The Dependability Approach for the New Version of VLS On Board Software

Elmer Dotti - Carlos H. N. Lahoz
Instituto de Aeronáutica e Espaço - IAE
elmer.dotti@gmail.com - lahozchnl@iae.cta.br

Abstract

This paper aims to present the dependability approach proposed for the new version of the embedded software of the Brazilian Satellite Launch Vehicle (VLS). It is proposed a process whose activities cover, in an integrated manner, the techniques FTA, FMECA and HAZOP to obtain a set of compensating provisions that may become non-functional requirements to be incorporated into the embedded software system. This work, still in the beginning phase, is part of a plan that covers the verification and validation (V&V) activities, to be executed for the next release of the VLS software. The example purpose in this paper is to show how the process works and expects to attend the software quality assurance of critical mission applications in the space area.

1. Introduction

The Brazilian Satellite Launch Vehicle (VLS) is a small rocket designed to launch satellites for environmental data collection and remote sensing. It is a conventional solid propulsion rocket, 19 m long, weighing 50 ton, designed to insert satellites of 100-350 kg into circular low orbits of 250-1000 km.

The VLS-1 onboard software (SOAB) implements the navigation, guidance–loop compensation and control systems algorithms to assure that the vehicle accomplish its mission. Moreover, during the powered and ballistics phases, the software is responsible for assuring a sequence of events that characterizes the several phases the vehicle must accomplish, and also for the data packing to be sent to ground stations by telemetry.

In order to identify possible problems related to software reliability, availability, maintainability and safety, a Verification and Validation (V&V) plan was defined by IAE and Critical Software [1], proposing dependability activities for the SOAB. The activities envisaged in this plan aims: to identify, classify, reduce and manage the software hazards, to analyze the failure causes, their effects and severities according specific criteria and provide compensating provisions to the software that may become non-functional requirements (NFR).

2. Context of this work

This paper addresses the dependability analysis that should be made by the software team of the Aeronautics and Space Institute (IAE) inside the verification and validation activities planned for the next VLS software version. For many years the dependability activities have been neglected by the software team, in favor of other activities related to code development. Besides the dependability activities proposed (also known as RAMS - Reliability, Availability, Maintainability and Safety) the intent of the V&V plan is to conduct project documents verification, code inspection, analysis of possible software hazards and software validation by a set of comprehensive test cases.

Space software has specific characteristics that impose special challenges for verification and validation activities. These systems are often connected to specialized interfaces and dedicated equipment like the attitude control systems that uses gyroscopes, sensors and actuators of various types.

In the space environment, obtaining flight data is subject to unforeseen circumstances and the test an analysis of the data acquisition systems must consider the various possibilities of failure that can cause these conditions. Conflicts of time, for example, are of great concern, demanding that these systems go through a rigorous verification and validation process.

Dependability techniques such as hazard analysis and analysis of failure modes, among others, are applied to identify common causes of failures, performance problems and hazards arising mainly from dysfunctional interactions between the vehicle system components.

Thus, a way to identify the more critical computer system components that should receive compensating provisions, IAE software team proposed to use integrated safety analysis techniques to identify, prevent, tolerate and also remove weaknesses in the software project.

The application of such techniques will therefore result in recommendations (compensating provisions) that will guide the inclusion of new requirements, both functional and nonfunctional in order to improve the software reliability and assure the mission accomplishment.

3. The dependability approach

The proposed approach is based on the integrated use of safety analysis techniques, aimed at identifying and correcting potential failures of systems relying on software. Three techniques are used for SOAB dependability analysis: SFTA (Software Fault Tree Analysis), SFMECA (Software Failure Mode, Effects and Criticality Analysis) and SHAZOP (Software Hazard and Operability Studies) [2] [3] [4]. The proposed approach is divided into four distinct activities:

1) Preparation for dependability process application: this activity defines the accessories items used to apply the process, like the severity classification, their consequences, the probability of failure, the generic failure modes, and possible causes of failure. The generic failure modes typically include the incorrect function execution, function not implemented and function running out of time.

2) SFTA analysis: a fault tree is designed based on the possible failures in the system requirements for software and the consequential failures for the respective software requirements. In this approach the top event is characterized by failure of the system requirements for software (system software specification) and the basic events are characterized by software failure in meeting system requirements (software specification). Thus, SFMECA is applied in the SFTA basic events, identifying potential failure modes, consequences and possible compensating provisions.

3) SFMECA application: the main objective of the SFMECA analysis is to classify the software requirements in terms of potential failures modes, severity and criticality in order to be able to identify ways to minimize the risks of possible failure (compensating provisions) especially for the critical and catastrophic severity. In this approach, SFMECA is applied in the SFTA basic events (software requirements), and HAZOP guidewords are used to classify generic failure modes, taking into account the deviation of design intent related to physical (equipment) or logical (flow of data) devices, that have a relationship with the system component analyzed [5].

4) Identify new requirements: from the compensating provisions extracted by SFMECA analysis, new functional and NFR could be suggested and incorporated into SOAB. The idea is to use the set of dependability attributes selected for space computer system discussed by [5] [6]. These attributes are based on international and Brazilian institutions standards (ABNT, UK Ministry of Defense, ESA, and NASA), as well as studies related to the dependability of some important authors in this area.

In addition to the dependability analysis, other studies will be conducted to verify the feasibility of new NFR proposed, using formal methods and fault tolerance techniques [7].

4. The application process

In order to explain how the process should be applied, a practical example based on the SOAB pre-flight requirement is presented.

These “initiate on-board system” system requirement function includes the requirements for initiating the SOAB and the requirements for initiating the Inertial Platform. The requirements associated with “initiate SOAB” are “verify auto-test results”, “initialize real-time executive” and “prepare the hardware”. The requirement associated with the “initiate Inertial Platform” comprises basically “alignment the equipment”, “calibrate the equipment” and “send data to telemetry to Ground System”.

4.1 Preparation for process application

At first, a generic functional failure modes table, was created, as shown in Table 1

Generic failure mode	Possible causes of failure
Function not implemented	Computations unrealized; Results are not produced.
Function performed incorrectly	Entries unavailable; Erroneous outputs; Wrong function performance.
Function run out of time	Timing issue; Unexpected latencies; Performance Issues.

Table 1 – SOAB generic failure modes

After that, it was defined the severity levels of the failures which were used in the dependability analysis, having as a reference the classification of NASA severity categories: Catastrophic, Critical, High and Minor level [8].

4.2 SFTA analysis

The fault tree is constructed initially from the identification of system requirements for SOAB. The top event of the tree is a possible failure arising from a non-compliance with this system software requirement. It explores then the failures in the software not meeting the system requirements - breaking down the functional requirement failures at several levels. For example, Figure 1 shows the decomposition of the failure related to the system requirement entitled “fail to initiate on-board system”.

In this paper example, the “fail to verify auto-test results” was chosen for analysis via SFMECA.

The main goal to build the SOAB fault tree is to quantify functional and nonfunctional requirements in a way to identify the failure probability of critical items, based on [9].

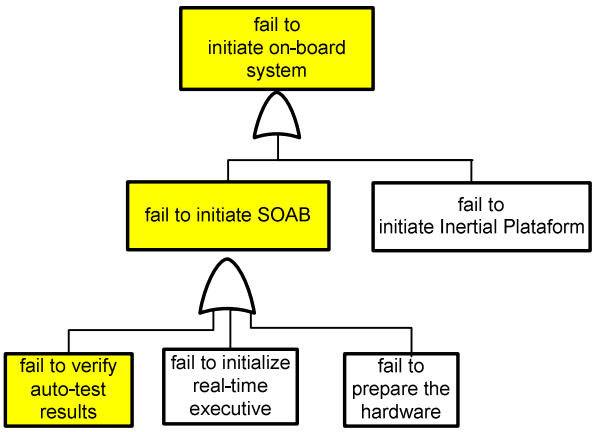

Figure 1 - FTA requirement of "failure to initiate on-board system"

4.3 SFMECA analysis

The SFMECA analysis is applied in SFTA basic events, considering the lowest level of SOAB software requirements. From these requirements the software design was built and the subsequent identification of their logical functions was made.

Through this approach is possible to discover functional software failures and suggest new non-functional requirements to assure the software mission.

Table 2 presents the SFMECA analysis for the basic event “fail to verify auto-test results” only for the failure mode "function not implemented".

The final phase of SFMECA provides recommendations to mitigate the cause of failure and assign severities levels to all failure modes. The severity is based on the worst effects, and failure modes are defined considering the malfunction assigned to the project.

The compensating provisions suggested for the “fail to verify auto-test results” may recommend changes to the architecture of software functionalities.

In the example shown in Table 2, the compensating provision for the failure cause entitled "computation not performed” and “results are not produced” implies that “the system must be remain in the verification state”.

In this case, no change to system architecture or software is mandatory, but it may need to take actions such as restarting the computer to retry the execution of the computer interfaces checks and the hardware initialization procedures.

Table 2 - Part of the FMECA table requirement "fail to verify auto-test results"

Failure Mode	Cause of Failure	Local Effect	Seve-rity
Function not implemen-ted (Results not produced)	Computation not performed. Checks of unrealized income	Function ends (abruptly) without performing the necessary checks to produce the function result Results are not produced.	High
	Results are not produced Although they performed the necessary checks, the results are not produced.	Function ends (abruptly) despite having carried out the checks necessary to produce the function result. The state variable ESTADO_AUTO _TESTE remains un determined. Results are not produced.	High

5. Conclusions

The SOAB dependability analysis, still in conception phase, is presented as a practical and methodological approach, identifying recommendations or compensating provisions that can generate new NFR. The authors believe that this approach could improve the software quality and safety.

It is intended to verify the NFR proposed using formal languages and fault tolerance techniques, developing software system models and verifying their logical properties [7].

It is highlighted that the special characteristic of this approach is the integrated use of two techniques for software dependability analysis - SFTA to identify gaps in meeting requirements - and SFMECA - to analyze the software requirements failures in its most detailed level. The compensation provisions resulted should help in obtaining non-functional requirements for subsequent verification through model checking [7].

The SOAB dependability analysis proposed here is still in its early stages, and the expected results are:

- significant advancement in knowledge of dependability and V&V techniques by the SOAB development team;
- a set of recommendations to improve the dependability and V&V practices for VLS software;
- a collection of more reliable models, architectures, and test components for using in future SOAB versions;
- testing the proposed V&V methodology based on dependability analysis and formal techniques, ready to be used in the Software Engineering Laboratory (LES) of the Electronic Division at IAE.

As future work we intend to implement the use of DEA (Data Envelopment Analysis) mathematical modeling to quantify the functional and nonfunctional requirements obtained of the dependability approach proposed in this work.

6. References

[1] Plano de Verificação e Validação do Software Aplicativo de Bordo do VLS-1, Critical Software- IAE, São José dos Campos, 2010. 104 p.

[2] ESA Guide for Independent Software Verification and Validation Technical Note, 2005.

[3] European Cooperation for Space Standardization. Draft ECSS-Q-80-03: Space Product Assurance – Methods and Techniques to Support the Assessment of Software Dependability and Safety. Noordwijk, 2006.

[4] Redmill, F.; Chudleigh, M.; Catmur, J. “System safety: HAZOP e software HAZOP”. Sussex: John WILEY, 1999. 248 p.

[5] Lahoz, C. H. N. “ELICERE: O Processo de Elicitação de Metas de Dependabilidade para Sistemas Computacionais Críticos”, Doctorade Thesis, Universidade de São Paulo, S. Paulo, 2009. 225 p.

[6] Romani, M. A. S. “Processo de Análise de Requisitos de Dependabilidade para Software Espacial”, Master Thesis, Instituto Tecnológico de Aeronáutica, São José dos Campos, 2007. 194 p.

[7] Projeto de Verificação e Validação de Sistemas de Software para Projetos Espaciais, CNPq-MCT-AEB 33/2010, Process 559973/2010-1, 2010.

[8] National Aeronautics and Space Administration, “Software Safety Guidebook”, NASA-GB-8719.13,2004.http://www.hq.nasa.gov/office/codeq/doctree/871913.pdf.

[9] Reis Filho, J. V. B. Uma Abordagem de Qualidade e Confiabilidade para Software Crítico, Master Thesis, Instituto Tecnológico de Aeronáutica, São José dos Campos, 1995. 177 p.

These dependability activities had the institutional support of CNPq given through research grant #559973/2010-1