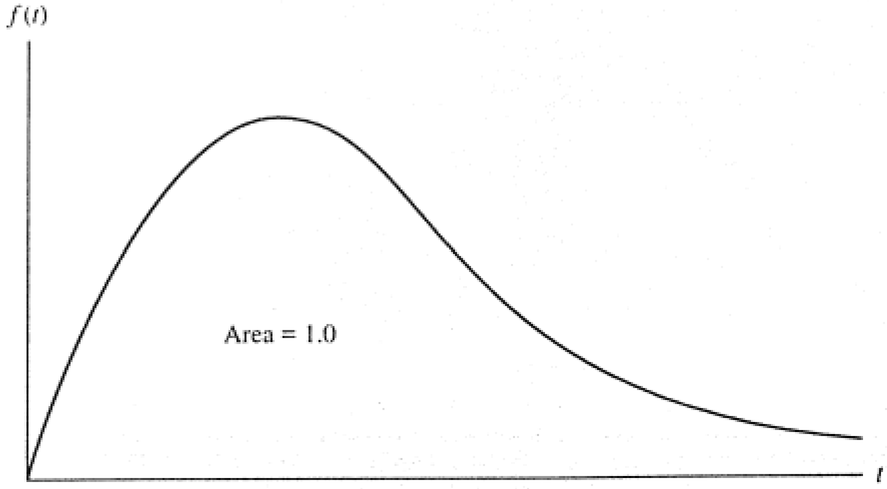


Fig. 66: Probability Distribution Function (PDF) [44]

The mean of the distribution defined by the PDF is the Mean Time to Failure (MTTF) which can be expressed as:

$$MTTF = \int_0^\infty t \cdot f(t)dt \tag{9.8}$$

It can also be related to the reliability function with the following relationship:

$$MTTF = \int_0^\infty R(t)d \tag{9.9}$$

Another important function used in reliability is the *failure rate function* (also called *hazard rate function*), usually represented with $\lambda(t)$. It provides a rate of failure distribution, which can be increasing, decreasing, or constant.

Appendix C

Appendix C contains most of the SysML diagrams developed during this thesis. They concern the three FCS architectures representations and their respective model-based RAMS analysis that have been performed.

C.1. Commercial airliner FCS components – SysML Diagrams

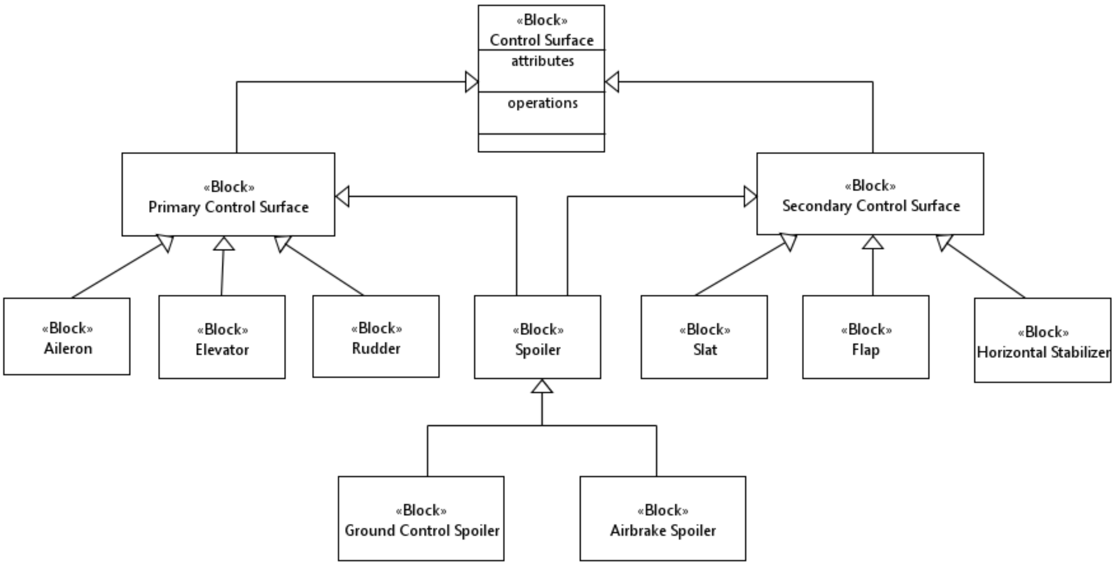


Fig. 67: Flight Control Surfaces Blocks definition – BDD

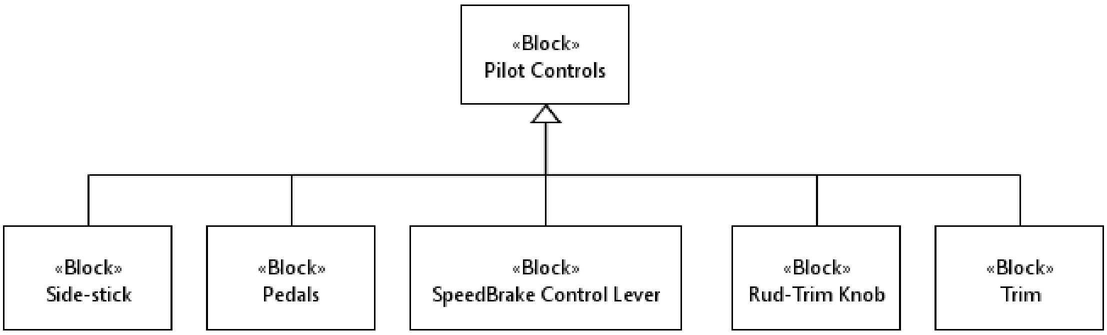


Fig. 68: Pilot Controls Blocks definition – BDD

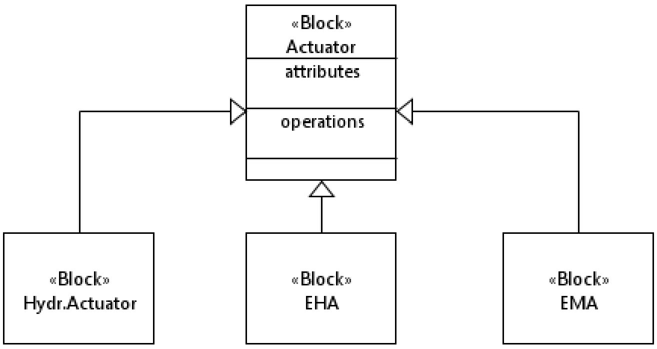


Fig. 69: Actuators Blocks definition – BDD

C.2. A320 FCS More Electric Architecture – SysML Diagrams

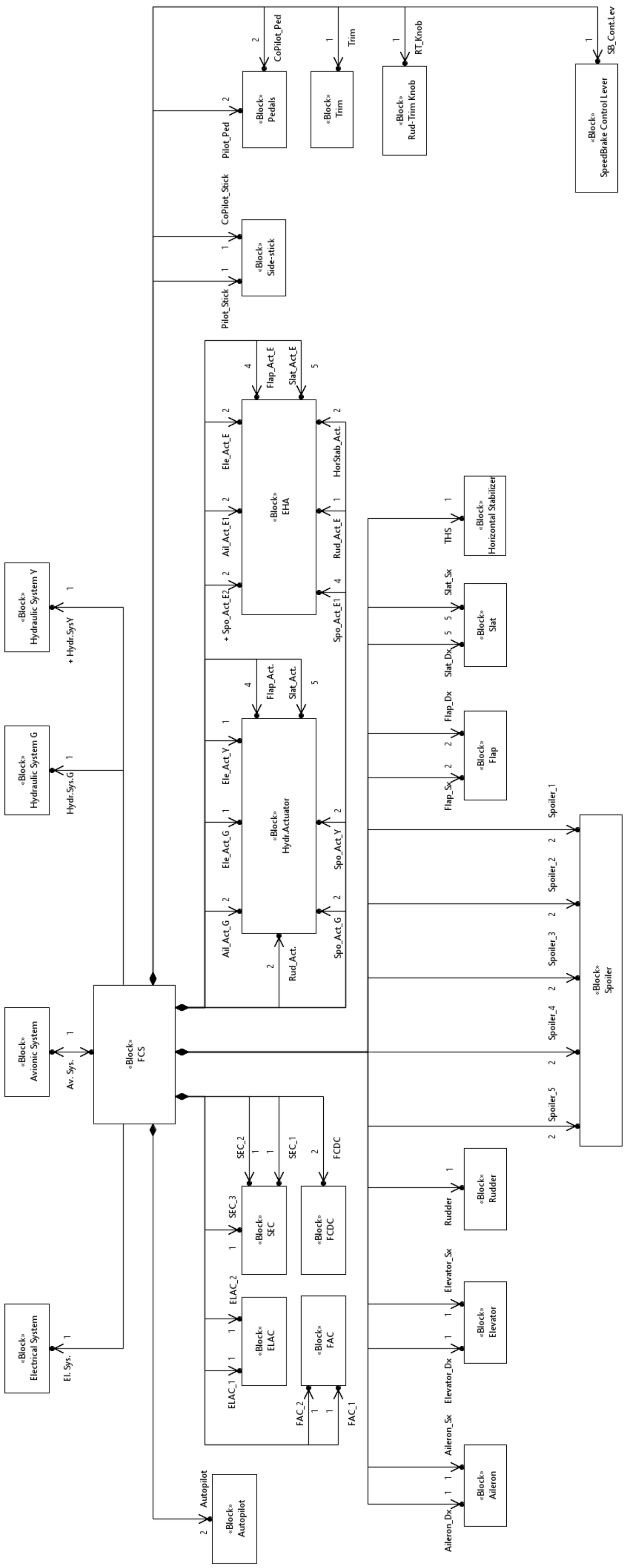


Fig. 70: More Electric A320 FCS BDD

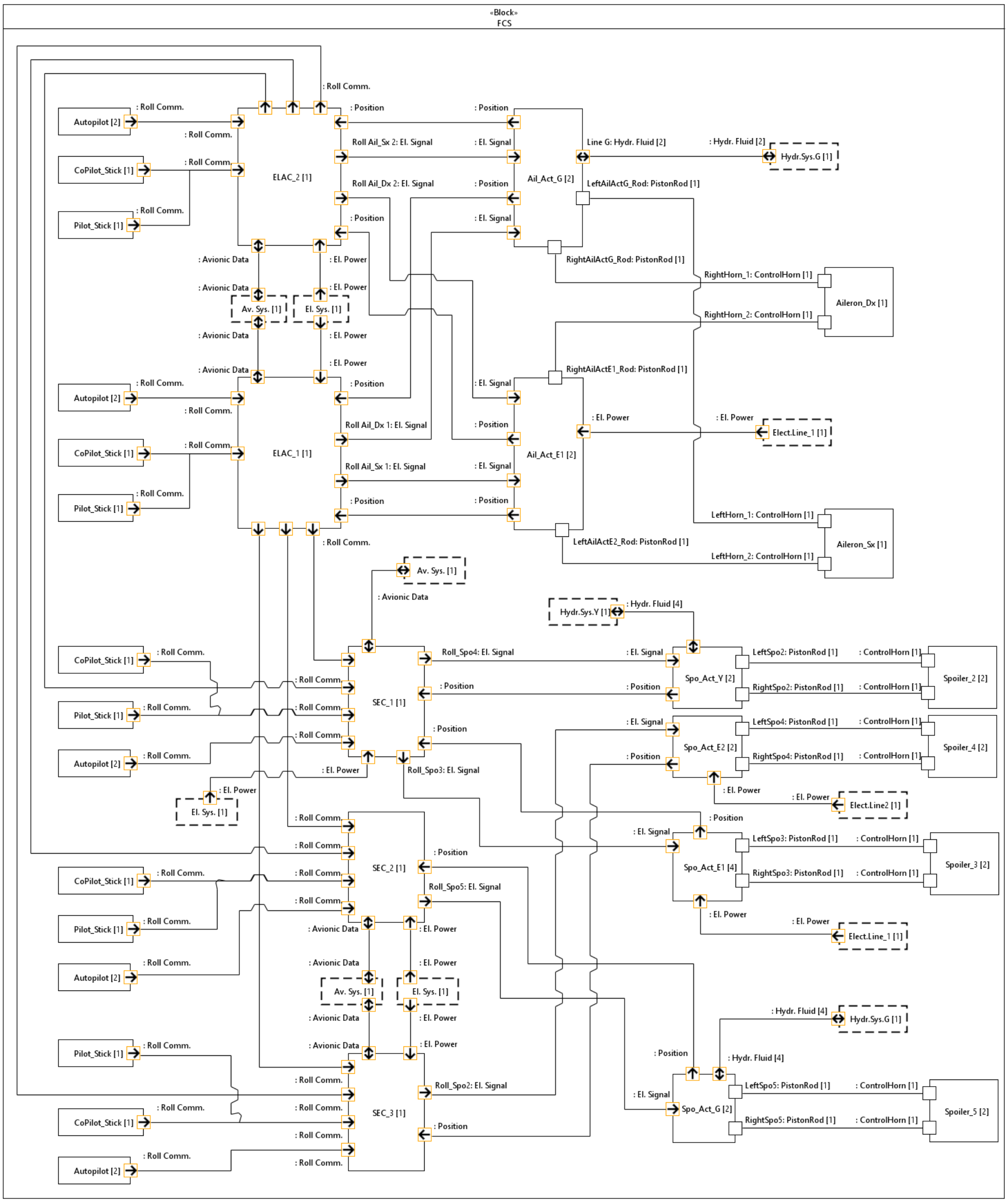


Fig. 71: More Electric A320 FCS IBD – focused on Roll Control

C.3. A320 FCS All Electric Architecture – SysML diagrams

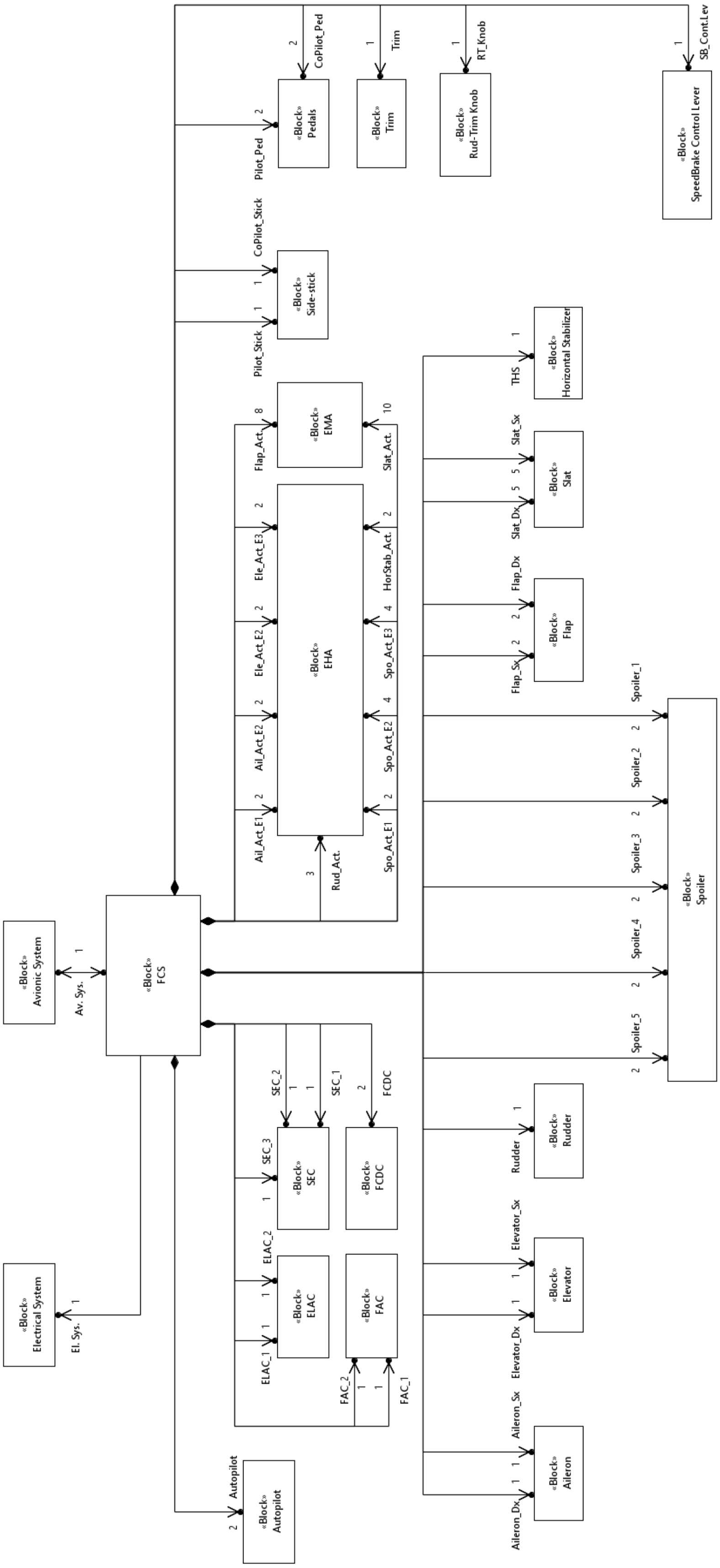


Fig. 72: All Electric A320 FCS BDD

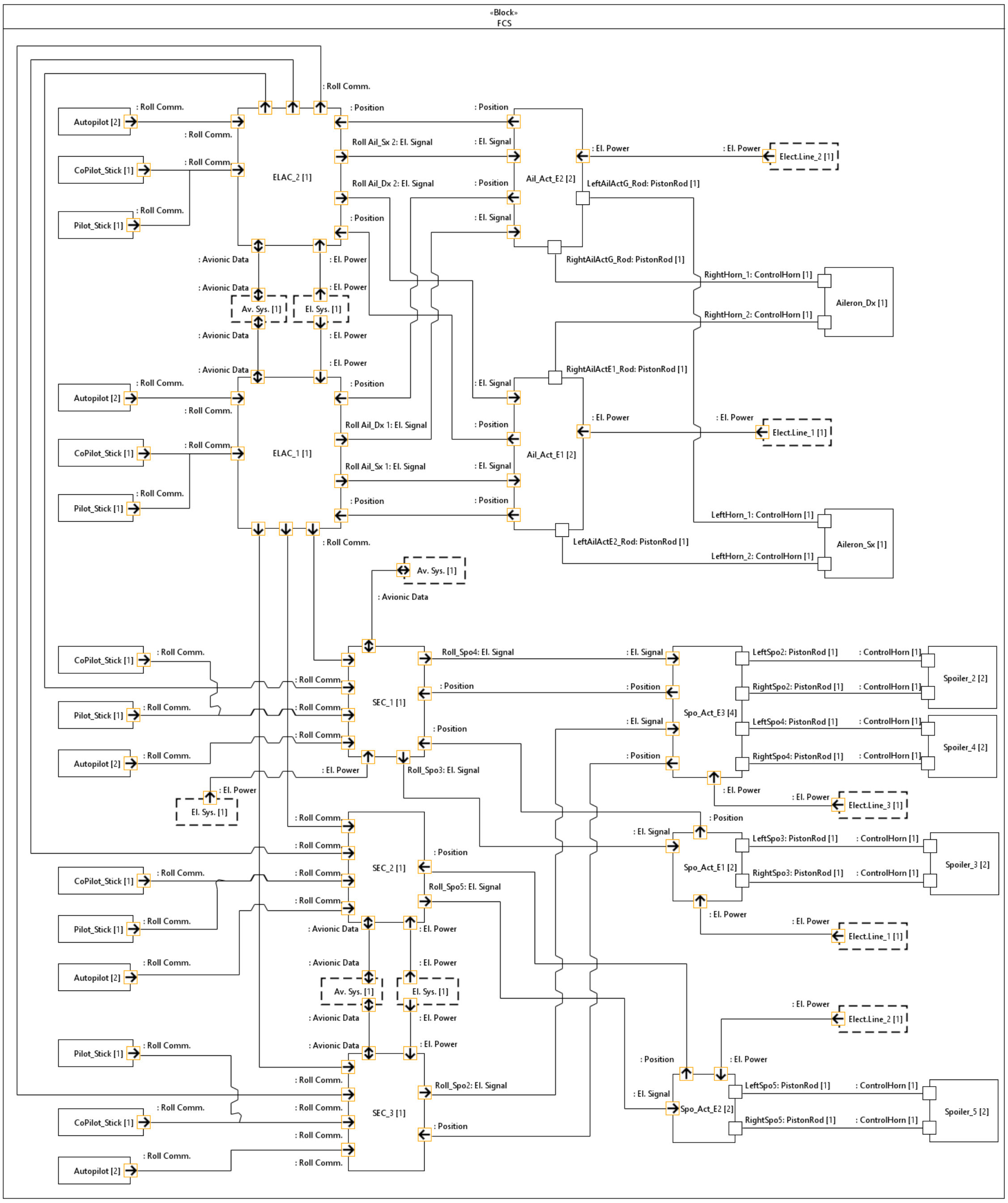


Fig. 73: All Electric A320 FCS IBD – focused on Roll Control

C.4. A320 FCS – Model-Based FHA

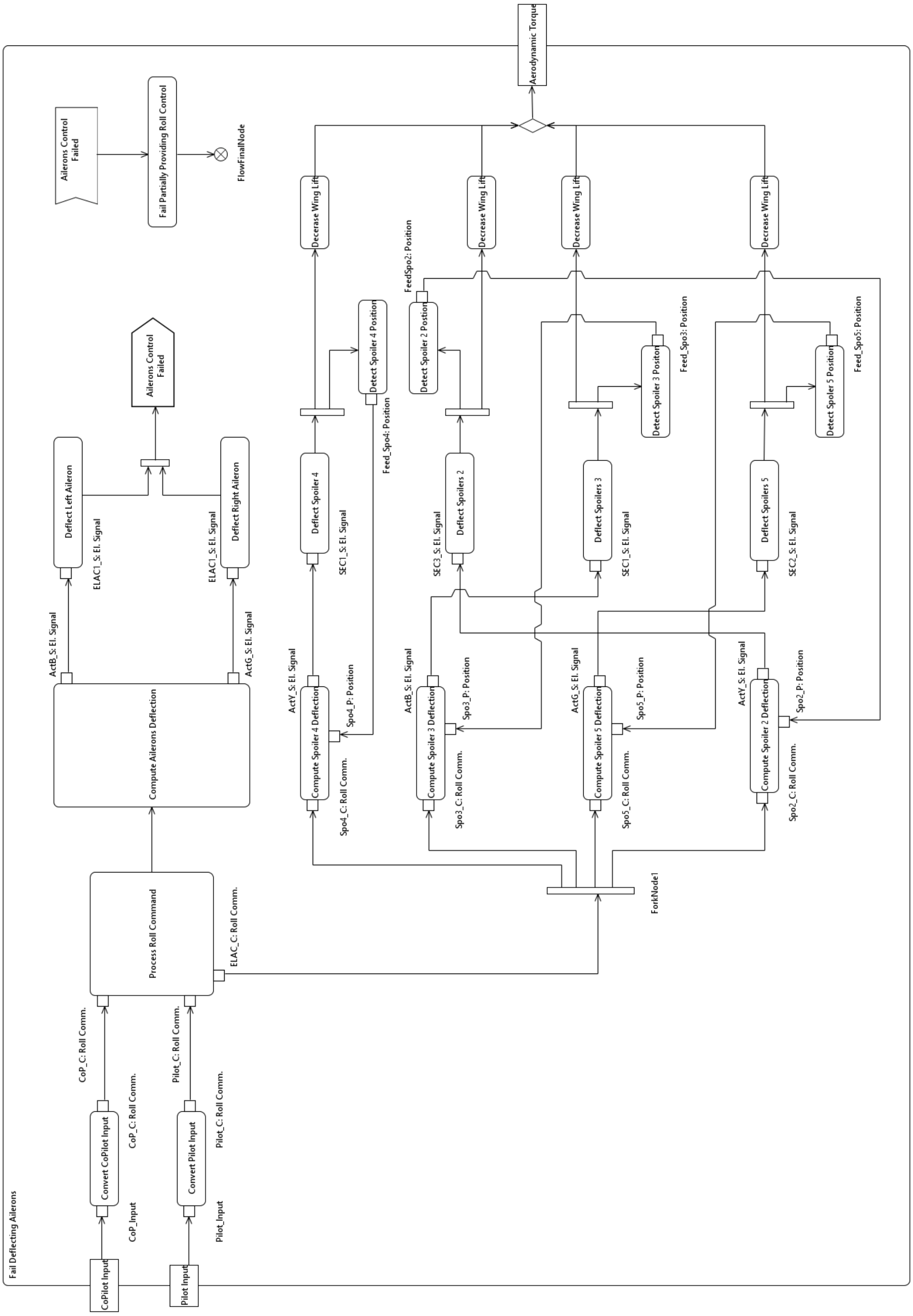


Fig. 74: FCS Model-Based FHA – Ailerons control failure

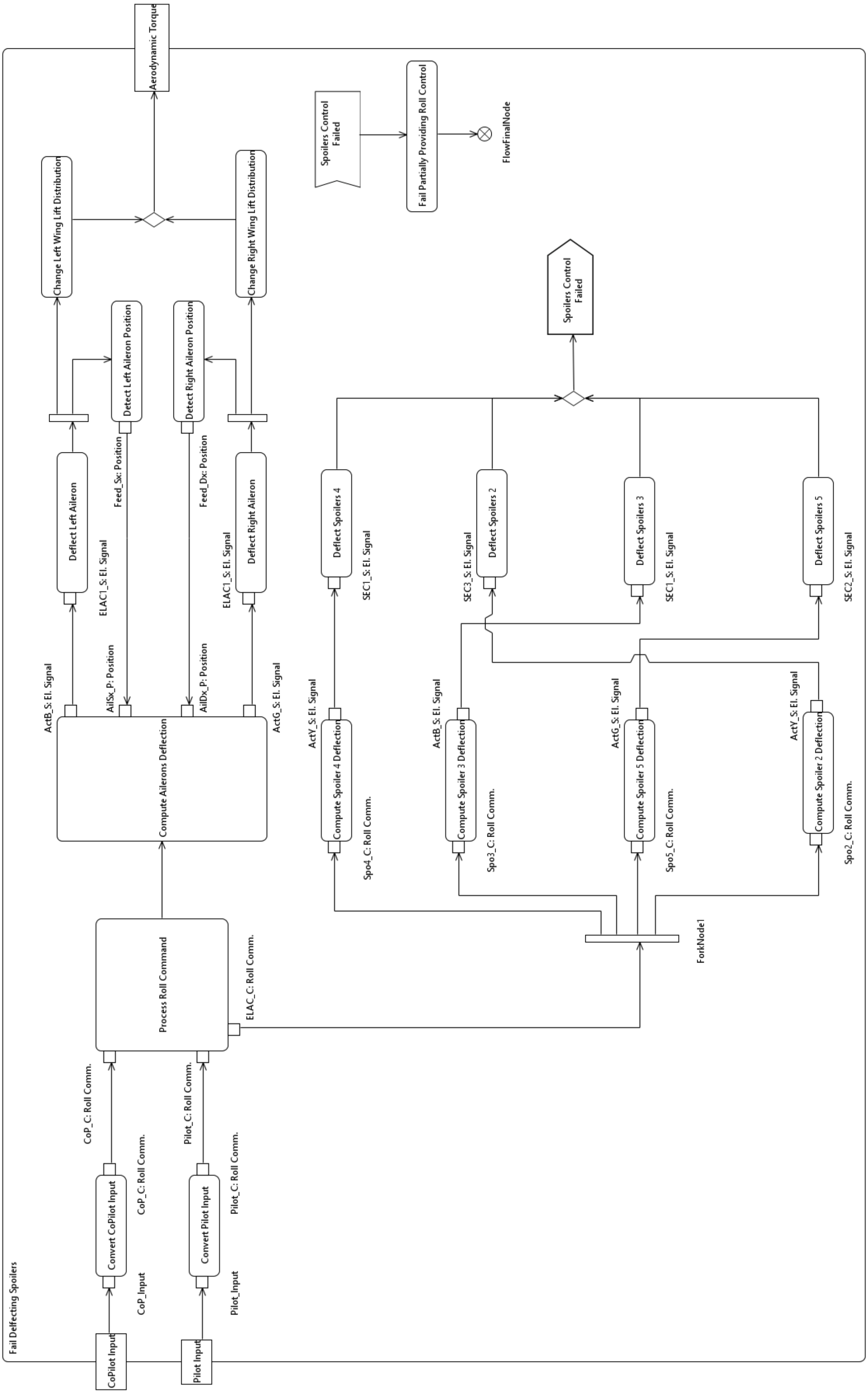


Fig. 75: FCS Model-Based FHA – Spoilers control failure

C.5. FCS– Model-based FTA

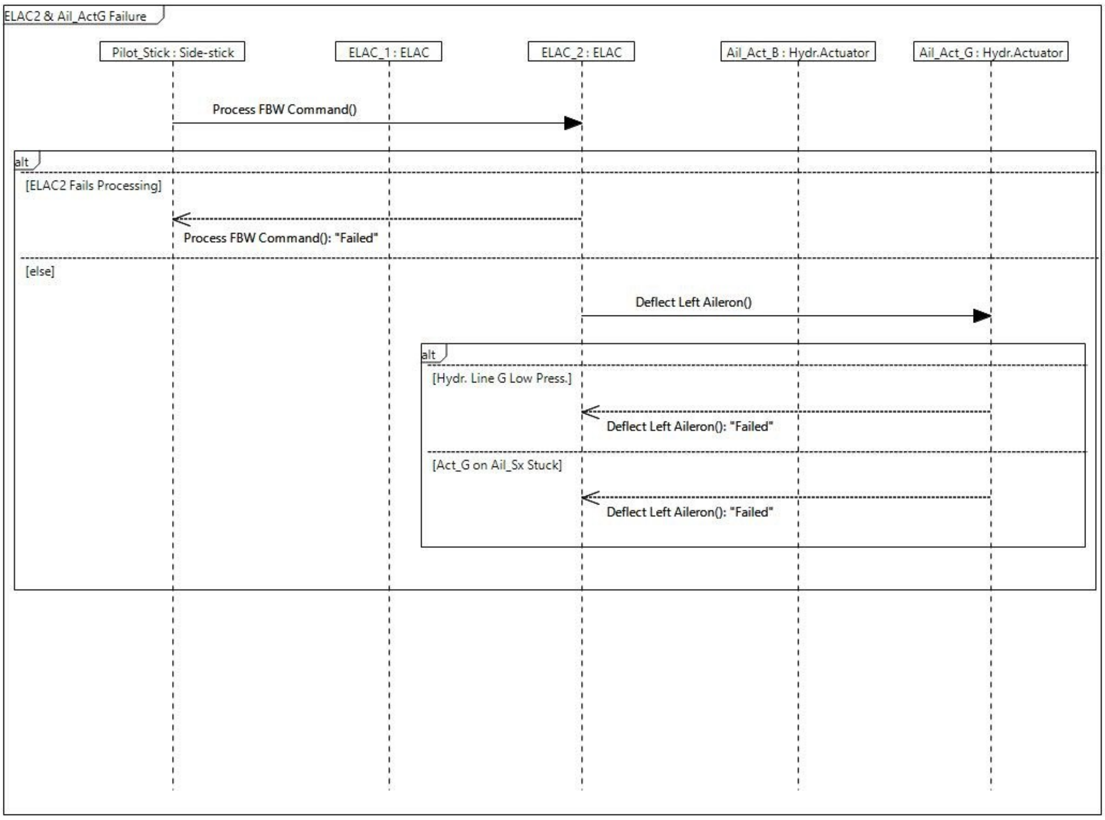


Fig. 76: Interaction use reference – ELAC 2 and its relative actuator on left aileron

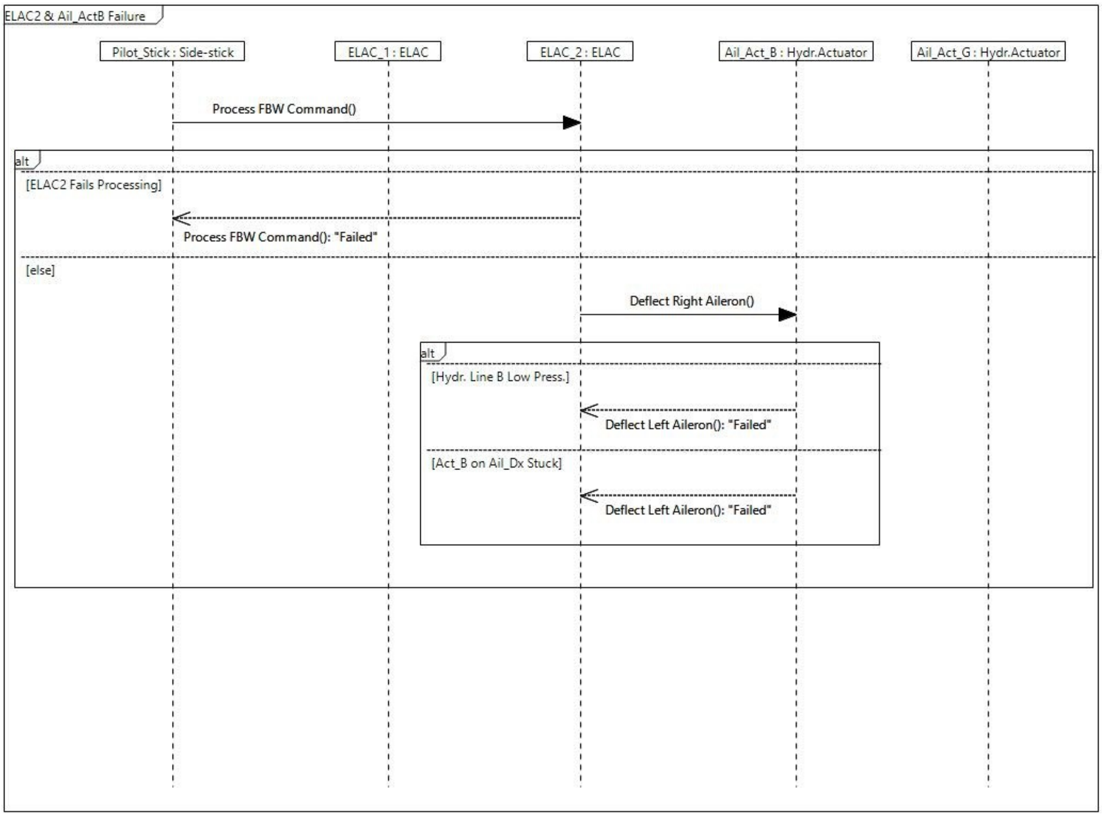


Fig. 77: Interaction use reference – ELAC 2 and its relative actuator on right aileron

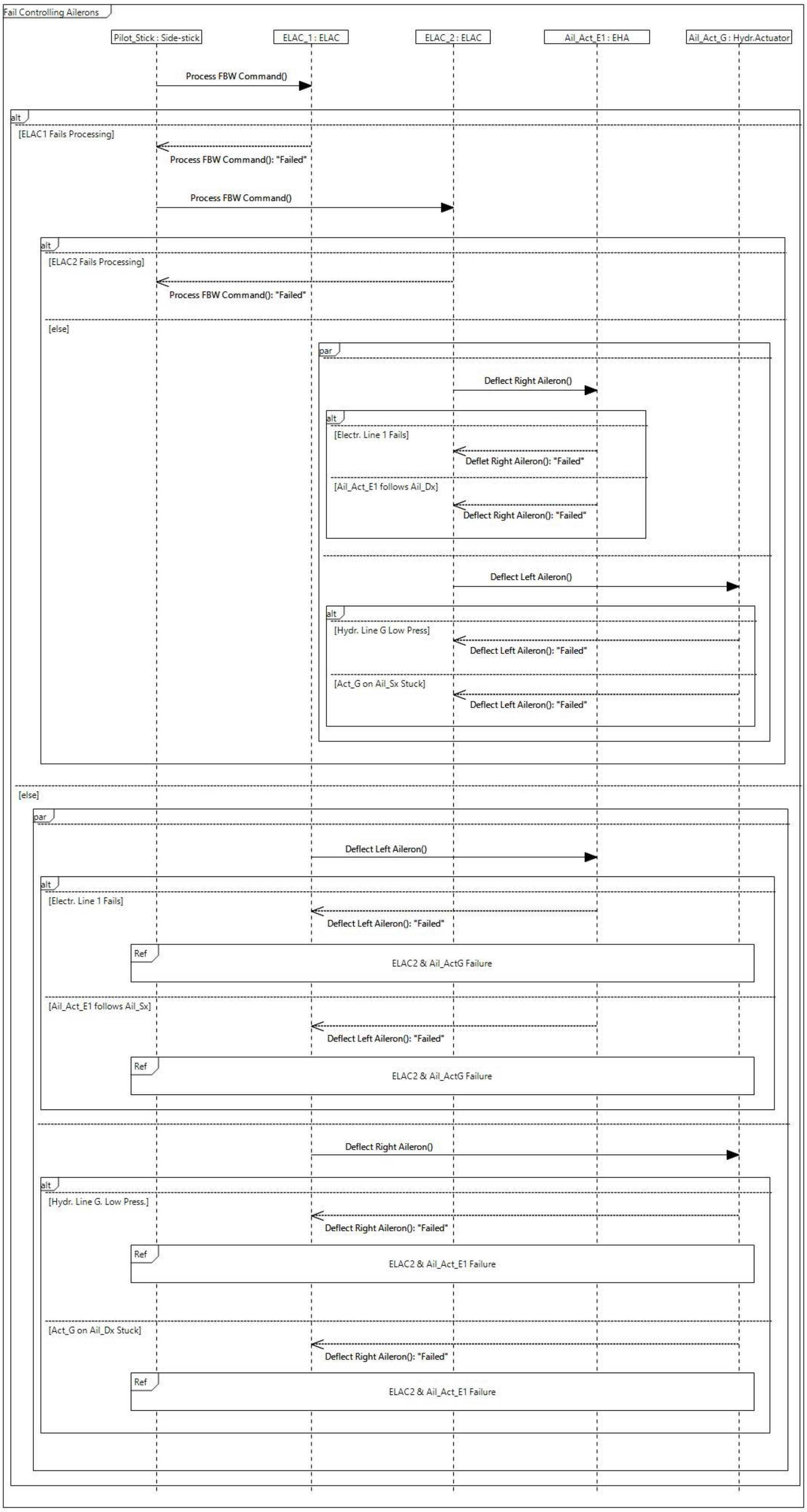

Fail Controlling Ailerons
Pilot_Stick : Side-stick
ELAC_1 : ELAC
ELAC_2 : ELAC
Ail_Act_E1 : EHA
Ail_Act_G : Hydr.Actuator
Process FBW Command()
alt
[ELAC1 Fails Processing]
Process FBW Command(): "Failed"
Process FBW Command()
alt
[ELAC2 Fails Processing]
Process FBW Command(): "Failed"
[else]
par
Deflect Right Aileron()
alt
[Electr. Line 1 Fails]
Deflet Right Aileron(): "Failed"
[Ail_Act_E1 follows Ail_Dx]
Deflect Right Aileron(): "Failed"
Deflect Left Aileron()
alt
[Hydr. Line G Low Press]
Deflect Left Aileron(): "Failed"
[Act_G on Ail_Sx Stuck]
Deflect Left Aileron(): "Failed"
[else]
par
Deflect Left Aileron()
alt
[Electr. Line 1 Fails]
Deflect Left Aileron(): "Failed"
Ref
ELAC2 & Ail_ActG Failure
[Ail_Act_E1 follows Ail_Sx]
Deflect Left Aileron(): "Failed"
Ref
ELAC2 & Ail_ActG Failure
Deflect Right Aileron()
alt
[Hydr. Line G. Low Press.]
Deflect Right Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E1 Failure
[Act_G on Ail_Dx Stuck]
Deflect Right Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E1 Failure

Fig. 78: Failed ailerons control on More Electric FCS – Sequence Diagram

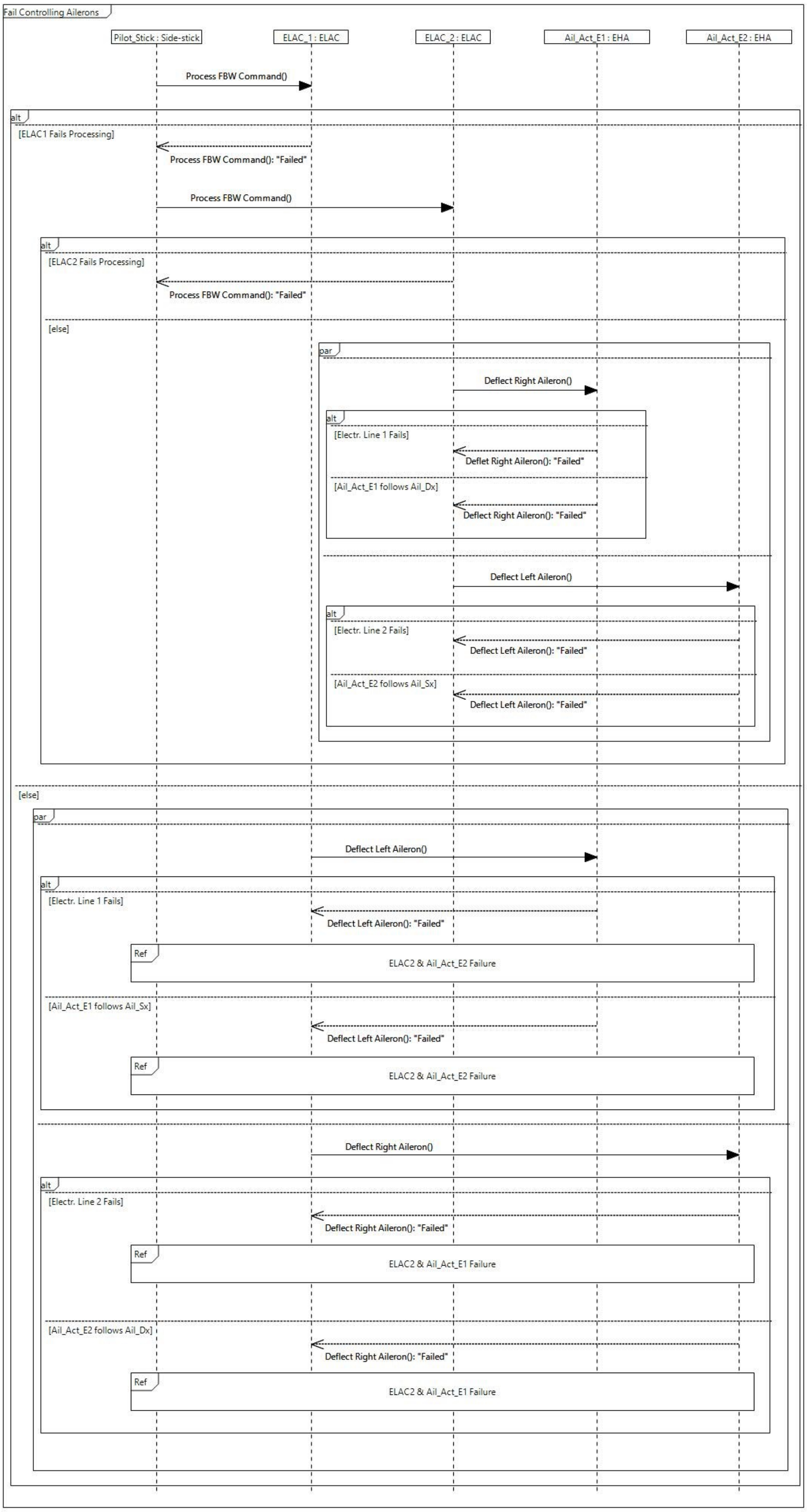

Fail Controlling Ailerons
Pilot_Stick : Side-stick
ELAC_1 : ELAC
ELAC_2 : ELAC
Ail_Act_E1 : EHA
Ail_Act_E2 : EHA
Process FBW Command()
alt
[ELAC1 Fails Processing]
Process FBW Command(): "Failed"
Process FBW Command()
alt
[ELAC2 Fails Processing]
Process FBW Command(): "Failed"
[else]
par
Deflect Right Aileron()
alt
[Electr. Line 1 Fails]
Deflet Right Aileron(): "Failed"
[Ail_Act_E1 follows Ail_Dx]
Deflect Right Aileron(): "Failed"
Deflect Left Aileron()
alt
[Electr. Line 2 Fails]
Deflect Left Aileron(): "Failed"
[Ail_Act_E2 follows Ail_Sx]
Deflect Left Aileron(): "Failed"
[else]
par
Deflect Left Aileron()
alt
[Electr. Line 1 Fails]
Deflect Left Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E2 Failure
[Ail_Act_E1 follows Ail_Sx]
Deflect Left Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E2 Failure
Deflect Right Aileron()
alt
[Electr. Line 2 Fails]
Deflect Right Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E1 Failure
[Ail_Act_E2 follows Ail_Dx]
Deflect Right Aileron(): "Failed"
Ref
ELAC2 & Ail_Act_E1 Failure

Fig. 79: Failed ailerons control on All Electric FCS – Sequence Diagram

C.6. FCS – Model-based FMEA

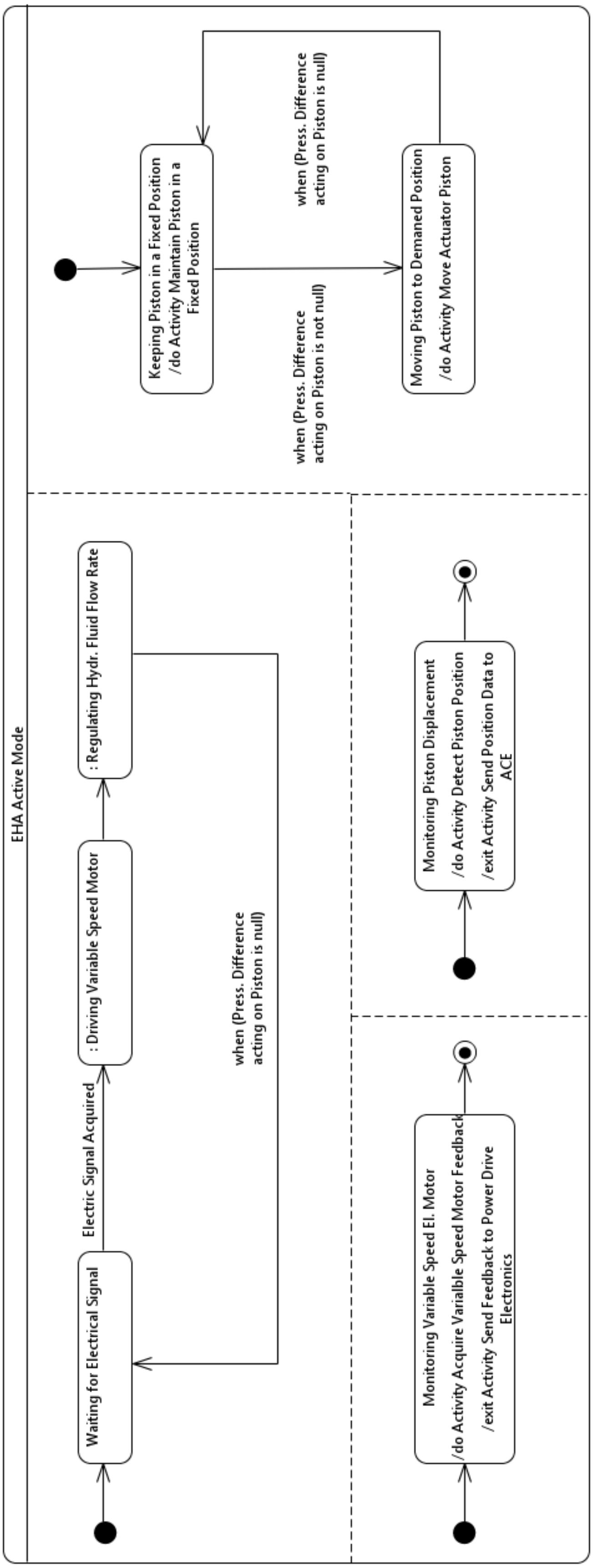

EHA Active Mode
Keeping Piston in a Fixed Position
/do Activity Maintain Piston in a Fixed Position
when (Press. Difference acting on Piston is not null)
Moving Piston to Demaned Position
/do Activity Move Actuator Piston
when (Press. Difference acting on Piston is null)
Waiting for Electrical Signal
Electric Signal Acquired
: Driving Variable Speed Motor
: Regulating Hydr. Fluid Flow Rate
when (Press. Difference acting on Piston is null)
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE
Monitoring Variable Speed El. Motor
/do Activity Acquire Varialble Speed Motor Feedback
/exit Activity Send Feedback to Power Drive Electronics

Fig. 80: EHA active mode – State Machine Diagram

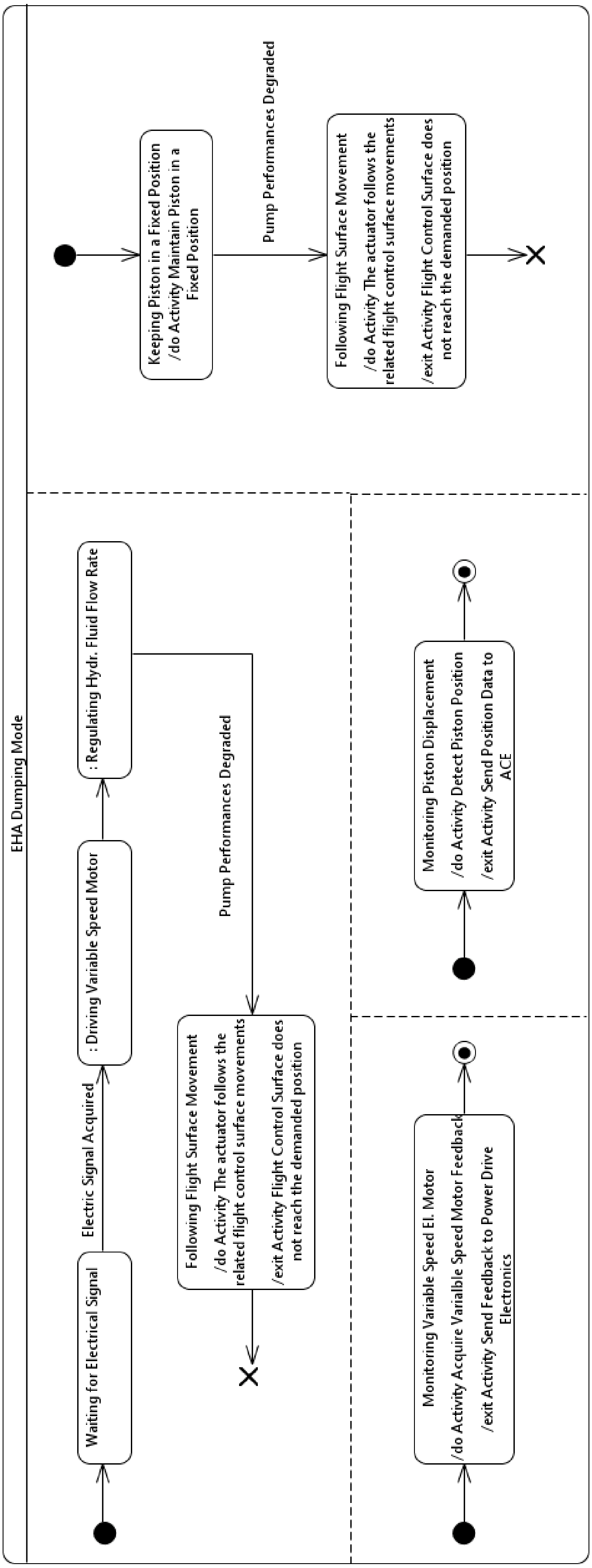

EHA Dumping Mode
Waiting for Electrical Signal
Electric Signal Acquired
: Driving Variable Speed Motor
: Regulating Hydr. Fluid Flow Rate
Pump Performances Degraded
Following Flight Surface Movement
/do Activity The actuator follows the related flight control surface movements
/exit Activity Flight Control Surface does not reach the demanded position
Keeping Piston in a Fixed Position
/do Activity Maintain Piston in a Fixed Position
Pump Performances Degraded
Following Flight Surface Movement
/do Activity The actuator follows the related flight control surface movements
/exit Activity Flight Control Surface does not reach the demanded position
Monitoring Variable Speed El. Motor
/do Activity Acquire Varialble Speed Motor Feedback
/exit Activity Send Feedback to Power Drive Electronics
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE

Fig. 81: EHA dumping mode – State Machine Diagram

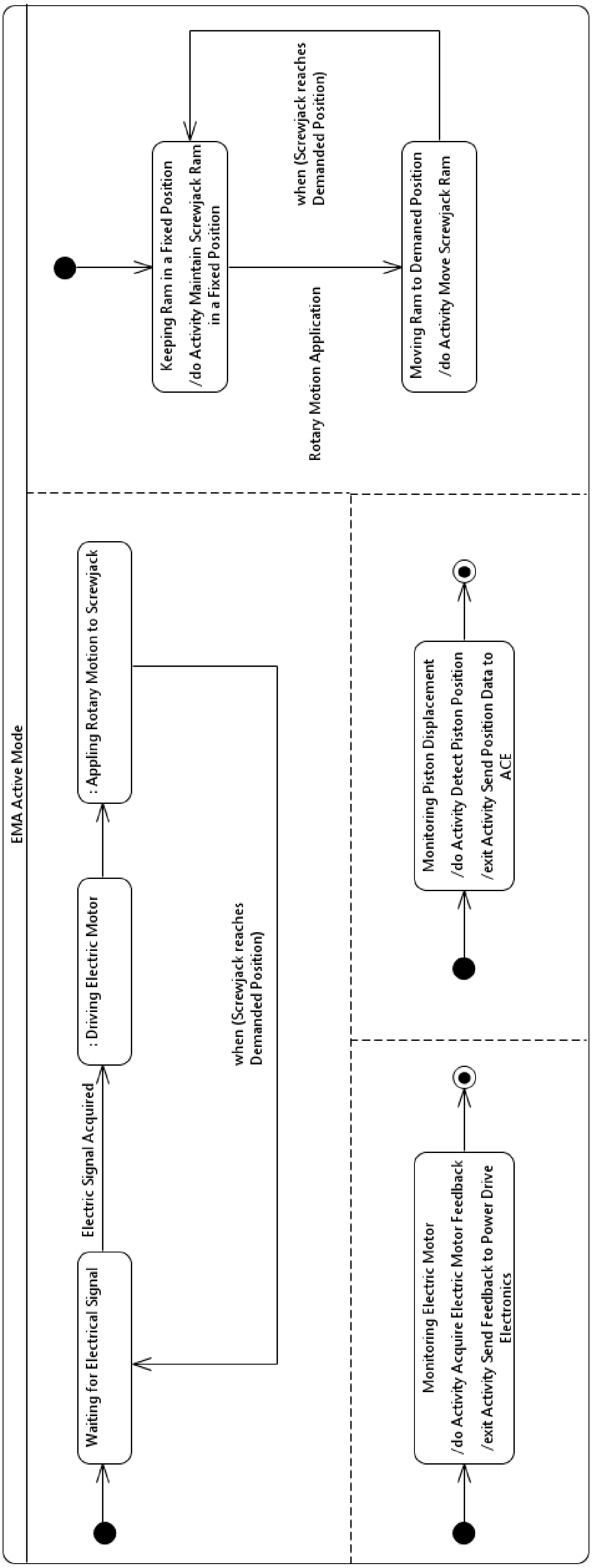

EMA Active Mode
Waiting for Electrical Signal
Electric Signal Acquired
: Driving Electric Motor
: Appling Rotary Motion to Screwjack
when (Screwjack reaches Demanded Position)
Monitoring Electric Motor
/do Activity Acquire Electric Motor Feedback
/exit Activity Send Feedback to Power Drive Electronics
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE
Keeping Ram in a Fixed Position
/do Activity Maintain Screwjack Ram in a Fixed Position
Rotary Motion Application
when (Screwjack reaches Demanded Position)
Moving Ram to Demaned Position
/do Activity Move Screwjack Ram

Fig. 82: EMA active mode – State Machine Diagram

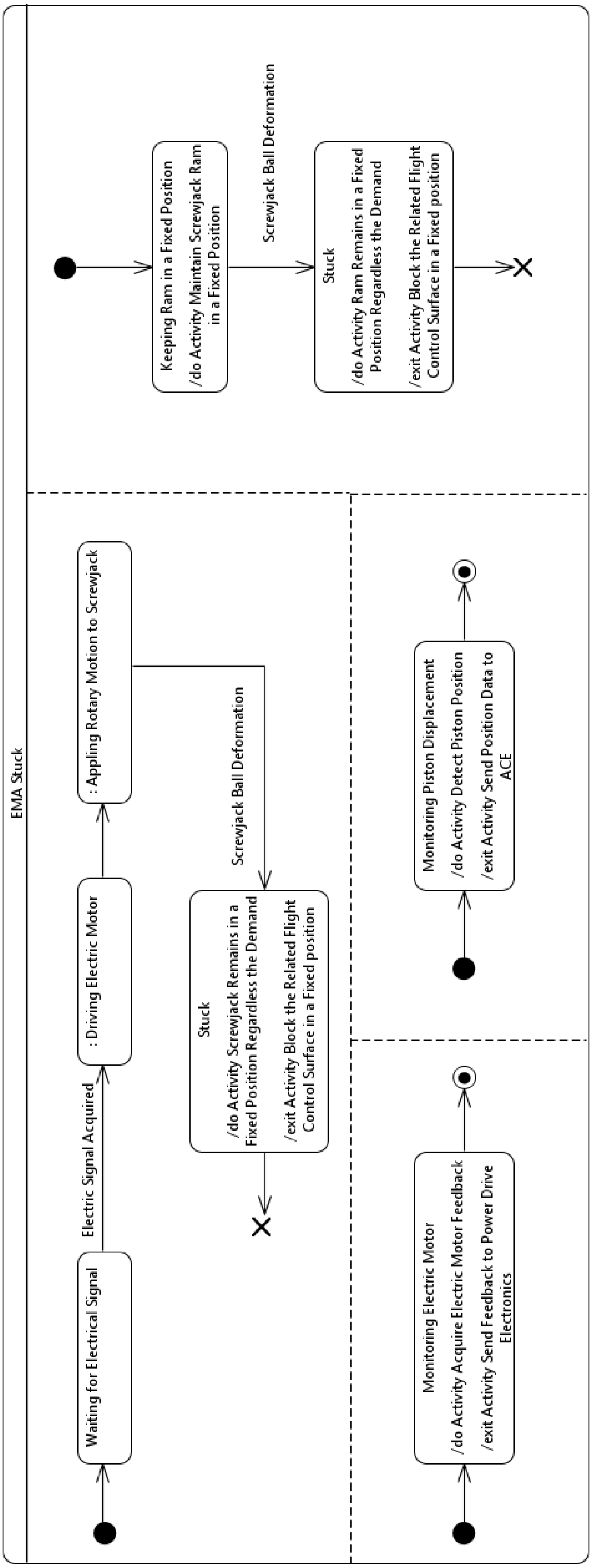

EMA Stuck
Waiting for Electrical Signal
Electric Signal Acquired
: Driving Electric Motor
: Appling Rotary Motion to Screwjack
Screwjack Ball Deformation
Stuck
/do Activity Screwjack Remains in a Fixed Position Regardless the Demand
/exit Activity Block the Related Flight Control Surface in a Fixed position
Monitoring Electric Motor
/do Activity Acquire Electric Motor Feedback
/exit Activity Send Feedback to Power Drive Electronics
Monitoring Piston Displacement
/do Activity Detect Piston Position
/exit Activity Send Position Data to ACE
Keeping Ram in a Fixed Position
/do Activity Maintain Screwjack Ram in a Fixed Position
Screwjack Ball Deformation
Stuck
/do Activity Ram Remains in a Fixed Position Regardless the Demand
/exit Activity Block the Related Flight Control Surface in a Fixed position

Fig. 83: Stuck EMA – State Machine Diagram

C.7. FCS – Model-based RBD

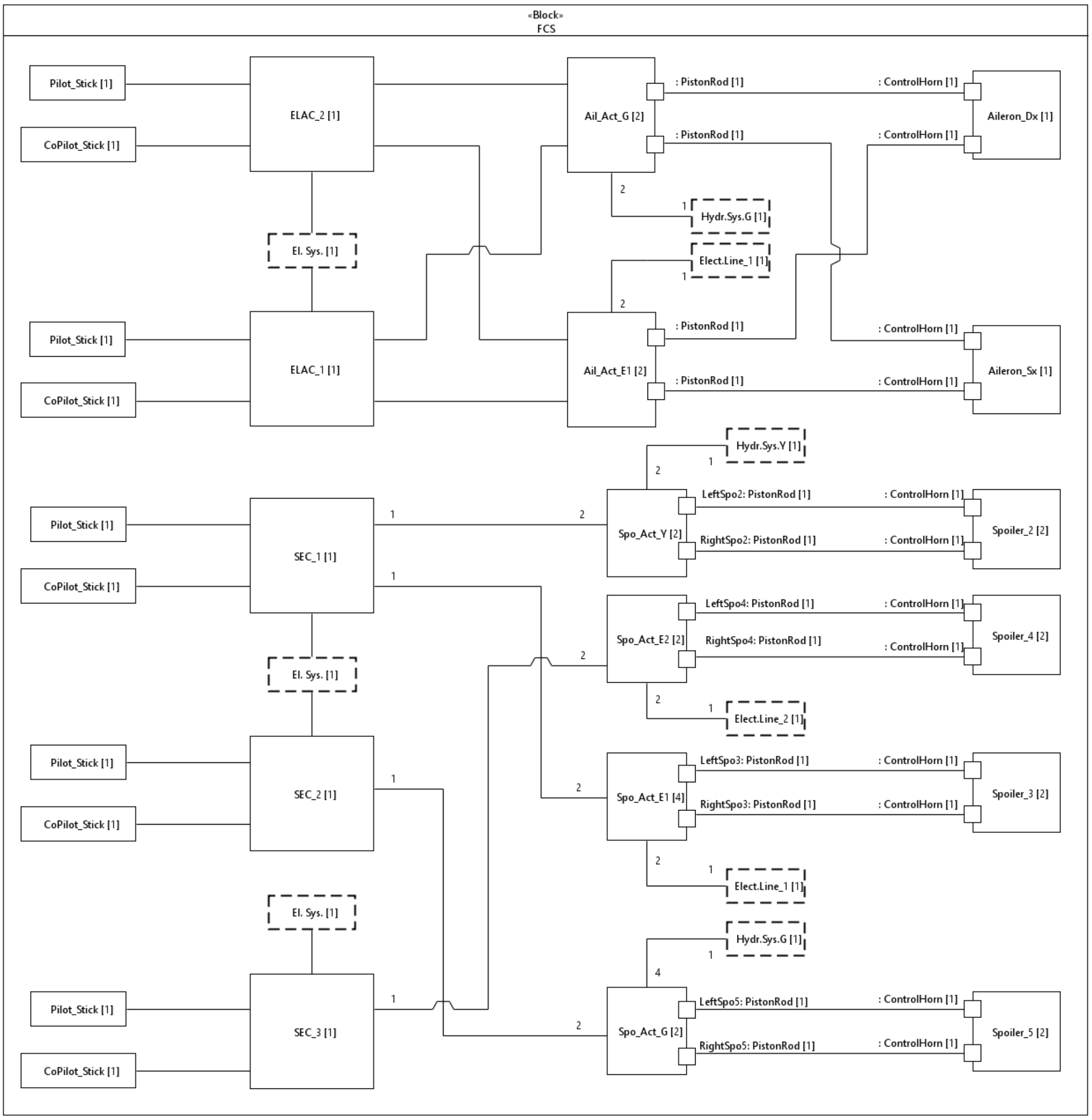


Fig. 84: FCS More Electric Architecture – IBD Simplified focused on Roll Control

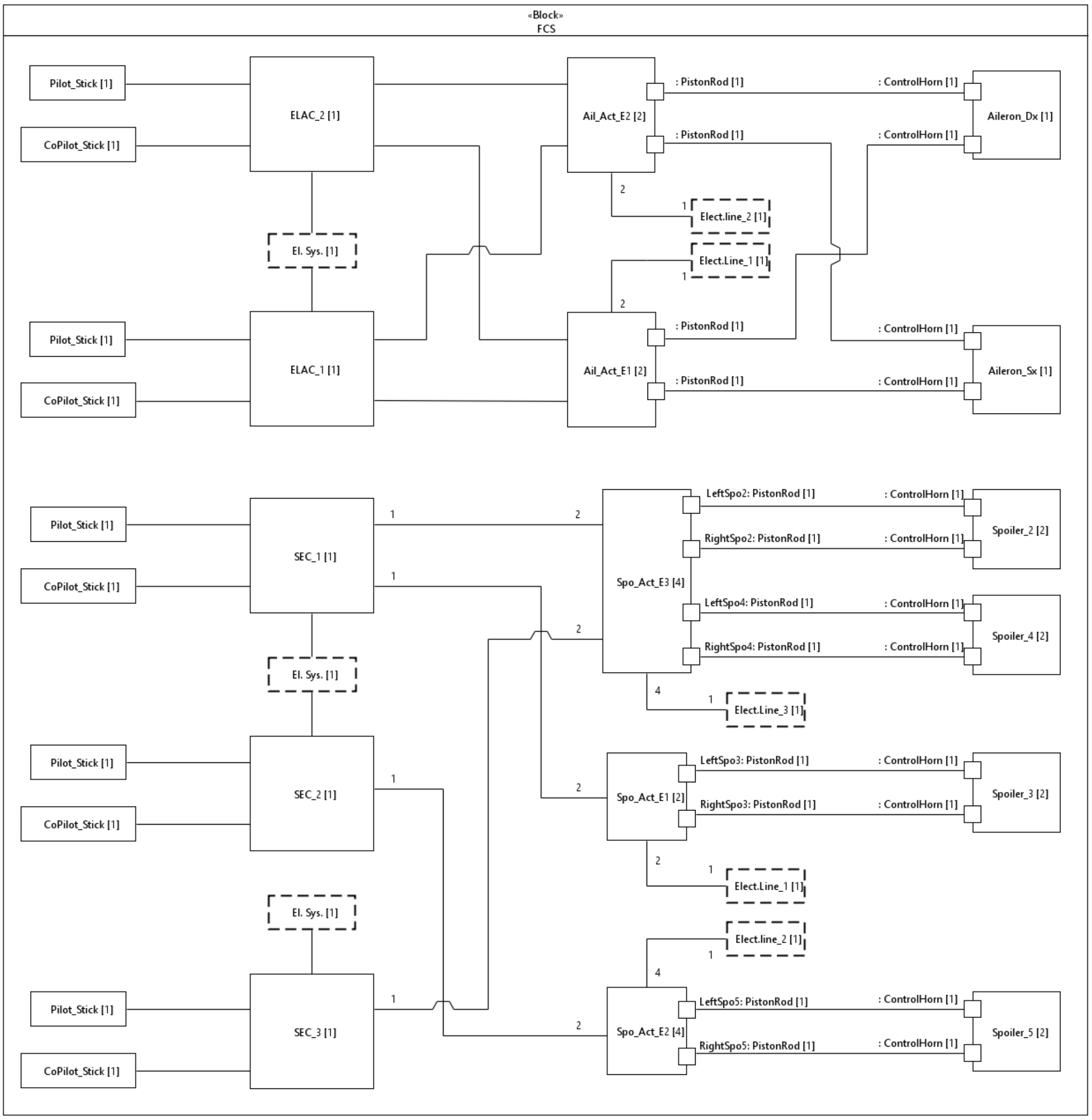


Fig. 85: FCS All Electric Architecture – IBD Simplified focused on Roll Control