33RD CONGRESS
OF THE INTERNATIONAL COUNCIL
OF THE AERONAUTICAL SCIENCES
STOCKHOLM, SWEDEN, 4-9 SEPTEMBER, 2022

ICAS 2022 SWEDEN

METHODOLOGY FOR THE AUTOMATED PRELIMINARY CERTIFICATION OF ON-BOARD SYSTEMS ARCHITECTURES THROUGH REQUIREMENTS ANALYSIS

Carlos Cabaleiro[1&2], Marco Fioriti[1] & Luca Boggero[2]

[1] Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10129 Turin, Italy
[2]German Aerospace Center (DLR), Hein-Saß-Weg 22, 21129 Hamburg, Germany

Abstract

Aircraft on-board systems architectures are defined by the subsystems and the connections among them. The requisites for these connections are not directly established in the certification specifications but they are indirectly derived from other requirements. In addition, generally only a small number of architectures taken from previous studies are considered when performing on-board systems design. This makes it difficult to generate certifiable connections when assessing an extensive number of architectures. Considering certification aspects during early design stages can be used as a filter to save computational time by calculating only potentially certifiable architectures. The aim of this paper is to develop a methodology to automatically assess certification requirements of on-board systems architectures that come from the certification specifications. One part of the methodology consists of a list of requirements to be considered to define the connections among on-board systems during architecture design in order to find safe and certifiable solutions. The other part is focused on the automation of the reliability block diagram technique. This is needed in order to verify safety assessment requirements which have a high influence on the architectures and connections. The advantages of this study are mainly the capability to assess multiple architectures and to verify certification requirements during early design stages. A full automation for this process was achieved and showed through an example test case. An aeronautical application case is also shown. This analysis could also be implemented for the study of innovative on-board systems architectures.

Keywords: On-board systems, certification, reliability block diagram, systems architectures

Nomenclature

AMC	Acceptable Means of Compliance
AEA	All-Electric Aircraft
APU	Auxiliary Power Unit
CS	Certification Specification
$EBHA$	Electric Backup Hydraulic Actuator
EHA	Electro-Hydrostatic Actuator
EMA	Electromechanical Actuator
FH	Flight Hour
λ	Failure Rate
MDO	Multi-Disciplinary Design Optimization
MEA	More-Electric Aircraft
OBS	On-Board Systems
PTU	Power Transfer Unit
R	Reliability
$RAMS$	Reliability, Maintainability, Availability and Safety
RAT	Ram Air Turbine
RBD	Reliability Block Diagram
$TLARs$	Top Level Aircraft Requirements

1. Introduction & State of the Art

Aircraft on-board systems guarantee that the essential functions to be carried out during the mission are achieved (e.g. flight control, communications). Their correct functioning is essential to fulfill a safe flight and therefore are considered as critical systems. On-board systems include the power generation systems (e.g. engines and auxiliary power units), the power distribution systems (e.g. hydraulic and electrical systems) and the power consuming systems (e.g. flight control system and environmental control system). Connections among systems are needed in order to lead the power from the generation to the users through the distribution systems. Since the on-board systems are critical, redundancies are needed. Redundancies are applied at a component level (e.g. two actuators per control surface) and at a connection level (e.g. each actuator is connected to two different power sources).

Certification authorities set a minimum value for the reliability of a system which has to be guaranteed through the usage of redundancies. The architecture of the on-board systems is influenced by these redundancies and connections. Certification authorities give guidelines and requisites for the correct functioning of the systems but requirements about the connections are not explicitly mentioned under a specific section. Some of these specifications can be derived from other systems requirements or found in appendices. A detailed study is key to ensure that all these conditions are met.

Usually just a small number or previously-defined architectures are selected while analyzing on-board systems [1, 2]. A selection among these is then performed depending on which one grants more optimum results given the design objectives. On these cases, the architectures can be individually assessed, making sure all the devices meet the redundancy requirements set by the designers. However, future analysis should allow the exploration of multiple and numerous architectures. This will allow a better exploration of the solutions potentially leading to new more-optimal and innovative solutions for the on-board systems architectures that are not necessarily based on previous studies. The huge number of combinations while defining on-board systems connections, and hence architectures, makes it non-viable to evaluate all the possible solutions and combinations. Optimization algorithms are usually used in order to converge to the potential optimal solutions without needing to assess all of the possible design variables. A proper pre-filtering of these architectures is needed in order to save computational time and guarantee that only suitable architectures are evaluated. This implies that some architectures should be discarded before evaluation since they do not meet the requirements and specifications of connectivity or redundancies. Furthermore, multiple architecture assessment requires models automation. This means that the automation of tools, as well as the links among tools, is needed in order to evaluate multiple and numerous architectures. Some semi-automated methods for on-board systems architectures generation were proposed in the past [3]. These models have been enhanced reaching a full automation that also follows the rules of model-based modeling [4]. However, an automated evaluation of multiple automatically-generated architectures has not been performed, leaving a gap between architectures generation and architectures evaluation [5, 1].

Another important aspect for future studies is the capability to reach a component level even during early design stages. A proper architecture definition should describe the system, subsystem and components and reach each of them individually. This is important for RAMS assessment (Reliability, Availability, Maintenance and Safety) for instance. In this kind of studies one typical study regards the evaluation of individual component reliability. This allows to find those components with higher failure rating and take measures (e.g. changing it for one with higher reliability or increasing the redundancy). Old methods in RAMS analysis were based on top down approaches [6]. This implies that component level is not fully reached since the analysis considers the subsystem as a whole and not as a compound of individuals. Some improvement to these models has been done being now able to perform the safety assessment of complex architectures. However, a direct link with architecture generation is still missing [7]. Some authors expressed the need to link systems sizing together with systems reliability [1].

Some RAMS methods focused on safety analysis can be found in technical standards [8, 9]. For instance, FMECA (Failure Modes, Effects and Criticality Analysis) identifies possible occurring failures for each component of the system. FTA (Fault Tree Analysis) represents the architecture of the system and its failure mechanisms. FHA (Functional Hazard Assessment) identifies failure modes, severity and risk associated for each subsystem. The results from different analysis converge into a final safety assessment, applied to the whole system [10]. The literature [1] and the technical standards [9] emphasize the need of evaluating systems performance and RAMS analysis since the early stages of design. However, these methods are not easily automatable and need human supervision, so they are taken out of the automated part of the methodology. Reliability block diagram (RBD) is a tool that allows automation and that is used to estimate safety values of the system. It was chosen for this study for this reason.

The motivation of this analysis was augmented during the enhancement of a on-board systems sizing tool (ASTRID [11]), developed by Politecnico di Torino, within the framework of the AGILE 4.0 project [12]. In this project, four on-board systems architectures are assessed for some specific application cases (conventional aircraft, two versions of more-electric ones and an all-electric version). These four options come from experience and previous knowledge and the connections among subsystems are predefined. However, a multiple architectures evaluation is needed in order to potentially find new and innovative solutions in the future. With this approach the architectures cannot be individually checked by the engineers. Hence there is a need to develop an automated generation and assessment of certifiable on board systems architectures.

The objective of this work is to define a methodology that allows to automatically check certification requirements of on-board systems architectures connections. This methodology can be used to evaluate the impact of new innovative technologies. And it is the first step to achieve a virtual certification process during on-board systems design. The benefits of this methodology rely on two aspects. First one is the automation, which allows the evaluation of a huge number of design variables. The other aspect concerns the fact that certification aspects are usually not considered during early stages of the design and the results could potentially lead to new more optimal solutions.

Chapter 1 of this paper was an introduction to the problem to be assessed. Chapter 2 gives a quick overview on on-board systems architectures, explaining the different systems that are present on aircraft and giving an example of conventional and newer architectures. Chapter 3 describes the developed methodology. An aeronautical application case is shown in chapter 4. Chapter 5 summarizes the conclusions and remarks.

2. On-Board Systems Architectures

Aircraft on-board systems can be separated depending on their functionality [13]. Systems that perform a specific objective needed to accomplish the mission (e.g. remove ice from the wings or transfer the fuel to the engines) are called power consuming systems, these systems need power in order to work properly. Power is generated by the power generation systems and then transformed and transferred by the power distribution systems [2, 1]. The connections between these subsystems define the on-board systems architecture. A proper architecture definition contemplates all the components from the power source (e.g. APU) to the specific devices (e.g. actuators, compressors). These differentiation is now explained more in depth focusing on aircraft systems.

2.1 Power Generation Systems

Power generation systems are the source of all the power needed by the users. They can also be called prime movers. Mainly the engines are used for this purpose although other components can be used as redundancies (e.g. ram air turbine (RAT), auxiliary power unit (APU)), alternative sources (e.g. fuel cells) or as a power source during certain mission phases (e.g. batteries).

2.2 Power Distribution Systems

The main function of these systems is to distribute and transform the energy from the power generation systems and deliver it to the power consuming systems. Three subsystems are generally used for this purpose:

- Electrical System: distributes the electrical power (typically given by the engine generators) and delivers it at the correspondent voltage to each of the users.

- Hydraulic System: transforms the generated power into hydraulic power by the usage of hydraulic pumps. Then distributes it through pipes to the users.

- Pneumatic System: takes hot and compressed air from a certain source (e.g. engine bleeding or external compressors) and delivers it to the users (e.g. ice protection system).

2.3 Power Consuming Systems

These systems need the energy given by the power distribution systems to perform a specific task needed for the fulfillment of the mission. They can generally be listed as it follows:

1. Flight Control System
2. Landing Gear (Retraction, Steering & Braking)
3. Environmental Control System
4. Ice Protection System
5. Avionics
6. Fuel System
7. Others (e.g. lights, water waste)

2.4 Architecture Example: More Electric Aircraft

Figure 1 shows the comparison between a conventional on-board systems architecture and an all-electric one. It can be seen how the connection among systems changes. This image represents a simple draft of the whole architecture definition. It can be noticed that the current tendency is to electrify these systems as much as possible due to the potential advantages in terms of fuel burn and RAMS [14]. This enhances the idea of exploring a bigger amount of new architectures to assess the impact they might have. Hence, figure 1 shows previously defined architectures, not automatically generated ones.

The whole connection draft should show into detail the connection between each component to each lane to each power source. An example is now given to the reader. A classic flight control system (e.g. A320) could be compound by seventeen control surfaces, each surface is moved by one, two or even three actuators to meet the safety requirements. Each actuator might have a different technology (e.g. hydraulic, electro-hydrostatic (EHA), electromechanical (EMA), electric backup hydraulic actuator (EBHA)) and is connected to a different hydraulic or electric lane. Each lane is connected to different power sources. Redundancies are needed in other to guarantee a safe functioning of the system, this means that each surface could need a link to different lanes and/or power sources. The number of connections is huge and needs to be defined in other to fully define the architecture.

3. Methodology

As stated in the previous chapters, the objective is to develop a methodology to automatically check certification requirements of on-board systems architectures connections. This can also be used as a filter when the number of architectures to evaluate is huge. In addition, certification aspects are usually not taken into account during early design stages and this could lead to new potential more optimal solutions. The methodology should also:

- Allow multiple architectures evaluation
- Be easily automated
- Consent changes in the requirements (e.g. change from CS25 to CS23)
- Reach a component level
- Support the assessment of innovative technologies

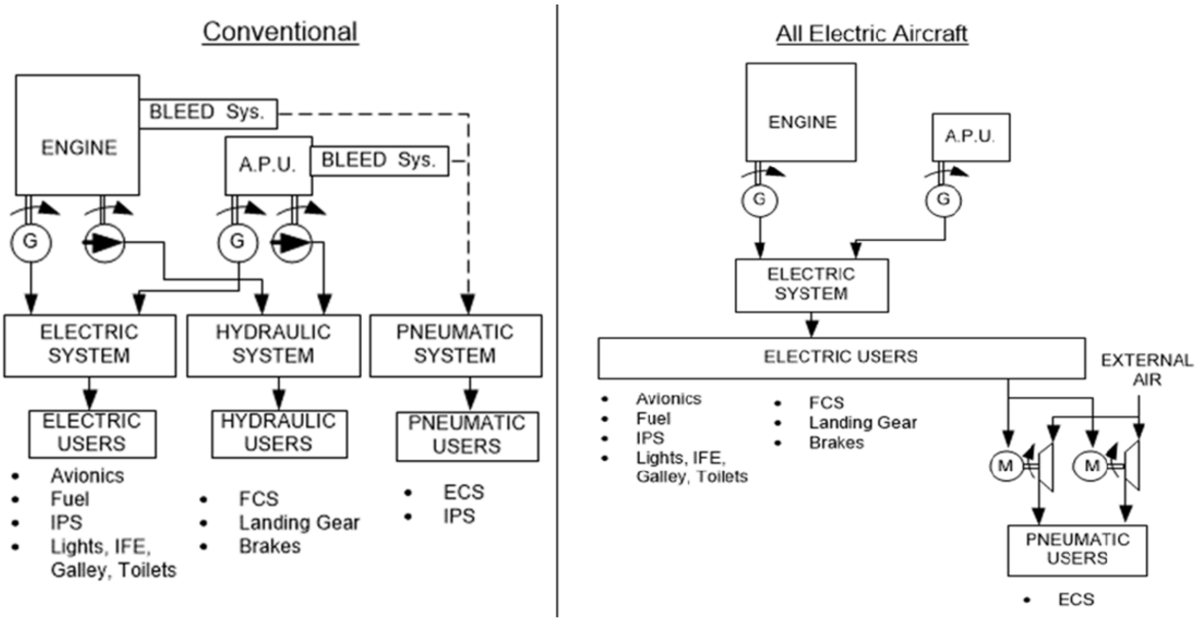

Figure 1 – Comparison between a conventional and an all-electric architecture.

The methodology is divided in two parts. The first one collects and defines connectivity requirements to be used as a certification check. It explains how to filter the generated architectures based on qualitative and quantitative certification requirements. The second one focuses on the automation of reliability block diagrams that allows to assess the global system's reliability value and compare it with the minimum established by the certification authorities. These two parts are represented in figure 2 as blocks with continuous lines. They can be modeled as two filters located between the OBS architectures generation and the OBS sizing, which are existing tools and are represented as blocks with discontinuous lines. Figure 2 explains the general framework of the methodology. Previous studies have also attempted similar methodologies structures [15].

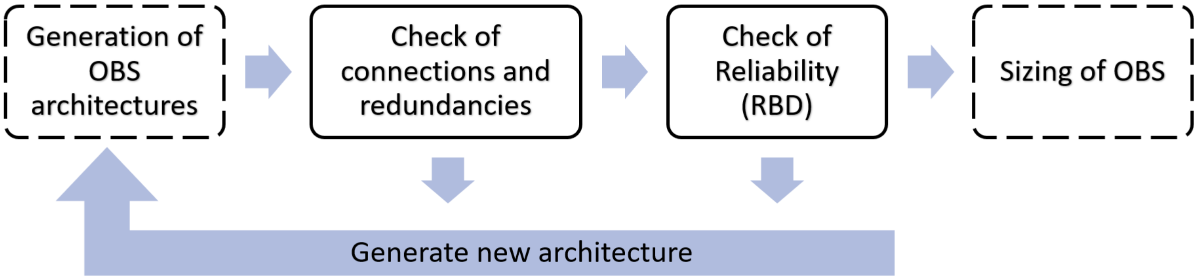
Generation of OBS architectures
Check of connections and redundancies
Check of Reliability (RBD)
Sizing of OBS
Generate new architecture

Figure 2 – Methodology schema: previously existing tools are encapsulated by discontinuous lines, new added ones by continuous lines.

Figure 2 is now explained more in detail.

- Generation of OBS architectures: The first part is the automatic generation of on-board systems architectures. This depends on the specific problem to analyse and the scope of the study. If only a subsystem is to be assessed, the generation must be concordant and originate possible architecture combinations. This problem has already been solved in literature [3, 4, 1]. Usually this is approached inside a optimization process through the use of genetic algorithms [16].
- Check of connections and redundancies: This first filter checks whether or not the generated architectures fulfill some specific requirements given by the certification authorities. It is explained more in depth in section 3.1.

- Check of Reliability: This second filter makes sure that the given architecture meets the minimum reliability values specified by the certification. It is explained more in depth in section 3.2.
- Sizing of OBS: This block represents a classic OBS sizing tool (e.g. ASTRID [11]). This tool now receives a partially certified architecture that meets the certification requirements. The aircraft TLARS are also needed. The results from this analysis are usually integrated inside a bigger MDO workflow in which more disciplines and domains are analysed (e.g. aircraft performance, mission analysis).

The two mentioned filters are now explained in the following subsections and they represent the proposal to fill the gap in the state of the art.

3.1 Check of connections and redundancies: First filter

This part of the methodology consists of the extraction of different rules, requisites and requirements from the certification specifications (e.g. CS25 [17], CS23 [18]). All of them are merged and summarized on a list of requirements to be used while defining certifiable on-board systems architectures. As said before, this will allow to filter the on-board systems architectures through a first certification check to verify that the generated architectures meet the minimum requirements set by the certification authorities. If the conditions specified inside this filter are not met, the following blocks are not run and a new architecture is generated. This prevents the following more-time-consuming tools to be run with a non-certifiable architecture.

This filter must be done in such way that is easily linkable with the architectures generation tool. Both blocks (i.e. "Generation of OBS architectures" and "Check of connections and redundancies") could actually be merged into one if wanted. However separating them brings some advantages. For instance, changing from CS-25 to CS-23 can be done by only changing the set of rules stored inside of this diagram block, without needing to change the architectures generation tool. There is not only one way of making this tool and it depends on the scope and particular analysis that is being performed. A model that allows allows automation is suggested. It should also be easily integrated in a tool and allow switchability from one set of rules to another. The important part is to understand and translate the adequate requirements into rules and store them inside an automated code. These requirements are explained in depth in the following paragraphs and specifically under subsection 3.1.1.

Before listing the certification requirements it is noteworthy to say that this filter also admits rules other than the ones explicitly extracted from the certification. These rules might come from experience, previous analysis or literature. An example of this was made by I. Chakraborty [1] and it is now shown as an example to the reader:

- “Control surfaces such as ailerons and elevators, which are flight-critical, are provided with two actuators per panel… Each actuator is supplied by a single power system. Then:” [1]
  (a) “If the aileron group and/or the elevator group is powered by the same type of power (i.e., either hydraulic or electric but not both), then three such power systems are required” [1]
  (b) “If the aileron group and/or the elevator group is powered by both types of power (i.e., both hydraulic and electric), then two power systems of each type are required” [1]

This example shows a connection requirement for ailerons and elevator that is not explicit in the certification but extrapolated form good practices seen in literature. Both concepts (i.e. good practices and certification) can be merged into the same analysis. Other more current studies are following this line of research [19].

3.1.1 Set of OBS architectures requirements extracted from certification

A list of the main requirements regarding on-board systems connections certification is listed in this subsection. The main focus of this study is on civil aircraft, for this reason all the rules are extracted from the CS-25: Large Aeroplanes [17].

In general terms, the most important rule is related to catastrophic failures. During all the document the statement "catastrophic failure conditions must be extremely improbable" is found for several conditions. In particular, for on-board systems one can refer to CS 25.1309: Equipment, systems and installations. The most relevant information under this paragraph related to this study is the following:

- CS 25.1309(b): "The aeroplane systems and associated components, considered separately and in relation to other systems, must be designed so that:
  - (1) Any catastrophic failure condition
    - (i) is extremely improbable; and
    - (ii) does not result from a single failure"

The first condition "CS 25.1309(b)(1)(i): any catastrophic failure condition is extremely improbable" needs to be quantitatively defined. The instructions for this are found in CS-25 AMC-Subpart F, Chapter 7: failure condition classification and probability terms. More precisely under AMC 25.1309 System Design and Analysis.

- AMC 25.1309(7.c): Quantitative Probability Terms:
  - (1) Probability Ranges
    - (iv) "Extremely improbable failure conditions are those having an average probability per flight hour of the order of 1×10^{-9} or less."

Summarizing, two rules are extracted form the previous information:
First requirement: one single failure (i.e. the single failure of a single component) must never lead to the failure of the whole subsystem (extracted from **CS 25.1309(b)(1)(ii)**). This means that redundancies (e.g. components in parallel) must always be used in order to fulfill this requirement. This requirement is emphasized in further paragraphs. For instance CS 25.671(c)(1) says that a single failure cannot risk the continuous safe flight, referring to the flight control system. CS 25.745(c) implies the same condition but for the nose-wheel (i.e. steering function). As a general rule, it can be expressed that a single failure cannot compromise the functioning of any subsystem.
Second requirement: Catastrophic failure conditions (e.g. loss of one subsystem functionality) must occur with an average probability lower than 1×10^{-9} per flight hour (extracted from **CS 25.1309(b)(1)(i)** and **AMC 25.1309(7.c)(1)(iv)**). This condition is more difficult to assess and it is considered individually in the second filter in section 3.2 of this paper.
Another sizing condition can be found in CS 25.671(d). This requirement expresses extra safety requirements for the flight control system and landing gear. The following statement is written:

- The aeroplane must be designed so that, if all engines fail at any time of the flight:
  - (1) it is controllable in flight;
  - (2) an approach can be made;
  - (3) a flare to a landing, and a flare to a ditching can be achieved; and
  - (4) during the ground phase, the aeroplane can be stopped.

Also CS 25.729(c) gives more restrictions to the landing gear design:

- Emergency operation. There must be an emergency means for extending the landing gear in the event of –
  - (1) any reasonably probable failure in the normal extension and retraction systems; or
  - (2) the failure of any single source of hydraulic, electric, or equivalent energy supply.

This leads to two more subsystem-specific requirements.

Third requirement: There must be at least one back-up system to ensure the correct functioning of the flight control system and landing gear in case of loss of all engines (extracted from **CS 25.671(d)**. This results in the necessity of having at least an auxiliary power unit (APU), or a ram air turbine (RAM) or another non-engine-dependant redundant power source installed in the aircraft.

Fourth requirement: The function of the landing gear extension cannot be connected to a single line (electrical or hydraulic). Not even if this line has enough redundancies, and this function cannot be dependant of a single actuator eather (extracted from **CS 25.729(c)(2)**. This implies that a minimum of two lines need to be connected to the landing gear extension. This can be done with two hydraulic lines, or two electrical lines or one of each type. If this function is fulfilled by a single actuator there must be another means of extending the landing gear in the event of failure.

These requirements are specific for CS-25. An equivalent analysis should be done for other certification specifications (e.g. CS-23). Additionally, no further conditions were found regarding on-board systems connections. Other requirements can be found but they generally express more specific conditions. For example, CS 25.831(a) provides a minimum airflow value that has to be guaranteed by the environmental control system. This type of requirements must be assessed after the on-board systems design, once all the systems are properly sized and calculated. They are left out of the scope of this analysis since they do not directly affect the OBS connections.

3.2 Check of reliability: Second filter (Reliability Block Diagram automation)

This second filter is based on the second requirement previously defined in section 3.1.1. It states that the probability of a catastrophic failure has to be lower than 1×10^{-9} per FH. The reliability block diagram (RBD) technique is used in order to estimate this reliability value of a subsystem. Hence this filter estimates the minimum reliability that one or more subsystems reach and compares it with the value specified by the certification authorities. If the minimum value is not fulfilled the architecture is discarded and a new one is generated, preventing the on-board systems sizing tool from evaluating a non-certifiable configuration.

In general more than one RBD is performed for each subsystem leading to several diagrams. Every failure condition of the system is addressed by a dedicated RBD [20]. Usually RBDs are conducted manually but for the scope of this analysis an automation is needed. The RBD architecture is defined by the architectures generation and its linked to the connections among components. However the RBD evaluation is yet to be automated. A methodology to fully automate the RBD assessment is proposed in the following paragraphs.

Components inside a RBD can be, generally, in series or in parallel. Another configuration, that we call H-configuration or bridge configuration, can also be found in literature. The main characteristic of this configuration is that on component can work in two directions, being active only as a backup in case a main component fails. Figure 3 shows the three cases in a schema. Here the reader can see that in the H configuration case, if component C fails, components H and D can fulfill the function. Same logic applies to components A, B and C. This case can be easily found in aerospace systems in which redundancies are highly required (e.g. cross-feed valves, power transfer units).

Series configurations can be solved by just multiplying the reliability of each of the components involved [21], it is shown in equation 1, where *i* represents the individual components and *n* the total amount of components involved. Using the same notation, equation 2 shows how to solve parallel configurations [21].

$$R_{series} = \prod_{i=1}^{n} R_i \tag{1}$$

$$R_{parallel} = 1 - \prod_{i=1}^{n} (1 - R_i) \tag{2}$$

H architectures need to be solved following a different approach. A statistical approach is used. The model can be solved by applying Bayes theorem of conditional probability. The architecture is divided into two separate cases or partitions. One represents the component H always working, the other shows the case with component H never working. Both cases are represented in figure 4

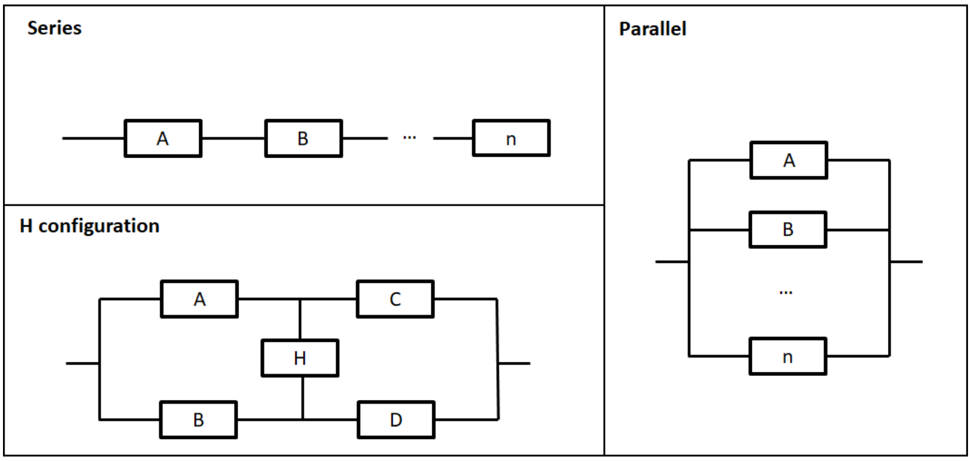

Figure 3 – RBD components configurations schema.

where this concept shows this in a schematic way. Equation 3 shows how to evaluate the reliability of the whole compound of components by the use of Bayes theorem after the two partitions are done. What the theorem says on this case is that the probability of the system working is the probability of component H working multiplied by the probability of partition 1 working plus the probability of the component H not working multiplied by the probability of partition 2 working. This statistical approach is specified in the IEC-61078 reliability block diagram standards [22]. More non-series and non-parallel architectures are found here, they are solved in an analogous was so just the H-configurations are analyzed in this study.

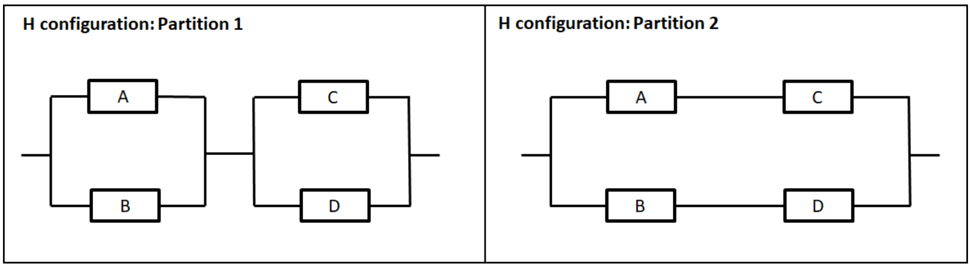

Figure 4 – Bayes model partitions for H-configurations solving through statistical analysis.

$$R_{configuration_H} = R_{partition_1} R_{component_H} + R_{partition_2}(1 - R_{component_H}) \tag{3}$$

The possible connections among components have been defined. The next step to reach the RBD automation is to define the components. The proposal is to create an object per component with certain information stored. Each component has six attributes that are needed in order to be able to properly define it. Figure 5 shows the schema of the object. The attributes are now explained in depth:

- The attribute name serves just as support for the designer to keep track of the components and to identify them in an easier way. Some examples could be "hydraulic actuator", "pump" or "engine".

- The ID attribute is a unique number for each component and is used by the algorithms to count and keep track of them. It can be easily read from the architecture generation. Two components cannot have the same ID number. It is represented with natural numbers from one to the total amount of components.

- The attribute type is used as a mark for components that do not work on a classic way. This means for instance H-configuration components but also other concepts can be used here (e.g. r-out-of-n models [21], stand-by architectures). It is used as an extra support for the

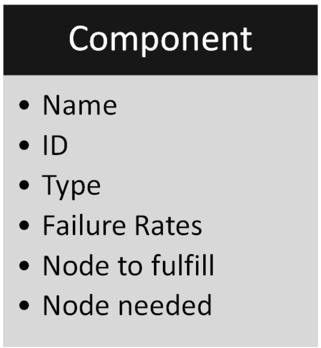

Figure 5 – Object-oriented structure of the components.

algorithms to identify non-conventional RBD components that are not solved by just series or parallel equations. And example of type would be type "H-configuration" or "r-out-of-n".

- The attribute regarding the failure rates can vary depending of the scope of the study. It can be a constant value if a simplified analysis is performed. However, another approach needs to be done if the whole component's failure rates bathtub curve is to be considered. In this case the attribute could be a vector which components store mathematical parameters that define the curve (e.g. Weibull distribution [21]). It is important to estimate the reliability from the failure rates in a consistent way. Each components reliability can be estimated following equation 4. The variable time can be set as one flight hour if the failure rates are consider as constant. Otherwise all the calculations can be performed time-dependant.

$$R_{component} = e^{-\lambda t} \tag{4}$$

- The last attributes are called nodes. This is a key item that makes the automation process possible. Each component is located between two nodes, denoting then the connections. The node to fulfill refers to the node after the component (can be also seen as the function the component fulfills). While the node needed is the node before the component (or function needed by the component). Components from H-configurations need the extra information stored in the type attribute since the nodes to fulfill or needed are not enough to define the architecture.

Figure 6 shows a RBD example with the components and nodes definition. This RBD is just an example in which all the different cases and architectures can be found. It does not represent any real architecture in particular. The automation algorithms are now explained in the following paragraphs and the RBD shown figure 6 is used a a test case. As seen in the figure there are some rules regarding the nodes definition. There has to be always at least one component between two nodes and there has to be always only one node between two components.

Regarding the numbering of the nodes, some rules are also required. The initial node (initial function) must be assigned with the number zero. The last node (last function) must be assigned with the number one. The rest of the nodes must be numerated sequentially with natural numbers until all have been covered. For instance in figure 6 nodes 2 and 5 could switch the numbers, the specific number given to each node is not relevant as well as they are always sequential and natural numbers are not skipped. Component G for example has the attribute "node to fulfill" equal to 4 and the attribute "node needed" equal to 9. Components A, B and C have the same node to fulfill (i.e. node 1) and node needed (i.e. node 2). It can be noticed that component J is one of the previously called "H-configuration component". For this specific component the node to fulfill and node needed are 5 and 6, which one is assigned to which one is irrelevant but it is important to designate the attribute "type" as "H-configuration".

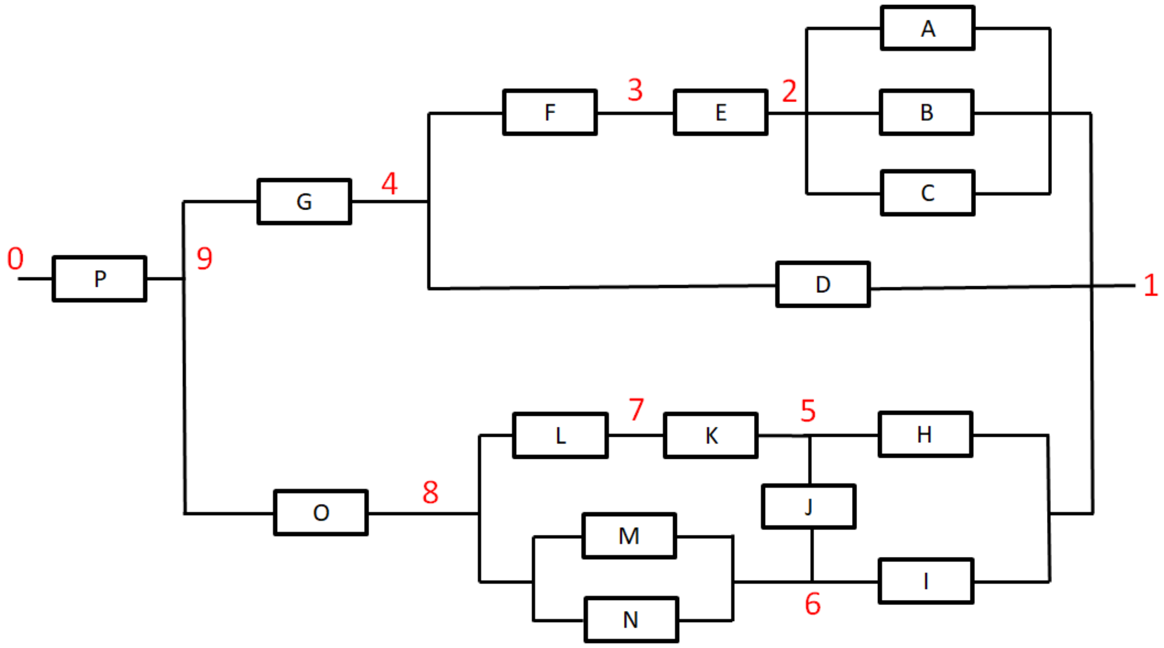

Figure 6 – RBD example with components in black and nodes in red.

3.2.1 Algorithms logic

The algorithm logic for RBD automation is now explained in these paragraphs. The first step is to count the total number of components and nodes, this can be easily done since the input consists of a list of components following the data structure specified in figure 5. Then all components are transformed into a new class that we called conjunct. This allows to merge and separate later components together and to keep track of them during the RBD transformation. This conjunct class has the same structure of the components but with the failure rates already transformed into a reliability value. Up to this point all the components specified in the input have been transformed into conjuncts (one conjunct per component) and have a reliability value associated to the specified time frame. The algorithms are now applied as it follows.

The algorithms look for specified structures in the RBD. Components (now called conjuncts) are merged in other conjuncts if these said structures are found. For example, if two components (A and B) are in parallel, the algorithms transform them into two conjuncts (A and B) and them perform the pertinent mathematical operations returning as a result a single conjunct (A-B) with the reliability value equivalent of the parallel between both of them. Another example, a component (C) is in series with these two previous components. The algorithms first create the parallel conjunct (A-B) and then perform the series model between conjunct (C) and conjunct (A-B). The result is conjunct (C-A-B) that has the same reliability value as A-B in parallel, plus C in series with A-B. Through a series of iterations the RBD gets reduced on each of them after the specified structures are found. The main idea has been explained in general terms, some specific instructions on how to perform this automation are now given.

The diagram represented in figure 7 explains the logic. It can be noticed that the first step is to transform all the components into conjuncts of themselves. Then a iteration process starts until all the RBD has been reduced to one conjunct. This final conjunct represents the reliability value of the whole system. The iteration process works as follows.

- First of all, the algorithms look for pure H-configurations. This means, only five components are involved. This condition is can be easily checked thanks to the information stored in the nodes. There is another loop inside this block from one to the total number of conjuncts that checks each of them looking for the conjunct type "H-configuration" (previously called component type). If the condition is met then the nodes associated to this component are checked. If there is only two components associated to each node this means that a potential pure H-configuration could be present. A last check needs to be done. The conjuncts linked to this two nodes are checked,

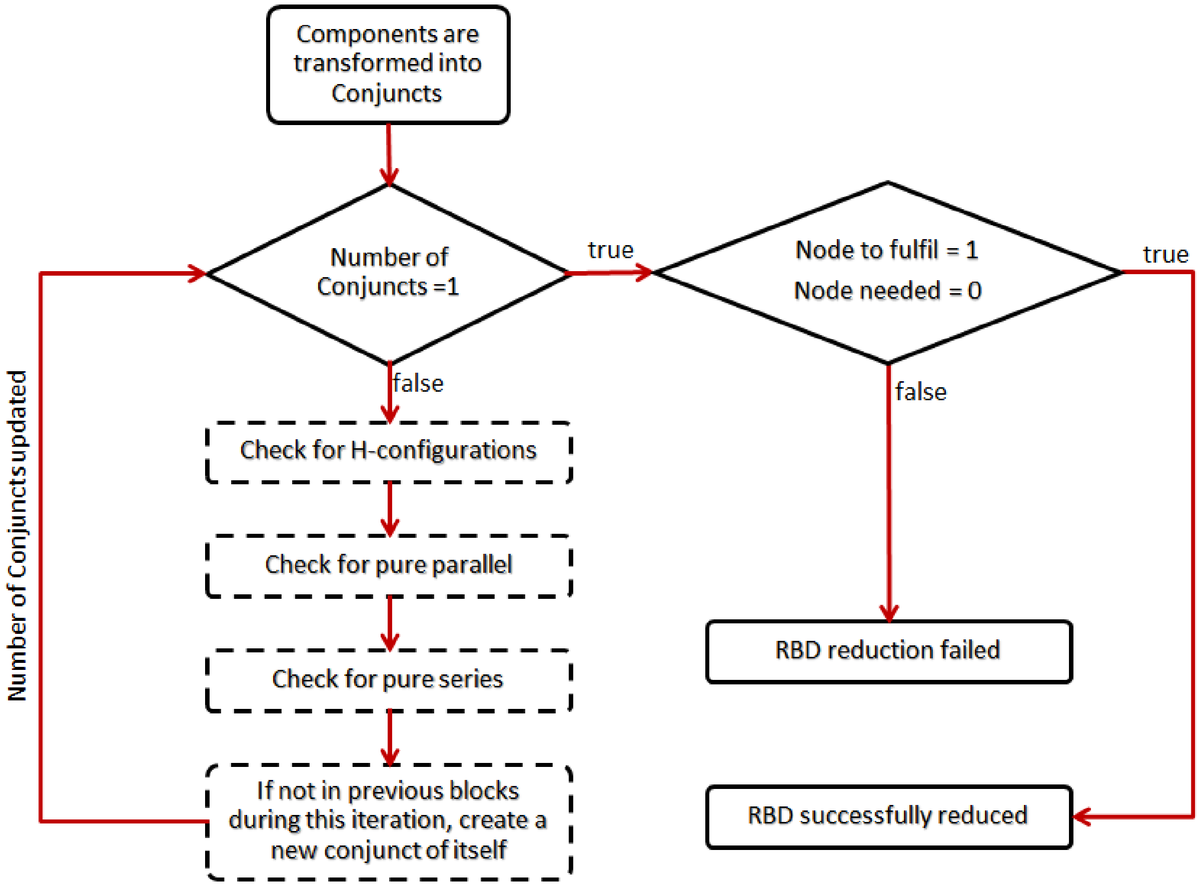
Components are transformed into Conjuncts
Number of Conjuncts =1
true
Node to fulfil = 1
Node needed = 0
true
false
false
Check for H-configurations
Check for pure parallel
Check for pure series
If not in previous blocks during this iteration, create a new conjunct of itself
RBD reduction failed
RBD successfully reduced
Number of Conjuncts updated

Figure 7 – Algorithms logic diagram for RBD automation.

if two and two of them share needed nodes and nodes to fulfill respectively, then a pure H-configuration is found and those 5 conjuncts are reduced into one following Bayes equation (equation 3). The ID associated to this new conjunct is 1. If more than one H-configurations are found in the same iteration then the IDs are sequentially numerated from 1 following the natural numbers. If no H-configuration is found while checking all the conjuncts then the next block is run. Other non-conventional configurations can also be added here (e.g. r-out-of-n).

- Pure parallel configurations are seek in this step. This means, only two conjuncts in parallel. Here a loop that checks all the conjuncts is needed. Another loop is added inside in which the algorithms compare the current conjunct with all the others. If a specific condition is met between both conjuncts a parallel is performed (following equation 2) and both conjuncts are reduced into one. This condition is that both conjuncts have the same node to fulfill and the same node needed. Another extra step needs to be done. When two conjuncts are reduced, their IDs are saved into a list. When the loops are iterating among the conjuncts, if these IDs are found the loop is skipped. This avoids the algorithms to perform the parallel two times between the same components (i.e. parallel A-B and parallel B-A). Also, if a third component is in parallel with other two, the algorithms reduce two of the components and stay on hold for the third one. There is an example of this in figure 8 which is explained in later paragraphs. It is noteworthy to mention that the list with repeated IDs is temporary and gets restarted after the algorithms finish renumbering the conjuncts at the end of the nested loop.

- Here the algorithms look for pure series configurations. The logic is analogous to the one used in the parallel configurations. Only two conjuncts are reduced at each time and their IDs are saved to avoid repetitions. However the logic to identify whether a conjunct is in series or not is different. In this case the upper loop checks nodes. Then for each node the lower loop checks the conjuncts. Every time a conjunct has the current node as node to fulfill a counter increases. Every time a conjunct has that node as node needed another counter increases. At the end of the conjunct count for each node a condition is evaluated. If the total amount of conjuncts with that node as node to fulfill is one and the total amount of conjuncts with that node as node needed is one, then that means that those two conjuncts are in series. Both get reduced into a series conjunct following equation 1.

- All the previous new-made conjuncts were renumbered with new IDs and saved on specific lists. The ones that remained unchanged during this iteration shall also be renumbered now. As a result, the total number of conjuncts is always lower after each iteration. Convergence is found when this number is equal to one. As an example to the reader a hypothetical case is now explained. A set with a initial number of ten conjuncts is considered. During the iteration a parallel and a series operation are performed. This implies that four conjuncts are merged into two. The new parallel and series conjuncts have IDs one and two respectively. The rest of conjuncts shall now be renumbered from three to eight.

If the number of conjuncts is equal to one that means that the whole RBD has been reduced after the iterations to just one conjunct which reliability equals to the whole system reliability. A last check is performed in order to double check that the algorithms correctly reduced the diagram. This check ensures that the initial function (node 0) and the final function (node 1) are respectively needed and fulfilled by this conjunct. The single conjunct fully represents the systems behaviour if the condition is true. There is some internal mistake if the condition is false.

3.2.2 Test case

The iteration process through the algorithms for the example RBD in figure 6 is represented and explained now in this section. Figures 8, 9 and 10 show the process for extra clarification and as an example to the reader. It can be noticed how at each iteration new conjuncts of components are made and some nodes disappear. Also it is noticeable how the triple parallels are solved by performing a double parallel between two of them and them doing another double parallel between the third component and the conjunct with the two previous ones. H-configurations also stay on hold until it is reduced to five components, moment in which all get reduced to just 1 conjunct. After seven iterations the whole RBD gets reduced to just one big conjunct which reliability value equals to the one produced by the initial diagram. Automation has been achieved. Iterations one and two are now fully explained.

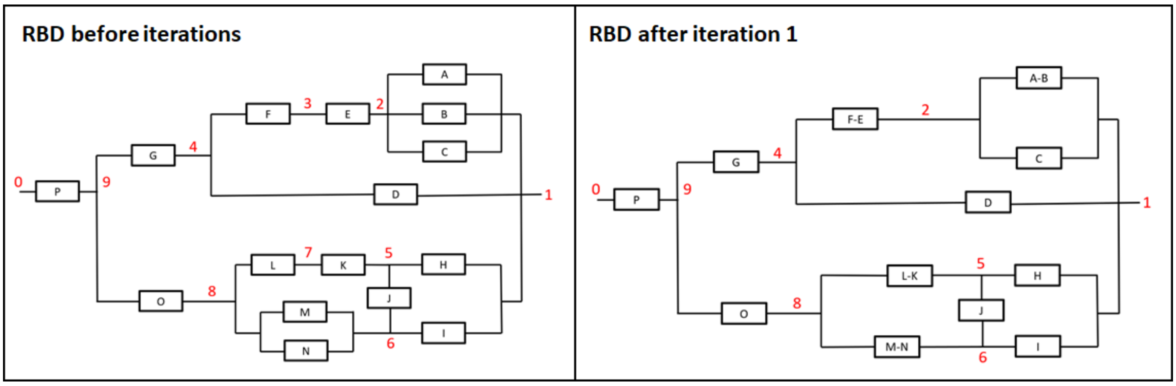

Figure 8 – RBD reduction through the algorithms: iteration 1.

Focusing now on the first iteration. The initial number of conjuncts is 16, which is equal to the number of components since it is the first iteration. The algorithms find a component with the type corresponding to the H-configuration. However the other conditions are not met so the system is not reduced yet into a smaller conjunct. Checking for parallel configurations the algorithms reduce conjuncts A and B and M and N into two conjuncts (A-B and M-N respectively). Since conjuncts A and B have already been taken during this iteration, conjunct C stays on hold. The IDs for these two conjuncts are 1 and 2. Then, series configurations are seek. The algorithms reduce two more conjuncts (F-E and L-K) with IDs 3 and 4. The rest of conjuncts are not changed during this iteration

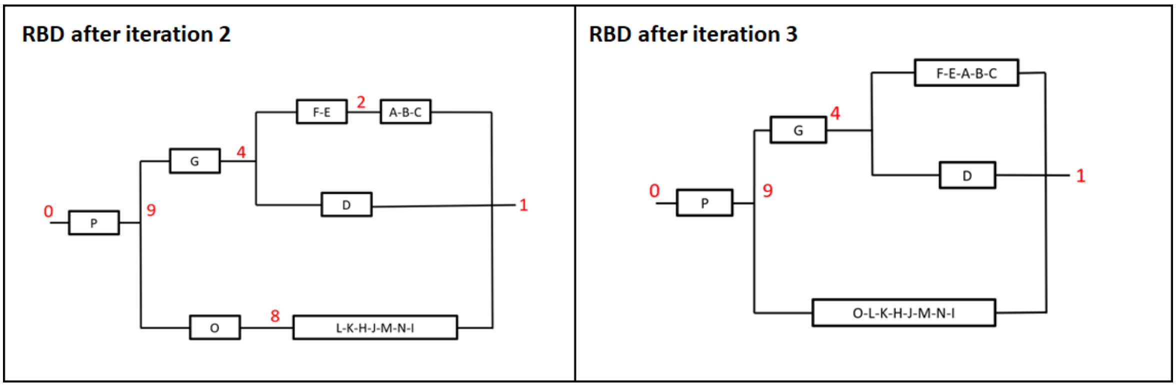

Figure 9 – RBD reduction through the algorithms: iterations 2 and 3.

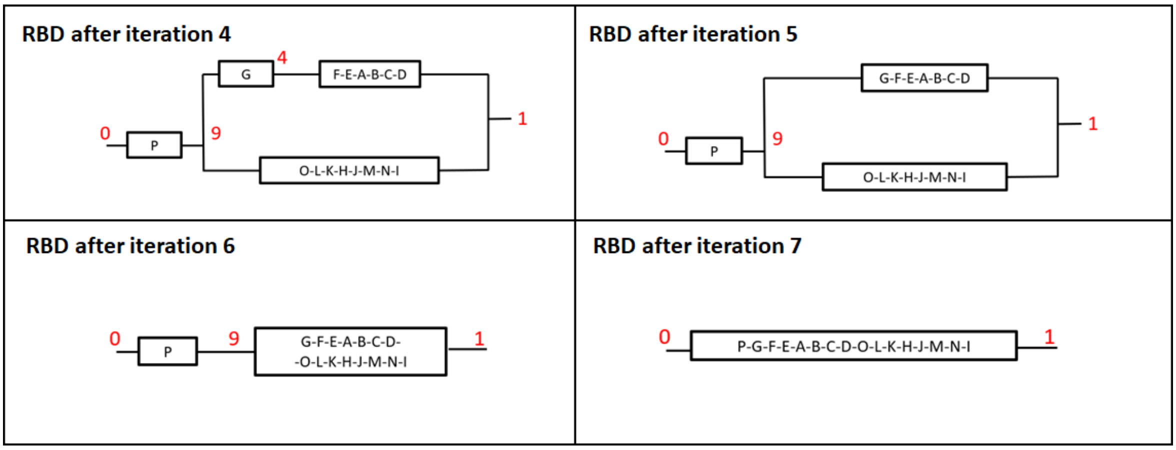

Figure 10 – RBD reduction through the algorithms: iterations 4, 5, 6 and 7.

and their IDs are now reconfigured sequentially from 5 to 12. Leaving a total amount of 12 conjuncts for the next iteration.

During the second iteration the algorithms immediately reduce the H-configuration into 1 conjunct since the conditions are met. Giving these new five-blocks conjunct (LK-J-MN-H-I) the ID number 1. Then conjunct AB is joined in parallel with conjunct C, giving it ID number 2. No series conjuncts are found during this iteration. The rest of conjuncts are left as before renumbering the IDs from 3 to 7. The rest of the iterations perform the same analysis until convergence into one conjunct is achieved. Seven iterations are needed for this specific case.

4. Aeronautical application case: A320 flight spoilers for roll control

An aeronautical application case is now shown. The aircraft chosen is the A320 since information about its on-board systems can be found. This example focuses on the reliability of the function of roll control particularized for the case of the hydraulic part of the spoilers.

The spoilers schema is shown in figure 11. It can be seen that the spoiler group is formed by ten spoilers, five on each wing. However, for roll control only eight of them are used, leaving the most internal ones only as speed brakes. Roll control is performed also by the ailerons but only the spoilers are considered right now. Both groups (i.e. aileron group and spoiler group) should be considered in parallel to assess the whole roll control function reliability. Eight spoilers are considered and each is connected to a different hydraulic line. The connections are symmetric, for example, spoiler two left and two right are connected to the same line. This logic also applies to the other ones. This allows a symmetric control, which means, roll is performed by pairs of spoilers. If one spoiler actuator fails (e.g. spoiler three right) the associated pair is not used (i.e. spoiler three left) and another pair of spoilers is used for the control of roll moments. This translates as left and right spoilers being in a series configuration in a reliability block diagram.

The hydraulic system schema is shown in figure 12. Three hydraulic lines are present, blue, green

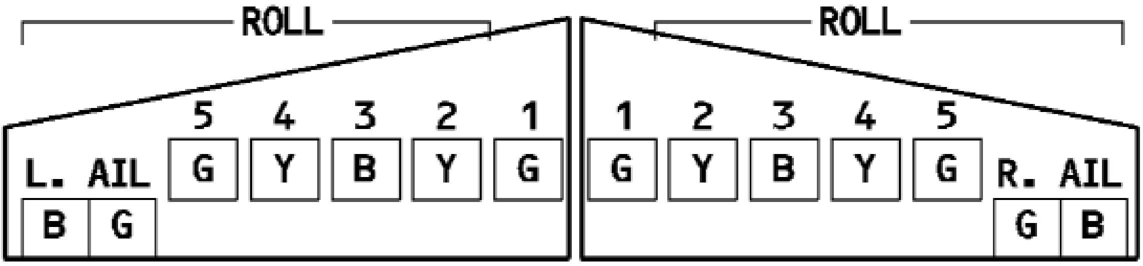

Figure 11 – A320 spoilers schema, from [23]

and yellow. Each line has a reservoir that stores the hydraulic fluid. The green line is powered by a pump connected to engine one. The yellow line is powered by two pumps, one connected to engine two and the other one connected to an electric motor. The hand pump is only used for the cargo doors so it is not considered for this analysis. The green and yellow lines are also connected between them by a power transfer unit, a component that allows to transfer the hydraulic fluid from one line to the other one. Lastly, the blue line, usually used as back-up, in powered by also two pumps, one connected to an electric motor, the other one connected to the ram air turbine. This line is not connected to the others.

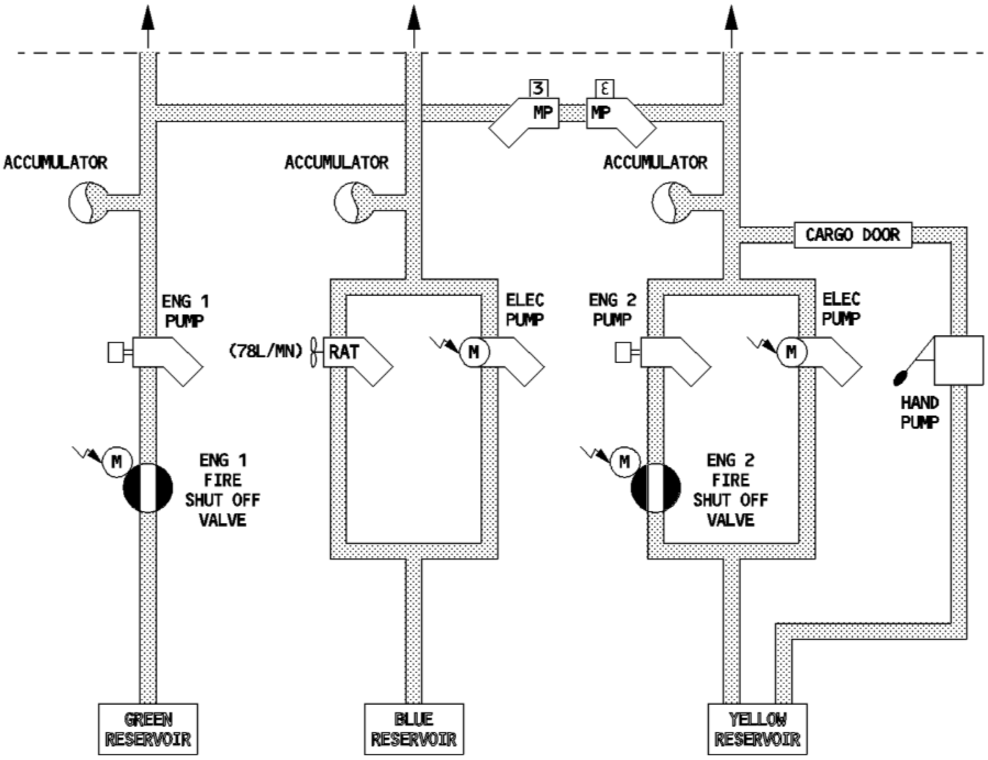

Figure 12 – A320 hydraulic system schema, from [23]

Figure 13 shows the resulting reliability block diagram. Some notation explanations are now provided for a better understanding. "Sp.5 R" means actuator of the right spoiler number five. It can be seen how each spoiler group is represented in series and connected to the corresponding hydraulic line. These lines are shown from the reservoir (which contains the reliability of the valves, sensors, etc...) to the end of the line going through the pumps and their corresponding movers. The power transfer unit is located as a H-configuration. At least one of the four spoiler groups must be functional to guarantee the fulfilment of the function (i.e. roll control with spoilers), so they are connected in parallel. It is noticeable to see that this RBS can be solved by the proposed methodology and can be easily automated.

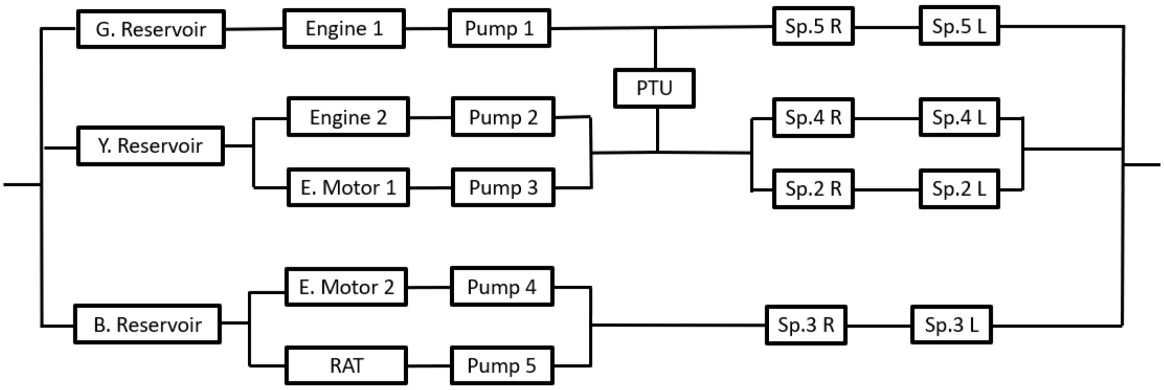

Figure 13 – RBD: Hydraulic side of flight spoilers for roll control

It is noteworthy to say that the RBD depends on the functionality to asses and not on the components. This means that the position of the components in the diagram might change depending on the mission phase and/or the function. For instance, when spoilers are used as speed brakes during landing the RBD is completely different. On this case the ten spoilers should be considered (five spoiler groups). A deeper analysis should be performed but as an example one could say that at least three spoiler groups shall function in order to carry out the function. Hence another RBD configuration should be proposed in which the five spoiler groups would be in parallel and connected to a "r-out-of-n" component. For this example it would be a "3-out-of-5" configuration.

5. Conclusions

The need for automated models for on-board systems architectures design has been highlighted. This comes from the fact that the huge number of possible connections among systems is non-viable to be assessed without automation. Certification aspects are key to ensure that automatically generated architectures are viable and doable. This is important when assessing innovative technologies that might require a huge exploration of architectures in order to find new more-optimal solutions.

This paper proposes an automated methodology to assess certification requirements of aircraft on-board systems architectures during early stages of design. It is divided in two parts. First part extracts architectures requirements from other requirements that do not explicitly refer to on-board systems. These requirements are used as a filter to assess which architectures are potentially certifiable and which ones do not met the minimum expected requisites. The second part focuses on one of the requirements that is related to safety assessment. As a result, the reliability block diagram technique needs to be automated in order to fully automate the methodology and be able to verify all of the extracted requirements. A full automation is achieved and showed through an example test case. An aeronautical application case for flight spoilers is also shown to prove the applicability and relevance of the methodology. Hence this analysis allows to automatically verify preliminary certification requirements for on board systems architectures. This allows to analyze and investigate a huge number of possible solutions without loosing certification aspects.

6. Acknowledgements

The research presented in this paper has been performed in the framework of The AGILE 4.0 project [12] and has received funding from the European Union Horizon 2020 Programme under grant agreement n° 815122.

The authors would also like to thank and acknowledge Andrew K. Jeyaraj from Concordia University, Motreal, for the support given on this topic during the discussions within the AGILE 4.0 project.

7. Contact Author Email Address

mailto: carlos.cabaleirodelahoz@dlr.de or carlos.cabaleiro@polito.it

8. Copyright Statement

References

[1] Chakraborty I. *Subsystem architecture sizing and analysis for aircraft conceptual design*. PhD thesis, Georgia Institute of Technology. 2015.

[2] Liscouët-Hanke S. *A model-based methodology for integrated preliminary sizing and analysis of aircraft power system architectures*. PhD thesis, Institut National des Sciences Appliquées de Toulouse. 2008

[3] Armstrong M.J. *A process for function based architecture definition and modeling*. PhD thesis, Georgia Institute of Technology. 2008

[4] De Tenorio C. *Methods for collaborative conceptual design of aircraft power architectures*. Georgia Institute of Technology. 2010

[5] Jackson D.W. *Robust aircraft subsystem conceptual architecting*. PhD thesis, Georgia Institute of Technology. 2013

[6] Chiesa S. *Affidabilità, sicurezza e manutenzione nel progetto dei sistemi*. CLUT. 2008

[7] Armstrong M.J. *Identification of emergent off-nominal operational requirements during conceptual architecting of the more electric aircraft*. Georgia Institute of Technology. 2011

[8] SAE Aerospace. ARP4761: *Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment*. U.S.A., SAE Committee S-18, Soc. Automotive Eng., Warrendale, PA, USA. 1996.

[9] SAE Aerospace. ARP4754: *Certification Considerations for highly-integrated or complex aircraft systems*. SAE Systems Integration Requirements Task Group AS-1C, ASD., REV. A, Society of Automotive Engineers, Inc. 2010

[10] Brusa E. *Digital twin: toward the integration between system design and RAMS assessment through the model-based systems engineering. IEEE Systems Journal*, 15(3), 3549-3560. 2020.

[11] Chiesa S et al. *ASTRID–aircraft on board systems sizing and trade-off analysis in initial design. Research and Education in Aircraft Design–READ*, 1-28. 2012

[12] Towards cyber-physical collaborative aircraft development. AGILE 4.0 H2020 research project [Online]. Available: https://www.agile4.eu/.

[13] Moir I., & Seabridge A. *Aircraft systems: Mechanical, electrical, and avionics subsystems integration (Vol. 52). John Wiley & Sons*. 2011.

[14] Sarlioglu B., & Morris C.T. *More electric aircraft: Review, challenges, and opportunities for commercial transport aircraft. IEEE transactions on Transportation Electrification,* 1(1), 54-64. 2015.

[15] Bornholdt R., Kreitz T., & Thielecke F. *Function-driven design and evaluation of innovative flight controls and power system architectures.* Technical report, SAE Technical Paper, 2015.

[16] Haitao Q.I., Yongling F.U., Xiaoye Q.I., & Yan L. *Architecture optimization of more electric aircraft actuation system. Chinese Journal of Aeronautics*, 24(4), 506-513. 2011.

[17] CS25 EASA. *Certification specifications for large aeroplanes*. 2009.

[18] CS23 EASA. *Certification specifications for normal, utility, aerobatic, and commuter category aeroplanes*. 2009.

[19] Jeyaraj A.K., Bussemaker J.H., Liscouët-Hanke S., & Boggero L. *Systems architecting: a practical example of design space modeling and safety-based filtering within the AGILE 4.0 project. In the 33rd congress of the international council of the aeronautical sciences. ICAS* 2022.

[20] Schallert C. *Integrated safety and reliability analysis methods for aircraft system development using multi-domain object-oriented models*. 2016.

[21] Rausand M. & Hoyland A. *System reliability theory: models, statistical methods, and applications* (Vol. 396). John Wiley & Sons. 2003.

[22] IEC. 61078. *Analysis techniques for dependability-reliability block diagram and boolean methods. Geneva: IEC*, 2006.

[23] Airbus S.A.S, ”Airbus Training - A320 Flight Crew Operating Manual”