Applying the CoFI Testing Methodology to a Multifunctional Robot End-Effector

Jose Marcos Silva Anjos, Juliano Gripp, Rodrigo Pastl Pontes, Emília Villani
Instituto Tecnológico de Aeronáutica (ITA)
São José dos Campos, Brazil
jmarcos.anjos@gmail.com, juliano.gripp@gmail.com, rpastl@gmail.com, evillani@ita.br

***Abstract*—This paper describes the application of the CoFI testing methodology to the software of a multi-functional robot end-effector. FARE is a prototype end-effector for drilling, sealing and inserting fasteners in aircraft fuselage barrels, as part of the fuselage assembly process. This work is the result of a partnership between Brazilian aircraft industry and the Aeronautics Institute of Technology (ITA).**

Keywords: model based testing; industrial automation; robotic end-effector.

I. INTRODUCTION

This paper describes the application of CoFI (Conformance and Fault Injection) testing methodology to the embedded software of a multi-functional robot end-effector, called FARE (Fuselage Assembly Robotic End-effector).

This work is part of the ASAA project (Aircraft Structure Assembly Automation), which is the result of a partnership between the Brazilian aircraft industry and the Aeronautics Institute of Technology (ITA). The purpose of the ASAA project is the automation of the fuselage assembly process. The automation is a key-factor for maintaining competitiveness of Brazilian aircraft industry. It is essential for improving product quality, reducing costs and production times.

The CoFI methodology was proposed and has been traditionally used to validating space embedded systems. In this paper, we illustrate its application to an example from the aircraft manufacturing industry. FARE is a prototype end-effector for drilling, sealing and inserting fasteners in aircraft fuselage barrels.

Traditionally, the use of robots in aircraft structural assembly has been a challenge. The riveting operation is performed either manually or in dedicated machines. Dedicated riveting machines are used to assembly small parts, while the assembly of large fuselage sections is usually done manually [4].

The use of robots in the aircraft assembly process has to overcome one major problem: the lack of accuracy of industrial robots. For this reason, the robot end-effector must have an auxiliary system, such as a vision camera, and interferes in the robots' trajectory in real-time [4].

As a result, the robot end-effector must provide a number of functionalities. The end-effector software must coordinate all the end-effector components, as well as the integration with other equipment of the manufacturing cell. In the case of FARE, the software must also provide the necessary flexibility to test different parameters and configurations. This flexibility is required by the innovative nature of the project.

In this context, this paper details the benefits and limitations of the application of the CoFI methodology to the validation of the FARE control software.

The approach adopted for the CoFI application is illustrated in Figure 1.

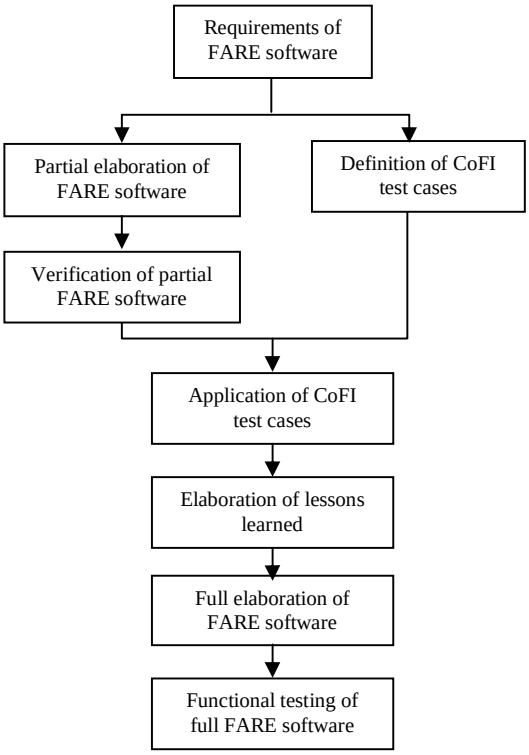

Figure 1. Approach adopted for the CoFI application.

Firstly, the FARE requirements are used for generating a partial version of the software, this partial version implements some of the functionalities required for the FARE end-effector. The partial version of FARE software is verified via simulation and basic functional testing.

In parallel, the requirements of FARE software are used to generate the CoFI test cases, which are applied to the partial version of the FARE software. The errors detected are used to elaborate a set of lessons learned, which are taken into account for the full version of the FARE software.

The approach illustrated in Figure 1 is adopted due to the limited time and human resources available for the application of CoFI methodology. The reuse of the lessons learned with the partial version of the software is possible due to the similarity among some modules of the FARE end-effector.

In order to evaluate the interference of the human factor in the CoFI results, a second person also elaborated the models of the CoFI methodology for one of the modules of the FARE end-effector. The first set of CoFI models was created by the engineer responsible for designing and programming the FARE software, while this second one was created by an independent person.

This paper is organized as following. Section 2 introduces the CoFI methodology. Section 3 presents the FARE end-effector. Section 4 illustrates the application of CoFI to the partial version of FARE end-effector, discusses the main errors and summarizes the lessons learned. Section 5 draws some conclusions.

II. THE COFI METHODOLOGY

The CoFI methodology [1] consists of a systematic way to create test cases for software or system. The CoFI is comprised of steps to identify a set of services. Each service is modeled as finite state machines (Mealy machines) from a black box point of view. The models represent the behavior of the system under the following classes of inputs: (i) normal, (ii) specified exceptions, (iii) inopportune inputs and (iv) invalid inputs caused by hardware faults.

The software behavior is represented by small models taking into account the decomposition in terms of: (i) the services provided by software and (ii) the types of behavior under the classes of inputs. The types of behavior defined in the context of the CoFI are: Normal, Specified Exception, Sneak Path, and Fault Tolerance. These behaviors are respectively associated to the following inputs: normal, specified exceptions, inopportune and invalid inputs. In this work, the fault tolerance behavior (related to invalid input) is not used. More than one model can be created in order to represent a type of behavior for a given service.

In this work, the application of CoFI makes use of the Condado tool [2] for the automatic generation of test cases. The Condado tool receives as input the state machine models and provides as output the test sequences.

The CoFI methodology has been traditionally applied to space embedded systems. One example is illustrated by Pontes et al. [3]. In this work, the CoFI is applied to the on-board data handling (OBDH) computer of a satellite. This work discusses issues related to the viability of applying the CoFI methodology, such as time for the system modeling, time for applying the test cases, number and severity of the errors detected by CoFI. For the OBDH software, the process of modeling, generating and applying the test suite spent forty hours. The main contributions of the CoFI methodology were the identification of critical error in the OBDH software and the improvement of the testing tool. The main limitations regard the combination of services, and the impossibility to assure code coverage.

III. THE FARE END-EFFECTOR

The FARE end-effector (Figure 2) was designed to automate the process of drilling, sealing and inserting fasteners in aircraft fuselage. The FARE end-effector contains a number of devices that operates in a coordinated way in order to perform a sequence of tasks.

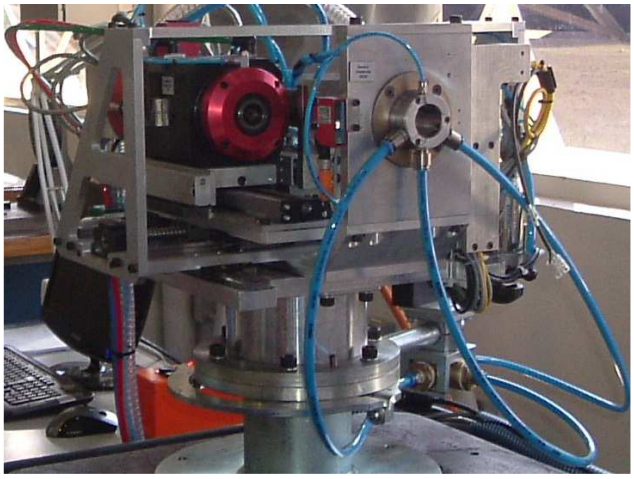

Figure 2. The FARE end-effector.

The devices of the FARE end-effector are organized in functional modules, which are illustrated in different colors in Figure 3. This organization is also reflected in the FARE control software.

The FARE modules are:

- Vision module: it uses a CCD camera to determine the error in the robot position related to a reference in the aircraft fuselage.
- Perpendicularity module: it determines the error in the robot orientation related to the fuselage surface.
- Positioning module: it corrects the robot position and orientation according to the errors calculated by the vision and perpendicularity modules.
- Clamp module: it pressures the FARE end-effector against the fuselage in order to avoid the formation of chips during the drilling operation.
- Drilling module: it performs the drill in the fuselage surface.
- Fastening module: it applies seal in the fasteners and inserts them in the drills.

- Mechanical platform: it integrates the FARE module and put them in necessary position to be used.

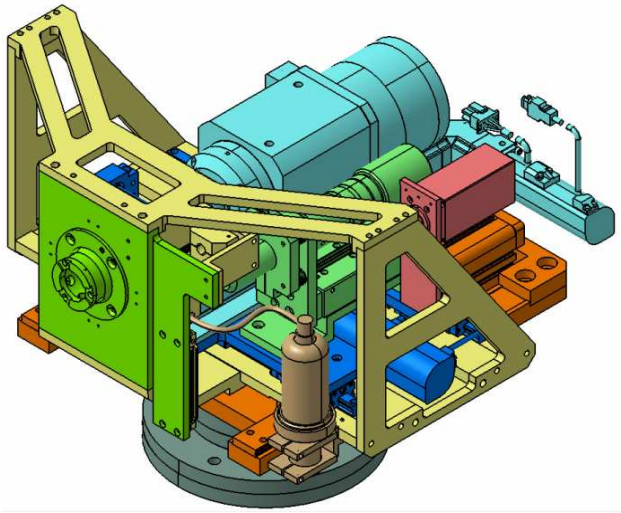

Figure 3. The FARE modules.

The hardware architecture of the FARE control system is composed of an industrial computer equipped with a set of interface boards and a motion control board. The communication with the industrial robot that holds the FARE end-effector is made via OPC protocol.

The software requirements are related to three modes of operations: manual, semi-automatic and automatic. All the three modes of operation must assure the safeness and integrity of both the FARE and the fuselage.

The manual mode of operation gives the control of the FARE devices to the user. It allows the user to activate each device individually. Examples of commands of the manual mode of operation are: turn on the spindle, capture an image with the CCD camera.

The semi-automatic mode provides to the user commands for performing the main functionality related to each module. Examples of commands of the semi automatic mode of operation are: make a drill, calculate the position error.

The automatic mode receives as input an assembly plan with the position of a set of references, drills and fasteners, as well as the corresponding parameters. The control software must then coordinate the operation of all modules in order to complete the assembly plan.

The FARE software is programmed in LabView. It uses a producer-consumer architecture that handles all the events generated by the user interface, the FARE sensors and the industrial robot. It coordinates the execution of the requested task with the appropriated priority.

IV. The Application of CoFI

A. *CoFI Modeling*

In order to make possible the application of the CoFI methodology, some of the FARE modules were selected for composing the partial version of the FARE software, following the approach illustrated in Figure 1.

The purpose of this approach is to apply the CoFI to these modules, compile a set of lessons learned and then apply the lessons learned to the others modules, trying to find similar errors. Although this approach reduces the reliability provided by CoFI, it is necessary due to the limited time and human resources available for the project.

The modules selected for the partial version are: drilling module, fastening module and mechanical platform. Each module is considered as a service. For the application of the CoFI methodology, only the semi-automatic and the automatic modes of operation are considered. The reason for not considering the manual mode is because it is basically composed by single events.

For each service, the state of the FARE devices are considered unknown when the command of the semi-automatic mode is requested. This indeterminism is necessary because the user can execute any manual command before entering in the semi-automatic or automatic mode.

Another important point is the definition of input events. In the case of the FARE software, the events can be either generated by the user or by the FARE devices. The inputs related to all the devices are read with a constant frequency. An event of the CoFI model may be characterized by a Boolean expression combining input values and events generated by the user interface.

As an example, this paper presents the models elaborated for the drilling module. Due to the limited space, it is not possible to distinguish each input and output event individually.

The normal behavior model (Figure 4) is composed of 25 states and 28 transitions. It is basically a sequence of commands. The specified exception model (Figure 5) is also composed of 25 states and 52 transitions. The exceptions considered in this model correspond to the interruption of the semi-automatic sequence, requested by the user via an emergency button.

The sneak path model is composed of 27 states and more than 100 transitions. Basically, when an input that is not expected occurs, the software has one of the following behaviors:

A) It ignores the input.
B) It detects a sensor fault and goes to a deadlock state from where the software cannot exit.
C) It detects a fault in the motion control board and goes to a fault state that requires the user to reset the motion control board.

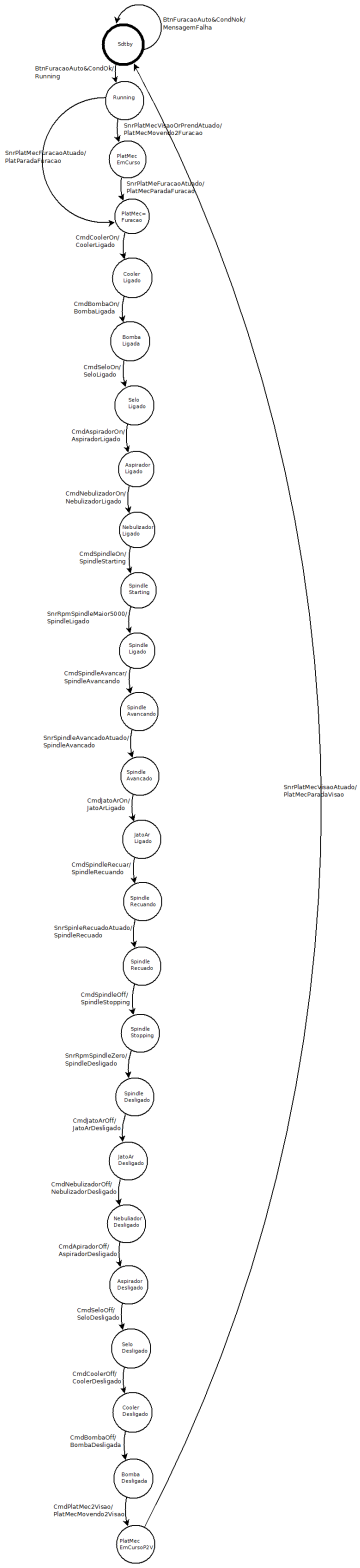

Figure 4. Normal behavior – Drilling module.

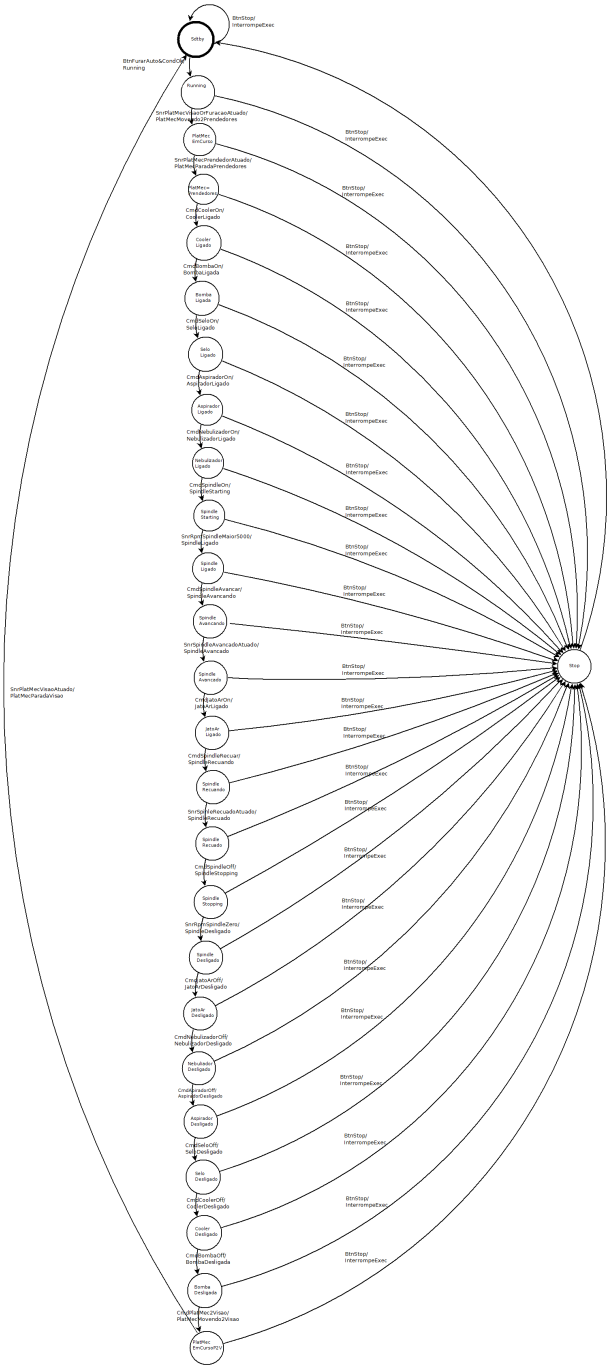

Figure 5. Specified exception behavior – Drilling module.

The CoFI modeling of the semi-automatic mode of the fastening module is divided in two services: (1) the sending of the fastener from a remote buffer to the FARE end-effector, and (2) the installation of the fastener. Each service has one state machine model for the normal behavior, one for the specified exception and one for the sneak paths.

The drilling module generated 119 test cases: 3 for the normal behavior, 26 for the specified exceptions and 90 for

sneak path. The fastening module generated 157 test cases: 7 related to the normal behavior, 33 for the specified exceptions and 117 for the sneak path. The mechanical platform is included in the test cases of the drilling and fastening modules, as it must position these modules for the correct execution of the drilling and fastening operations.

The CoFI modeling of the automatic mode of operation included only the events related to the mechanical platform, drilling, fastening modules. It is a composition of the sequence of each module. Each sequence of the semi-automatic mode is condensed in a single state. As a consequence, the automatic mode resulted in 2 test case for the normal behavior, 6 for the specified exception and 10 for the sneak path.

B. Lessons Learned from CoFI

The application of the test cases generated by the CoFI methodology resulted in the detection of a number of errors and in important contributions for the FARE control software.

Regarding the severity of failures detected by the CoFI methodology, some of the failures in the test cases were of low severity and were associated with an inappropriate behavior of the FARE devices. Other failures were of high severity and resulted in danger situations that may compromise the safeness and integrity of the manufacturing system.

Generally, the failures have two sources: (1) programming errors and (2) poor requirements regarding the behavior of the FARE control software in the case of failure of the FARE devices.

Most of the failures were associated with the test cases generated by the sneak path model.

One important observation is that a significant number of failures were resulted from systematic implementation errors that were repeated for all the semi-automatic sequences. This fact made possible to reuse the CoFI results to detect errors also in the implementation of the other modules that composed the full version of FARE control software.

The most important errors appointed by CoFI were:

- Error related to the control of the mechanical platform. Once the moving of the mechanical platform is started, it could not be halted by the emergency button. This error is due to the control loop of the servo positioning system. In order to stop the mechanical platform, FARE software must interfere in the motion control board. This was an error that were repeated in all the semi-automatic sequences.
- Error related to the interruption of a service. The tests appointed that when the emergency button is pressed, some devices requested the execution of a finalization routine in order to stop working in a safe configuration. Before the application of the CoFI methodology, the software simply interrupts the command to these devices, resulting in unsafe states. One example is fastener installation device, which must be retracted when the emergency button is pressed.
- The routine that implements the automatic mode of operation has a hidden fault. This routine has two path for positioning the mechanical platform: the first one when the mechanical platform is in the vision or drilling module, and the second one when it is in the fastener module. All the functional tests performed before the application of CoFI considered only the first path because of the initialization routine.
- Error in the implementation of the of the spindle movement. The signal from the spindle rotation sensor was not considered in the activation logic. This error allowed the spindle to be pressured against the fuselage without being turned on, resulting in damage to both the FARE and the fuselage.
- Problem in the detection of the fastener. Due to the mechanical design of the FARE, when the fastener is sent from the remote buffer, its detection is only possible in the receptor device. The analysis of the inopportune events leaded to the modification of the semi-automatic sequence in order to assure that the installation device has gripped the fastener.
- Errors caused by manual commands from the user interface when the semi-automatic sequence is under execution. The test cases generated by the sneak path models showed that some manual commands were still available and could cause damage to the FARE devices. In order to correct this problem, the entire graphical interface should be blocked during the execution of a semi-automatic or automatic sequence.
- Problems caused by faults in the FARE devices. The malfunctioning of the FFARE devices generated inopportune inputs, which were not adequately treated by the FARE software and could result in erroneous output to the FARE devices.

C. Influence of the Human Factor

The CoFI methodology was applied to the FARE end-effector by the engineer that designed and programmed the FARE software. In order to verify the influence of the human factor in the results of the CoFI methodology, a second set of CoFI models was created by an independent person for the drilling module.

The following results were obtained in this experiment. The models related to the normal behavior were similar. The model created by the designer had three additional states not considered by the independent person.

The two versions of the specified exception model considered different exceptions. While the designer modeled the effect of the emergency button, the independent person considered the overheating of the spindle, which was not in the designer models.

V. CONCLUSIONS

This paper discusses the application of the CoFI testing methodology to a multifunctional end-effector used in the assembly of aircraft fuselage sections.

The CoFI testing methodology has been originally proposed to space systems, which can be defined as critical applications. This paper discusses its applicability to industrial automation products. Differently from space systems, industrial automation products are usually submitted to functional tests defined according to the expertise of the developer. The contribution of this paper is on the analysis of the applicability of a model based testing methodology.

In this paper, a number of simplifications are introduced in order to adequate it to an application of lower criticality. The most important simplification is the selection of some of the end-effector modules (drilling, fastening and mechanical platform). The results of the application of CoFI to these modules were compiled in lessons learned and were used to revise and correct the software of these modules. Furthermore, the lessons learned were also applied to the other modules of FARE, allowing the detection of many problems and errors without the need of extensive modeling. This point suggests that a developer/programmer tends to commit similar errors. A possible application of the CoFI methodology is on the training of developers/programmers.

Another important simplification introduced in the CoFI methodology is the exclusion of the manual mode of operation from the list of services. Because of this simplification, the state of the FARE devices are considered unknown when a semi-automatic or automatic sequence of operations is performed.

The influence of the human factor was analyzed by elaborating two versions of the CoFI models for one of the FARE module. The first version was elaborated by the designer of the software, while the second one was elaborated by an independent person. The comparison showed that both models were incomplete and considered different specified exceptions and inopportune events. Even thought, both versions were able to detect the errors described in this paper.

As a general conclusion, this work shows that the CoFI can contribute to industrial automation products. However, differently from the space area, the methodology can be simplified (or “relaxed”). One of its most important contributions is on training the developers to think about exceptions and inopportune events.

REFERENCES

[1] AMBROSIO, A. M. **CoFI – uma abordagem combinando teste de conformidade e injeção de falhas para validação de software em aplicações espaciais**. Tese (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos. 2005.

[2] MARTINS, E.; SABIÃO, S. B.; AMBROSIO, A.M. ConData: a Tool for Automating Specification-based Test Case Generation for Communication Systems. **Software Quality Journal** v. 8, n. 4, p. 303-319, 1999.

[3] PONTES, R. P. et al. Embedded Critical Software Testing for Aerospace Applications based on PUS. In: XI Workshop de Testes e Tolerância a Falhas, p. 119-131, 2010.

[4] AGUIAR, A.J.C. Integração de rede de Petri e simulação gráfica para verificação de células robóticas colaborativas. Tese (Mestrado) – Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos. 2009