Reconfigurable IMA platform: from safety assessment to test scenarios on the SCARLETT demonstrator

Claire Pagetti, Pierre Bieber, Julien Brunel, Kushal Gupta, Eric Noulard, Thierry Planche, François Vialard, Clément Ketchedji, Bernard Bésinet, Philippe Desprès

▶ **To cite this version:**

Claire Pagetti, Pierre Bieber, Julien Brunel, Kushal Gupta, Eric Noulard, et al.. Reconfigurable IMA platform: from safety assessment to test scenarios on the SCARLETT demonstrator. Embedded Real-time Software and Systems, ERTS 2012, Feb 2012, Toulouse, France. hal-02170919

HAL Id: hal-02170919
https://hal.science/hal-02170919

Submitted on 2 Jul 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Reconfigurable IMA platform: from safety assessment to test scenarios on the SCARLETT demonstrator

Claire Pagetti[1] Pierre Bieber[1] Julien Brunel[1] Kushal Gupta[1] Eric Noulard[1] Thierry Planche[2] Francois Vialard[3] Clément Ketchedji[4] Bernard Bésinet[2] Philippe Despres[2] [1] ONERA - Toulouse, France, [2] Airbus Operations SAS - Toulouse, France, [3] Aeroconseil - Toulouse, France, [4] SII - Toulouse, France

***Abstract*—The next generation of IMA platforms should include reconfiguration capabilities in order to limit the effect of some hardware failures on aircraft operational reliability. The contribution of this paper is to describe the safety assessment process from the safety assessment on the preliminary design of a reconfigurable IMA architecture to the execution of the failure scenarios on the SCARLETT demonstrator. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7 / 2007-2013) under Grant Agreement n° ACP7-GA-2008-211439.**

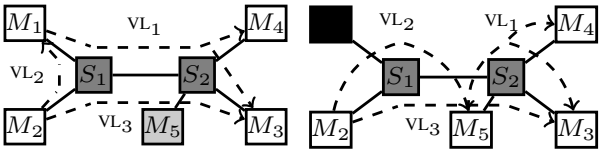
I. INTRODUCTION

The Integrated Modular Avionics (IMA) architectures have been defined to design avionics platforms that share communication and computation resources. The behaviour of such platforms is imposed by two standards. On the one hand, Arinc 653 [ARI97] specifies the management of computing resources (named modules): the scheduling of partitions on each module is defined off-line by a periodic sequence of slots. On the other hand, Arinc 664 [ARI09] describes the management of the communication resources (switches and end-systems): communication flows are statically organised into Virtual Links (VL) which correspond to multicast channels characterised by a minimal time interval separating two successive messages in a same VL.

A. Reconfigurable IMA platforms

These two standards globally define the IMA concept which has been implemented in the Airbus A380 and the Boeing B787 for instance. The partners of the SCARLETT[1] project are preparing the next generation of avionic platforms which will integrate reconfiguration capabilities. Reconfigurable IMA should be able to change the configuration of the platform by moving applications hosted on a faulty computing module to spare computing modules. The main objective of such an extension is to reduce the cost of unscheduled maintenance and to improve the operational reliability of the aircraft while preserving current safety levels.

Let us illustrate the notion of reconfigurable IMA through a simple example: the platform is composed of five modules $(M_1, \ldots, M_5)$ and two communication switches (S_1, S_2). The initial configuration is drawn in Figure 1(a): module M_5 is a spare initially shut down and free of application (grey in the picture). If some failure occurs on module M_1, the applications initially hosted on this module can be reconfigured on the spare M_5 and all communications from and to M_1 (VL_1,VL_2,VL_3) must also be rerouted according to this new allocation. The reached configuration is shown in Figure 1(b).



(a) Initial configuration (b) Configuration after M_1 loss

Fig. 1. Example of a reconfiguration

B. Purpose of the paper

In [BNP+09], we introduced the preliminary design of reconfigurable IMA-2G platforms. We discussed different reconfiguration policies that describe the set of rules and parameters that govern the deterministic procedure of reconfiguration. According to these policies, we enumerated the number of reachable configurations. In [BBN+10], we assessed the safety of the reconfigurable platform: we first performed a Functional Hazard Analysis (FHA) [SAE] in order to define the Safety Requirements that should be satisfied by the reconfiguration mechanisms, and then we performed a preliminary assessment of the proposed reconfiguration mechanisms to prove that the addition of these mechanisms does not degrade the safety of the aircraft.

For that purpose, we formally modelled the functional and dysfunctional behaviours of reconfigurable IMA-2G using the ALTARICA language [APGR99]. From this model, tools automatically generated failure scenarios leading to potentially unsafe situations. These scenarios were analysed in order to check that the proposed mechanisms satisfied qualitative safety requirements.

The contribution of this paper is to describe safety assessment activities related to the verification of reconfiguration mechanisms implementation. We explain how failure scenarios generated during preliminary safety assessment can guide a

[1]http://www.scarlettproject.eu/, Point of contact: didier.hainaut@fr.thalesgroup.com

part of the tests performed on the SCARLETT reconfiguration demonstrator.

To perform these activities, we need:

1) a representative safety model of the system implemented in the demonstrator,
2) a demonstrator including several tools to perform fault injection and a set of instrumentation tools to observe the behaviour of the equipments, partitions and VLs;
3) a manner to correlate failure scenarios generated from the safety model and the tests executed on the demonstrator.

The outline of the paper is the following. Section II recalls the general reconfiguration principles which include the off-line reconfiguration graph definition and the on-line safe steps of reconfiguration. Section III describes the ALTARICA model of reconfigurable IMA. The model was extended to take into account the network which was out of the scope of the previous papers. The model has also been tuned to closely represent the demonstrator. Section IV presents the SCARLETT reconfiguration demonstrator led by AIRBUS. We show how failure scenarios generated by the ALTARICA model can be run on the demonstrator.

II. RECONFIGURATION PRINCIPLES

In this section, we recall the principles of the reconfiguration mechanisms.

A. Reconfiguration choices

The objective of incorporating reconfigurable capabilities in IMA is to increase the operational reliability. When a computing module fails, a reconfiguration can be launched if this failure has an operational reliability impact, meaning that the aircraft becomes NOGO. In SCARLETT, only module failures can trigger a reconfiguration. Several reconfiguration scenarios have been considered which are characterised by:

- Granularity level: reconfiguration is either performed at module or at partition level. For module level reconfiguration, a spare computing module is allocated to all applications running on the faulty computing module. For partition level reconfiguration, spare partitions running on non-faulty modules are allocated to the applications running on the faulty computing module.
- Location: reconfiguration can be performed either locally on modules belonging to the same cluster than the faulty module or distantly on any module of the platform.
- Time: reconfiguration can be performed during the flight or on ground when the aircraft is stopped.

B. Reconfiguration graph

When building a reconfigurable platform, the designer computes off-line a certified *reconfiguration graph*. Such a graph contains all the admissible configurations (including the initial one) that can be reached after failures, and the allowed transitions between these configurations.

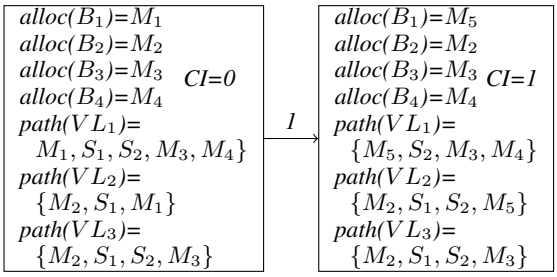
Definition 1 (Configuration)**.** *A configuration* $\langle alloc, path \rangle$ *simply describes the allocation of the partitions on the computing modules and the allocation of the VLs on the physical network elements (switches and modules):*

$$\begin{cases} alloc : Partitions \longrightarrow Modules \\ path : VL \longrightarrow 2^{Modules \cup Switches} \end{cases}$$

Definition 2 (Graph of reconfiguration)**.** *A graph of reconfiguration* $G = \langle V, E \rangle$ *describes the various states that could be reached during reconfiguration on an IMA platform.* V *is a finite set of nodes, where each node is a configuration, and* $E \subseteq V \times \mathbb{N} \times V$ *is the set of transitions. Each transition is labelled by an integer corresponding to the failure of a module.*

Each node (or configuration) is encoded by a unique identifier named configuration index *(CI).*

Example 1. *Let us describe the reconfiguration graph corresponding to Figure 1 given in introduction. There are two states: the initial configuration (with configuration index CI=0) and the one reached after the loss of module* M_1 *(with CI=1). We assume that there are exactly 4 applications* $\{B_1, B_2, B_3, B_4\}$*, one per module.*



In the configuration reached after the loss of module M_1*, application* B_1 *is allocated on module* M_5 *and the paths for communications sent and received by* M_1 *(*VL_1*,*VL_2*,*VL_3*) are changed. The allocation of other applications and communications remains unchanged.*

The number of modules and spares and the reconfiguration policy determines the shape and the size of the reconfiguration graph. Let's consider a very simple policy where a unique spare can be used to host the applications of any module. In that case the reconfiguration graph is very shallow. There is exactly one transition per module that links the initial configuration with a new configuration where the spare module hosts the application from the faulty module. There is no transition going out from these new configurations because as the spare is occupied no other reconfiguration is possible. More complex reconfiguration graphs have to be taken into account when dealing with multiple spares and with policies that restrict the usage of spares.

C. Actors

Reconfiguration capabilities are handled by 4 applications represented in Figure 2. The *Reconfiguration Supervisor* (RS) is the central entity that controls all reconfiguration activities including selection of the spare module that replaces the faulty module. The *Centralized Maintenance System* (CMS) hosts

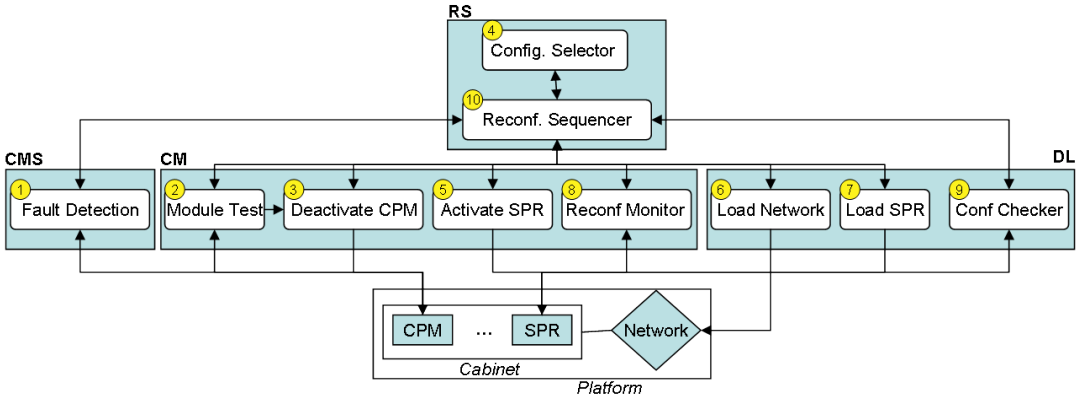


Fig. 2. Functional architecture

the fault detection and diagnosis function which periodically checks all computing modules (CPMs) for their health and elaborates a consolidated diagnostic. The *Cabinet Manager* (CM) monitors the reconfiguration activities, shuts down the failed module and powers up a spare (SPR). A *CM* is associated with every cabinet (or cluster) where a cabinet consists of a switch and all the modules connected to the switch. The *Data Loading and Configuration System* (DLCS) comes into play at the end of the reconfiguration. Its main function is to load the spare with the application, to collect and control the loaded configuration.

An alternative architecture for Reconfigurable IMA platform was proposed in the DIANA project [EJS+10]. This architecture shares the goals and assumptions of the SCARLETT architecture. The goal is also to improve aircraft operational reliability and it is assumed that the set of authorized configurations is computed off-line. The DIANA architecture is different because Fault detection, Fault diagnosis, Reconfiguration Supervision and Data Loading are implemented in a distributed way whereas they are implemented in a centralized way in the SCARLETT architecture. In the DIANA architecture, each module hosts a component that is able to test the health status of its module and to exchange it with other modules until a consensus is reached on the identity of faulty modules. Then, based on the result of the consensus protocol, the non-faulty modules select the new configuration and they load the new applications. The authors claim that the distributed implementation improves the availability and integrity of reconfiguration mechanisms. In the SCARLETT architecture, the loss of Centralized maintenance, Reconfiguration Supervisor or Data Loading leads to the loss of reconfiguration. This is not the case in the DIANA architecture. But to achieve this improvement this architecture relies on a complex and resource-consuming protocol between the modules in order to reach a consensus. The SCARLETT architecture brings a significant operational reliability improvement with a simpler architecture that inter-operates with existing system architectures for Maintenance and Data Loading.

III. SAFETY MODEL

The objective of this activity is to formally analyse the behaviour of the reconfiguration system when some of its functions fail. We are looking for combinations of function faults that lead to the loss of the reconfiguration system (reconfiguration is not performed when needed leading to the loss of some functions supported by the platform) or an erroneous reconfiguration (reconfiguration is performed incorrectly leading to the erroneous behaviour of some functions supported by the platform). We have considered that loss of reconfiguration has no safety effect but erroneous reconfiguration should be classified at least Hazardous.

A. ALTARICA *models*

The ALTARICA language [APGR99] was defined in the 90's in order to help the dependability analysis of systems. It is based on extended finite automata which can exchange values of specific variable (named flow variable) and which can be synchronised (synchronised product or broadcast). The idea is to describe the failure modes of a component as different states of an automaton.

We modelled in ALTARICA using the tool Cecilia OCAS [Sys07] developed by Dassault. The framework allows to define component libraries: this is particularly interesting when we construct several platforms since we can reuse components. For the safety assessment of the preliminary design of reconfigurable IMA, we built several ALTARICA models: a functional view of a cabinet and of a multi-cabinet. In this paper, we describe a model equivalent to the SCARLETT demonstrator architecture. We have reused several components and have specified the network components. The model is shown in Figure 3.

The model is compliant with the demonstrator design: it is composed of two cabinets. Two modules and CM_1 are connected to switch S_1. Three modules and CM_2 are connected to switch S_2. One of the modules in the second cabinet is a spare. Both switches are connected to a gateway separating critical avionics and maintenance systems. Behind the gateway are located the *RS*, *DL* and the *CMS*.

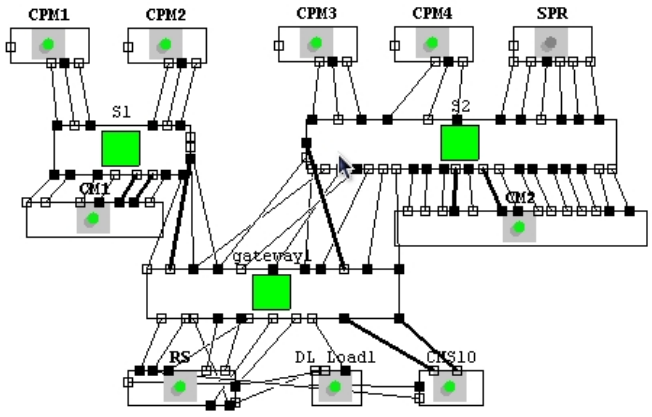


Fig. 3. ALTARICA model

The safety model describes the nominal behaviour of these components. Figure 4 shows the dialogue between a Module and reconfiguration functions in order to establish whether a module is faulty. The Module first sends its health status to the *CMS*, that informs the *RS* that module 5 is faulty. In that case the *RS* sends a request for fault confirmation to the function"Module Test" of the Cabinet Manager. If the fault is confirmed then the Cabinet Manager will switch off the faulty module and inform the *RS* that module 5 is faulty and that reconfiguration should be performed otherwise the Cabinet Manager informs the *RS* that no reconfiguration is needed.

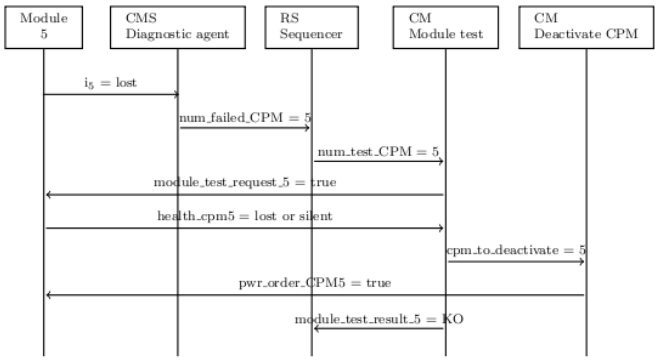


Fig. 4. Module Fault Detection

The safety model also describes the faulty behaviour of the components participating to the reconfiguration. Each component has two failure modes: lost and erroneous. During the Functional Hazard Analysis, that can be seen as equivalent to a function based FMEA (Failure Mode Effect Analysis), we have established the dysfunctional behaviour of each reconfiguration function. For example, when the *CMS* is lost the component is blocked and no fault is detected. In the dialogue shown in figure 4 the *CMS* would not inform *RS* that a module is faulty. Consequently the remaining steps of the fault detection dialogue are not performed and reconfiguration is not performed. When it behaves erroneously it detects a non failed module as failed. In that case, in figure 4 scenario, the *CMS* would inform *RS* that a module different from 5 is faulty, and the Module Test function would not confirm that this module is faulty. So, in that case, the reconfiguration would be stopped.

B. Failure scenarios

Cecilia OCAS generates automatically failure scenarios that lead to a given situation. We looked for combinations of function faults that lead to the loss of reconfiguration (e.g. the platform configuration is not changed when a module has failed). For the ALTARICA model corresponding to the demonstrator, we obtained scenarios of the form : $M_i.fault; R.fault$ where $M_i.fault$ indicates that module M_i is faulty, and $R.fault$ indicates that reconfiguration function R (one of the functions that appear in figure 2 as Fault Detection, Module Test, deactivate CPM, ...) is faulty. The sequence means that after the module has failed (so reconfiguration is needed) if function R fails then reconfiguration is not performed. For instance, when module M_1 fails if function "Configuration Selector" is lost, then a new configuration is not selected and reconfiguration is not be performed. In that case applications hosted by M_1 will be lost.

We also looked for combinations of function faults that lead to an incorrect reconfiguration. We obtained scenarios of the form : $M_i.fault; R.fault; R'.fault$ where R' is a reconfiguration function that acts as a mitigation means for R faults. For instance, when module M_1 fails if function "Configuration Selector" performs erroneously, it could select a new configuration that is inconsistent with the configuration that should be selected according to the reconfiguration graph. If the reconfiguration proceeds with this inconsistent new configuration it could be the case that some applications behave erroneously. But the fault of "Configuration Selector" is mitigated by function "Reconf Monitor" that checks that the configuration index of the configuration chosen by the selector is correct. If the configuration index is not correct then reconfiguration is not performed.

These scenarios were used to show that the preliminary design of reconfiguration mechanisms enforced their qualitative safety requirements. We showed that no single failure could lead to the loss of reconfiguration and that no double failures could lead to an erroneous reconfiguration.

The scenarios can also be used to test the implementation of reconfiguration mechanisms. For each function R of the reconfiguration architecture, it should be checked that the fault of the implementation of R actually leads to the loss of reconfiguration where the configuration of the platform is unchanged. For functions that need a mitigation function R', then the test should also allow for the observation of the mitigation function behaviour.

IV. SCARLETT DEMONSTRATOR

A demonstrator is developed within the SCARLETT project in order to test the reconfiguration system. The main objectives are:

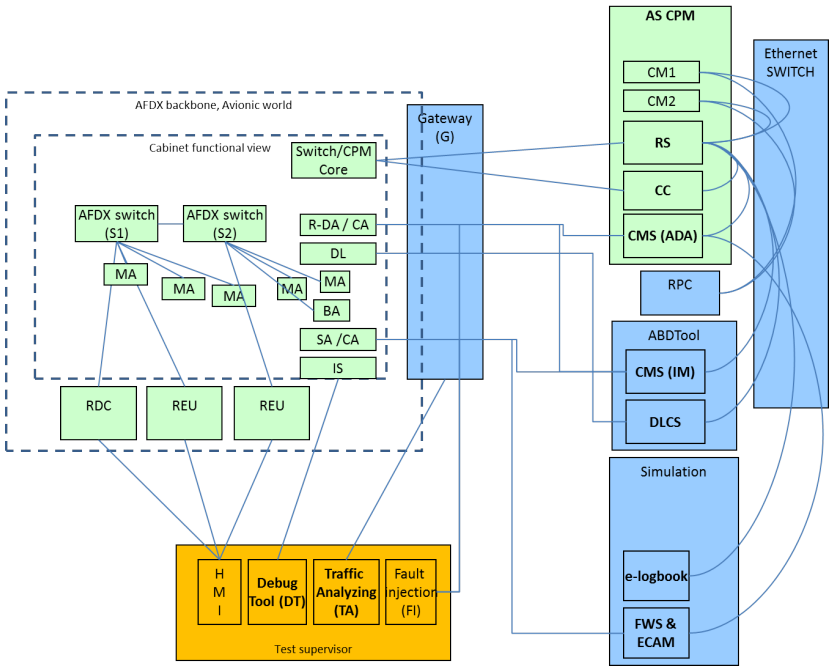


Fig. 5. Demonstrator architecture

1) to verify the feasibility of the reconfiguration principles. The mechanisms are implemented and evaluated. For instance Onboard Maintenance Tools such as failure confirmation or data-loading are automated in order to be invoked by the reconfiguration function. Also new core functions of the AFDX switches -transparent routing change- and CPM -transparent scheduling change- are under test.
2) to evaluate the performances. This includes the time required for each step of a reconfiguration, the CPU and network loads, the CPU schedule and network behaviour in term of jitter;
3) to test the failure scenarios depicted in the safety assessment.

The purpose of this section is to describe the failure scenarios on the demonstrator.

A. Demonstrator overview

The demonstrator is described in the Figure 5. The platform is composed of:

- a representative avionics bay: the avionic hardware is shown on the left side within the dashed rectangle. Partners of the project have developed 5 SCARLETT processing modules and 2 SCARLETT communication switch modules. The organisation of the connectors is flexible enough to allow several platform architectures. For instance the 5 modules can be linked to the same switch or arranged in any manner on the 2 switches like presented on the drawing. I/O modules are present on the avionics platform with one actuator (REU), 2 multi-purpose I/O modules (RDC).
- a representative airborne server: on the right of the gateway (G), in the open world area, this server hosts the 2 cabinet managers (CM1 and CM2) the reconfiguration supervisor (RS), the configuration checker (CC) as well as the aircraft diagnostic agent (ADA) which is an implementation of the CMS;
- several software simulated components, drawn on the right side, connected to the avionics bay via a gateway (G). They are composed of a data loader (DLCS), a BITE interactive mode manager (IM), a power controller (RPC) and the test supervisor that is part of the test bed. The test supervisor contains a traffic analyser (TA) to observe the traffic in the switches as well as the network behaviour during a reconfiguration, a set of debug tools (DT) to observe the modules and a fault injection (FI) unit that will generate a behaviour simulating a failure. The test bed also simulates functions that are not available on the avionic or open world modules : the flight warning system (FWS) is simulated through the capability to forward the alerts raised by the faulty system to the CMS, the e-logbook is able to present in an HMI the CMS diagnostics along with the report of reconfiguration built by the RS, an electrical interface to power up/down the processing modules, and various environmental condition generation (discrete and analogue I/O, Temperature, flight phase, time)
- the avionics applications can be classified in 2 categories. In the first category we can find the applications which are part of the health monitoring chain, the RDA and the FSA which report to the ADA and which are instrumented to inject faults (FI) under the control of the test supervisor.

Under such a command chain, RDA and FSA can raise system failures which are the conditions to trigger a reconfiguration. In the second category we have the avionics partitions which behaviour is studied when a reconfiguration occurs : the background application (BA) and the migrant applications (MA). The BA consists in a time critical application made of 2 partitions and a remote actuator. The MA is a set of 8 dummy applications distributed on several modules. When a module fails, the migrant applications initially hosted on the module are moved to a spare module or on spare on several module. During reconfiguration any side effect perturbing the BA such as loss of network frames, CPU schedule jitter, will be surveyed.

The objective of demonstration are played on the above platform using several scenarios which cover the different use cases: reconfiguration on ground as an alternative to a maintenance operation, reconfiguration in flight to preserve a good level of function, reconfiguration on a single CPM spare or reconfiguration on distributed spare, reconfiguration local to the cabinet or reconfiguration platform wide. The nominal test plan involves the health monitoring chain from detection to diagnostic, the management of failure effects and confirmation by the maintenance system, the reconfiguration decision and configuration selection, the application of a new configuration through the mean of data-loading and power management of the modules, configuration change and monitoring of operations, and finally the closure of reconfiguration with the report into the e-logbook.

B. Testing of failure scenarios

Tests are derived from the failure scenarios described in section III-A. The demonstrator designers have proposed a way to simulate each of the faults found in the scenarios. Several options are possible to simulate the faults. For instance, to simulate a function loss it is possible to physically or logically disconnect the link that is used to communicate the function output. It is also possible to implement in the software applications special modes to force the computation and communication of incorrect results. Demonstrator designers have also defined the way to observe the test scenario output in order to decide whether the test was successful or not.

Testing failure scenarios is an on-going work and preliminary results will be described during the presentation.

V. Lessons learnt

Several lessons were learnt during the design and safety assessment of SCARLETT reconfigurable IMA platform. The general lesson is that formal models can be used to support the numerous interactions between design activities and safety assessment activities.

At early steps of the system design when the Hazard analysis is performed, it is very important that the safety team can deliver safety requirements that will drive the architecture design: mitigation means, required segregation between functions and preliminary DAL allocation. At that level, an area of improvement would be to develop a library of re-usable generic components in order to have the ability to quickly build models for hazard assessment. Another area of improvement would be to apply tools that would use these models in order to generate automatically a set of safety requirements such as segregation and DAL allocation.

Fig. 6. Demonstrator

At the following steps of the system design when architecture is developed, a safety model was built. It both describes the nominal and faulty behaviour of reconfiguration mechanisms. The model was very helpful to understand the details of the propagation of faults among the main components of the reconfiguration architecture. One difficulty for the safety team was to take into consideration the various evolutions of the architecture proposed by the system designers. One area of improvement at that level would be to build the safety model as a systematic extension of a model describing the nominal behaviour that would be built by system designers.

When the system architecture is implemented and has to be tested, we have shown how to guide the testing of reconfiguration mechanism implementation thanks to scenarios generated from a safety model. The proposed approach helps to select test case that are relevant with respect to safety. One limitation of the work performed is that these scenarios might be difficult to test if the implementation does not include means to simulate the failure modes considered. To overcome this limitation it is important that the safety team provides to the implementation team, before the implementation has started, a description of function failure modes that should be tested.

References

[APGR99] André Arnold, Gérald Point, Alain Griffault, and Antoine Rauzy. The altarica formalism for describing concurrent systems. *Fundamentae Informaticae*, 40(2-3):109–124, 1999.

[ARI97] ARINC 653, Aeronautical Radio Inc. *Avionics Application Software Standard Interface*, 1997.

[ARI09] ARINC 664, Aeronautical Radio Inc. *Aircraft Data Network, Part 7, Avionics Full-Duplex Switched Ethernet Network*, 2009.

[BBN+10] Pierre Bieber, Julien Brunel, Eric Noulard, Claire Pagetti, Thierry Planche, and Franois Vialard. Preliminary design of future reconfigurable IMA platforms - safety assessment. In *27th Congress International Council of the Aeronautical Sciences (ICAS 2010)*, 2010.

[BNP+09] Pierre Bieber, Eric Noulard, Claire Pagetti, Thierry Planche, and Francois Vialard. Preliminary design of future reconfigurable IMA platforms. *In 2nd Workshop on Adaptive and Reconfigurable Embedded Systems (APRES'09), SIGBED Review*, 6(3), October 2009.

[EJS+10] Christian Engel, Eric Jenn, Peter H. Schmitt, Rodrigo Coutinho, and Tobias Schoofs. Enhanced dispatchability of aircrafts using multi-static configurations. In *Embedded Real Time Software and Systems Congress (ERTS 2010)*, Toulouse, France, 2010.

[SAE] SAE. ARP 4761, guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment.

[Sys07] Dassault System. Module OCAS: Analyse système par arbre de défaillance. manuel utilisateur ocasv4.3, 2007.