FDIR Architectures for Autonomous Spacecraft: Specification and Assessment with Event-B

Jean-Charles Chaudemar[1], Charles Castel[2], and Christel Seguin[2]

[1] ISAE-DMIA, Toulouse, France
[2] ONERA-DCSD, Toulouse, France

Abstract. On-board Fault Detection, Isolation and Recovery (FDIR) systems are considered to ensure the safety and to increase the autonomy of spacecrafts. They shall be carefully designed and validated. Their implementation involves a relevant knowledge of items like functions and architectures of the system, and a fault model in relation with these items. Thus, the event-B method is well suited to correctly specify and validate on-board safety architectures.
This paper focuses on the FDIR concept presentation and the use of event-B for formalising and for refining the FDIR concept.

1 Introduction

On-board Fault Detection, Isolation and Recovery (FDIR) systems aim at maintaining the safe spacecraft operation even when faults occur. They also enable to limit service interruptions with reduced ground operations. So, they contribute to the spacecraft autonomy.

They are complex systems composed of various mechanisms, ranging from the monitoring and activation of basic physical devices to the reconfiguration of the spacecraft mission. They are often structured in layers to master this complexity. One issue is to characterize and validate each layer and the relationship between each layer.

These characteristics require a rigorous and progressive validation of the system from the early phases of design by discharging proof obligations. Accordingly, it is necessary to use a formal method like event-B for the modelling and proof.

This paper focuses on the FDIR concept presentation and the use of event-B for formalising and for refining the FDIR concept.

The paper is organised as follows: after a short presentation of on-board FDIR concept strongly bounded with autonomy architecture concept, the next section suggest activities enabling to implement FDIR concept. Then, we present the framework of formal modelling that we will use to describe our architecture and the properties related to this architecture. The last section deals with the objectives for the future work.

2 FDIR concept

FDIR is the means to detect off-nominal conditions, isolate the problem to a specific subsystem/component, and recover of vehicle systems and capabilities [1]. In this paper, FDIR is considered as an operational function that contributes to the autonomy of the system and whose main purpose is to maintain the availability and the safety of the system [2].

The main activities related to FDIR design concern:

– identification of fault-classes that could impact the requested availability and safety objectives;
– proposition of a strategy in order to tolerate above fault-classes. A strategy suggests a logical solution to achieve these objectives;
– implementation of this strategy by proposing a static and dynamic architecture: combining functional components with safety components for diagnosis and reconfiguration.

One difficulty is often the lack of traceability between these activities. The strategy that gave the rationales of the architecture is left implicit. The proposed architecture is often the result of expert judgement. So it is hard to prove the consistency between the objectives and the architecture. Therefore our proposition is to model the objectives of the concerned system using event-B method. We first model some strategies or patterns of safety that allow to meet requirements described in the objectives of the system. Then, we show how these patterns are refined rigorously in concrete architectures by discharging proof obligations.

3 Work in progress using event-B method

Event-B is a formal method for the development of complex system. Its formalism supports the validation of some properties thanks to proof methods. Thanks to these distinctive features, the event-B method is well suited to correctly specify and progressively validate on-board safety architectures.

Let us illustrate how the three activities are modelled in event-B. For mission objectives of a spacecraft, two feared events are identified: the spacecraft loss and the interruption the mission. Safety architecture patterns propose micro architecture solutions that enable to mitigate such feared events. We propose to reuse and extend the patterns presented in [3].

For this paper, we model more specifically a safety architecture pattern that includes a primary functional component and a redundant one, under the hypothesis of no common fault. The safety property to be met is: “one single fault shall not lead to the total loss of the function”. We modelled this pattern at three abstraction levels successively refined which verify this property.

The most abstract model enables to formalise this property and hypothesis. Two basic components are considered. A fault counter (*fault_ci*, with i stands for 1 or 2) is associated to each component, whereas a boolean status (*status*) models the global system health. When a fault occurs (event *disci*, with i stands for 1 or 2), the component "i" is considered disconnected. When the event *final* occurs, nothing more happens, since the system remains in the faulty state. Moreover, for all this behaviour, the safety property is expressed by an invariant: $card(fault\_c1) + card(fault\_c2) \leq 1 \Leftrightarrow status = TRUE$. But at this step, there is no detail about the condition of spare component activation.

Accordingly, the second model refines the former with introduction of the switching process as a strategy. The activation variables are set in this new model. Two new events are defined: *sw1_2* event allows to switch from the primary active component to the spare component which becomes active when the primary one is disengaged; *sw2* event disconnects the system by disconnecting the secondary component.

At least, the third model is an implementation of this switching strategy by taking into account data flow: *normal1* (respectively *normal2*) event represents the "normal" data flow of the primary (respectively, the spare) component according to the specification; *fail1* (respectively *fail2*) event represents the "faulty" data flow of the primary (respectively, the spare) component.

4 Future work

The current work investigates B-event specifications and refinements of generic FDIR strategies by using the RODIN platform. The future work will consist in studying the impact of the concept of a "layered" FDIR and how it can be modelled and validated in the B-event framework, in the same spirit than the full constructive approach developed by [4].

References

1. NASA: Glossary - NASA Crew Exploration Vehicle, SOL NNT05AA01J, Attachment J-6. `http://www.spaceref.com/news/viewsr.html?pid=15201` (2005)
2. Chaudemar, J.C., Castel, C., Gabard, J.F., Tessier, C.: Z and ProCoSA based specification of a distributed FDIR in a satellite formation. In: CAR'07 - Second National Workshop on Control Architectures of Robots, Paris, FR (2007)
3. Kehren, C.: Motifs formels d'architectures de systèmes pour la sûreté de fonctionnement. SUPAERO, Toulouse, FR. (2005)
4. Arora, A., Kulkarni, S.: Component based design of multitolerant systems. Software Engineering, IEEE Transactions on **24**(1) (1998) 63–78