The Role of Software Testing on Modernizing a Balloon Ground Station

Fátima Mattiello-Francisco[1], Mariam Gomez[1], William K. Ariyoshi[2], Fernando Aranha[2], Marcelo Essado[3]

[1]National Institute for Space Research (INPE)
Caixa Postal 515 – 91.501-970 – São José dos Campos – SP – Brazil

[2]Beta/TELECOM
São José dos Campos – SP - Brazil

[3]EMSISTI- Sistemas e Soluções em Tecnologia da Informação
Franca – SP - Brazil

{fatima.mattiello,mariam.gomez}@inpe.br, fgaranha@uol.com.br,
wkariyoshi@gmail.com, marcelo.essado@emsisti.com.br

Abstract. *This paper reports the use of different software testing techniques and tools in a verification and validation (V&V) strategy for software and hardware improvements on an existing old generation balloon ground station. A model-based testing method was combined with both human machine interface and communication testing techniques to carry out service driven test cases specification. The strategy was effective on validating the behavior of the new flight control system (OPS/ES) integrated to the ground station and useful on verification of the server system (OPS/Server) on the tasks of routing the original ground station TM and TC channels to TCP/IP gateways.*

1. Introdution

Aiming at a cost reduction on the development and operations of the balloon-borne protoMIRAX telescope experiment, the system requirement for reusing the ground facilities available at INPE was a challenge to the ground system engineers.

Different age technologies were required to interoperate. The existing one decade-old balloon ground station needed little hardware maintenance, but significant effort on software updating was required. The improvements on INPE´s ground station comprise only investments on information technology. The standalone computer system available in the balloon ground station rack dedicated to run the flight control software was modernized both in hardware and software by new system named OPS/ES. In addition, a new server computer, properly configured for Ethernet connections, has extended the existing ground station facilities with a network switch, serial converters and a new software named OPS/Server in order to support the available uplink and downlink channels being mapped to TCP/IP gateways.

Both solutions was developed by only one supplier (Sup 1) and the acceptance phase leaded by costumer INPE at LabV&Vsis was supported by other company (Sup 2) as independent verification and validation (V&V) activities.

This paper reports the experimental results of the V&V strategy adopted for systematizing the OPS/ES and OPS/Server acceptance process. Section 2 presents the concept of service in the space system engineering and its use as a V&V strategy. Section 3 and 4 introduces the OPS/ES and OPS/Server respectively, the test bed and how the proposed V&V strategy was used. In Section 5, the V&V strategy is discussed in terms of the test cases set and fault detection capability, taking into account the role of the techniques and tools used for both the test cases specification and management of test execution and retesting cycles. Finally, the conclusions are presented in Section 6.

2. Service driven approach

In space system engineering, a **service** is a set of capabilities that a component provides to another component via an interface in order to associate the set of operations that can be invoked and performed through the service interface "Service specifications define the capabilities, behavior and external interfaces, but do not define the implementation". [CCSDS 2010]

In the context of V&V, the verification of the expected behavior of a system and the interfaces at system level (black box testing) can be oriented by the behavioral models of a particular service provided by one or more reactive components [Mattiello-Francisco at al. 2013]. The service defines the set of events accepted by the component at its available external interfaces which stimulate the component actions, performing the expected behavior. Thus, the expected behavior of a service at a particular abstraction level can be represented by the automata formalism, where the states of the automata represent the actions and the events are the state transitions.

Some model-based testing (MBT) approaches use such formalism to derive test cases from behavioral models of the system under testing (SUT). A test case is one path of the SUT model as result from a traverse algorithm implemented by a tool. In the last decade there are many contributions of MBT techniques, methodologies and tools in order to automatically generate test cases specification for reactive system [Shunkun at al. 2013]. In particular, the concept of service according to CCSDS (2010) can be found in two MBT methodologies developed at INPE and operational in LabV&Vsis to guide the construction of behavioral models of the space SUT [Ambrosio 2005] and [Mattiello-Francisco at al. 2012].

The proposed service driven approach is a V&V strategy that use automatic and manual testing techniques, for test case specification aiming at validating SUT aspects in terms of human machine interface (HMI), communication functionality and control actions. Conformance and Fault Injection (CoFI) [Ambrosio 2005] is a MBT methodology used to guide automatic test cases generation focusing control actions.

The TestLINK tool is used to supporting the management of the **Test Case** suites for OPS/ES e OPS/Server software acceptance testing phase as a **Test Project** manager [http://testlink.org/]. The TestLINK supports the SUT description and definition of **Testers** (users) involved on testing activities**. Test Plans** are built describing both the **Services** to be verified using each testing technique and the testing environment (Test bed) required. For each test case execution **Build** memorizes the **Test Case Result** (Passed, Failed, Blocked) providing testing management metrics and supporting **Report**.

3. Ground Balloon Flight Control System - OPS/ES

On the context of balloon ground station modernization for protoMIRAX campaigns operation purpose, the OPS/ES system substitutes the obsolete ground station standalone computer. As ground station equipment's integrator, this computer major functionality is to support the ground station operator on balloon flight control.

The new software OPS/ES was designed using LabVIEW 2013, National Instruments™ [http://www.ni.com/labview/pt/], and it is the integrator element of the ground station and on-board system equipment. It enables flight control of the balloon by sending telecommand and visualization of the flight parameters received by telemetry. These parameters are displayed as real-time values, status indicators, leds and graphics. OPS/ES also creates logs of all telemetry and telecommand information since the beginning of the campaign. In addition, OPS/ES has a tracker function to always keep the antenna pointed to the balloon to prevent connection losses. For this purpose it uses the GPS antenna placed in the on-board system. It also has a mapping function using a Google Maps API that tracks the balloon's flight path.

OPS/ES software code size is approximately 1563KB and runs in an industrial rack mounted computer whose hardware configuration is: 300W Power Supply; Motherboard ATX-SB600C with one PCI-Express x16 slot and six PCI-32bits slots; Processor Intel® Core™ i5-2400; RAM Memory Kingston 4GB DDR3 1333MHz; HDD Seagate 500GB; DVD-RW. OPS/ES has four RS232 serial cables to communicate with other ground station equipment: MUX/DMX – Telemetry; MUX/DMX – Telecommand; Antenna Positioning Unit; and RF Decoder.

3.1. OPS/ES Test Plans, Test Cases Specification and Test bed

Two Test Plans were elaborated to carry on the testing activities related to OPS/ES acceptance: (a) the Test Plan guided by Human Machine Interface Testing Technique, named **OPS/ES Interface TP**; and (b) the Test Plan guided by MBT using CoFI methodology, named **OPS/ES Control TP**.

The Test bed was composed of the OPS/ES software itself embedded in the Ground Station standalone computer, the MUX/DMX chains and channels for ground-space communication. In addition, a software simulator provided by SUP 1 simulated the telecommand and telemetry functions on board of the balloon.

Based on Inputs and Outputs available to control and observe OPS/ES operation and the software design artifacts such as user manual provided by SUP 1, an abstract view of the SUT, as a black box, allowed the identification of 4 services associated to the human interface and 3 services associated to control, as summarized in Table 1.

The OPS/ES Interface TP comprises the verification of the following 4 services **SI-1** – TM: all **telemetry** data visualized by the Flight Control Operator (**OPS/ES operator**) on OPS/ES display, as output; **SI-2** – TC: all input (buttons and parameters) and output event (led) available to the OPS/ES operator for **telecommands** selection and transmission to the balloon flight control on board; **SI-3** – RH: on time **recording** on **Historical** file all telecommands sent by the OPS/ES operator and all telemetry data received by OPS/ES software during the campaign; **SI-4** – IC: all OPS/ES operational parameters available for OPS/ES operator setting as **Initial Configuration**.

Two sets of test cases were manually specified for each **SI**, taking into account the inputs variables provided by OPS/ES software to **OPS/ES operator** performing the service. One set, referred as *Normal* (**N**), comprises OPS/ES expected behavior on the service execution. For instance SI-1: telemetry value within the specified range shall be displayed in the graphical interface and those out-of limit values shall be signalized with visual warnings. The other set comprises *Robustness* (**R**) aspects, for instance, wrong inputs. The number of test cases (N) and (R) per service related to **SIs** is presented in Table 1. The expressive number of (N) test cases compared to (R) is due to the quantity and nature of HMI screen components: (i) all output variables are displaying in graphical screens using LabView libraries in SI-1; (ii) input variables are mostly represented by buttons and parameters values already defined for OPS/ES operator selection in SI-2; (iii) data log needs to be tested only in terms of timestamp and delay in SI-3; (iv) only one input variable (Antenna Position) is open in SI-4 for OPS/ES configuration by means of writing localization coordinates, which required one (R) test case specification to validate the OPS/ES robustness in terms of invalid parameters.

The OPS/ES Control TP comprises the verification of the OPS/ES software behavior on the following 3 services: **SC-1** – TM: the tasks of acquiring and displaying on OPS/ES graphics accordingly all **telemetry** parameters related to the different equipment integrated in ground station; **SC-2** – TC: the tasks of both sending **telecommands** (sequence of buttons and parameters verified in **SI-2**) under selection of OPS/ES operator to the on board balloon flight control system and feed backing on screen telemetry data related to each telecommand effective execution on board for monitoring purpose on ground; **SC-3** – FE: the task of **Flight Ending**, transferring all files recorded by OPS/ES during the campaign to the mission historical data center.

The specification of test cases for these services followed three main steps of CoFI methodology: (a) the construction of a set of Finite State Machine (FSM) that models the expected behavior of the *services* provided by SUT, (b) models validation and (c) *abstract* test-case generated automatically using Condado tool [Martins 1999]. According to CoFI, a *service* behavior is modeled in different perspectives: (i) normal, (ii) specified exceptions, (iii) inopportune inputs (i.e., corrects but occurring in wrong moments) and (iv) invalid inputs caused by hardware faults. The models are generally small because two levels of decomposition are taken into account: (i) the services provided by system under test and (ii) the *types of behavior*, which are named as: *Fault Tolerance*, *Sneak Path*, *Specified Exception* and *Normal*, respectively associated to the input events: invalid, inopportune, specified exceptions, normal. Moreover, it is possible to create more than one model to represent the same type of behavior of a service. The selection of the inputs to be considered in the models must take into account the controllability and observability available in the test executing tools (or test environment). Thus, the test environment has to provide mechanisms for input events and observation of the system outputs.

For sake of space, the FSM models built for the control services are not presented. Table 1 summarizes the number of models built per service resulting from only two *types of behavior*: *Normal* (**N**) and *Sneak Path* (**R**) which covers **robustness** behavior related to wrong inputs. *Exceptions* models were not built because exception behaviors were not specified in OPS/ES software requirement document. From those

models, *abstract* test-case suites were generated automatically by Condado tool. One can observe in Table 1 a huge number of test cases generated per suite, in particular from (**R**) models. The reason is the combination of input events (transitions) that are added to each state of the nominal model in order to represent all unexpected input variables. A test selection was necessary and manually performed by test expert analyst using two criteria: (i) discarding all test case that comprise a step sequence already included in other test case, considering as duplicated test case or similar; (ii) choosing only one representative value in domain for each input variable, since most input and output variable were already covered by the test cases specified in Interface Test Plan. The numbers of test cases selected and effectively translated to *executable* test-cases for SCs are presented by numerators in cells of Table 1 (three columns corresponding to **SCs** on last two lines).

4. Gateways Mapping System – OPS/Server

The purpose of the software OPS/Server is to provide TCP/IP gateways to the payload controllers and scientists (mission end users) for mission operation. Protomirax's board and ground system originally uses asynchronous RS232 and synchronous RS422 communication. These channels are routed respectively to Ethernet converter and to USB converter in order to support OPS/Server with TCP/IP connections.

OPS/Server software was designed using LabVIEW and its code size is approximately 241KB running in a computer configured with W7 Professional, Processor Intel® Core™ i7-3770; RAM Memory 8GB; HDD 1TB; DVD-RW.

4.1. OPS/Server Test Plan, Test Cases Specification and Test bed

Only one Test Plan, named **OPS/Server Communication TP,** was elaborated to carry on the testing activities related to ground-ground communication protocols that were properly defined to support the OPS/Server routing functions.

The service driven approach was also useful for OPS/Server test case specification that comprised the verification of the ground-ground communication protocols involved on 3 services: **SM-1** – TC: the task of sending direct **telecommands** from TCP/IP gateways to particular Ground Station channel; **SM-2** – TM: the task of transmitting internally in ground all **telemetry** packets received from different ground station channels to the end users; **SM-3** – TMTC: the tasks of sending telecommands and receiving telemetry simultaneously, covering **SM-1** and **SM-2** execution together.

Test cases were manually specified for **SM**s addressing three main aspects: message format, communication faults and performance of the routing task.

5. Testing Results and Discussion

In total, **125 test cases** (N) and (R) were effectively executed, as presented in Table 1 by the cells numerators (last two lines). The corresponding denominators show the number of test cases Failed (9%) and/or Blocked (14%) during first cycle of testing. Blocked test cases were due to (i) lack of on board simulator functionalities or (ii) misunderstandings on test case specification. Test cases were rewritten and functions added to the simulator improving the Test bed to support the second cycle of testing, which test cases Passed.

Table 1. Numbers of Models, Test Cases generated and executed per service

V&V strategy		**OPS/ ES**							**OPS/ Server**			**Total**
		Interface Test Plan				Control Test Plan			Communication TP			
Services		SI-1	SI-2	SI-3	SI-4	SC-1	SC-2	SC-3	SM-1	SM-2	SM-3	**10**
#Models	N	-	-	-	-	3	1	2	-	-	-	**6**
	R	-	-	-	-	3	1	1	-	-	-	**5**
#Test Cases	N	36	26	7	1	51	13	41	2	5	1	**183**
	R	0	0	0	1	832	978	672	2	3	0	**2527**
#Execut	N	36/11	26/2	7/4	1	5/1	6/5	2/1	2	5/1	1/1	**91/26**
	R	0	0	0	1/1	4	19	5	2/1	3/1	0	**34/3**

The cells highlighted in gray show that most failures detection capability of the Test Plans are concentrated in two services: (i) Interface TP pointed out 6 failures, 3 detected by SI-1 test cases and all solved with the above mentioned Test bed improvements; (ii) Control TP pointed out 3 failures, all detected by SC-2; (iii) no failure was detected by Communication TP. Two failures detected in SC-2 were due to lack of on board hardware in the loop. Only one failure was in fact software fault that was corrected.

6. Conclusion

The V&V strategy demonstrates that: (1) focus the model-based testing efforts on the control services is cost-effective whether others testing techniques can be used to supplement the whole software product verification; (2) the correspondence between test cases specification and the resources available on the test bed for test cases control and observation is essential to reduce testing effort and risk.

7. References

Ambrosio, A. M.. “COFI: uma abordagem combinando teste de conformidade e injeção de falhas para validação de software em aplicações espaciais.”, 2005, 209 p. (INPE-13264-TDI/1031). Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2005

CCSDS-520.1-M-1. Mission Operation Reference Model. “Recommended for Space Data Systems Practice”. (July 2010)

Mattiello-Francisco, F.; Villani, E.; Martins, E.; Dutra, T.; Coelho, B.; Ambrosio, A. M.; “An Experience on the Technology Transfer of CoFI Methodology to Automotive Domain”; LADC2013 – Sixth Latin-American Symposium on Dependable Computing, Industrial Track – Rio de Janeiro 2-5 Abril 2013.

Martins, E.; Sabião, S.B.; Ambrosio, A.M. - "*ConData:* a Tool for Automating Specification-based Test Case Generation for Communication Systems". In: Software Quality Journal, Vol. 8, No.4, 303-319, 1999.

Mattiello-Francisco, F.; Martins, E.; Cavalli, A.R.; Yano, E.T.; “InRob: An approach for testing interoperability and robustness of real-time embedded software.” In: Journal of Systems and Software, 85,1, January 2012, 3-15.

Shunkun, Y.; Bin, L.; Shihai, W.; Minyan, L.; “Model-based robustness testing for avionics-embedded software.” In: Chinese Journal of Aeronautics and Astronautics, 26(3), 2013, 730-740.