IOP Conf. Series: Materials Science and Engineering **351** (2018) 012001 doi:10.1088/1757-899X/351/1/012001

Towards generating ECSS-compliant fault tree analysis results via ConcertoFLA

B Gallina[1] Z Haider[1] and A Carlsson[2]

[1] IDT, Mälardalen University, Västerås, Sweden
[2] OHB Sweden, Stockholm (Kista), Sweden

barbara.gallina@mdh.se

Abstract. Attitude Control Systems (ACSs) maintain the orientation of the satellite in three-dimensional space. ACSs need to be engineered in compliance with ECSS standards and need to ensure a certain degree of dependability. Thus, dependability analysis is conducted at various levels and by using ECSS-compliant techniques. Fault Tree Analysis (FTA) is one of these techniques. FTA is being automated within various Model Driven Engineering (MDE)-based methodologies. The tool-supported CHESS-methodology is one of them. This methodology incorporates ConcertoFLA, a dependability analysis technique enabling failure behavior analysis and thus FTA-results generation. ConcertoFLA, however, similarly to other techniques, still belongs to the academic research niche. To promote this technique within the space industry, we apply it on an ACS and discuss about its multi-faceted potentialities in the context of ECSS-compliant engineering.

1. Introduction

Attitude Control Systems (ACSs) play an important role within satellites. More specifically, ACSs contribute to maintaining a certain attitude (i.e., orientation of the satellite in three-dimensional space). Controlling the attitude is necessary to enable satellites (space crafts, which, typically, orbit earth [1]) to accomplish their mission including the fulfillment of their pointing requirements, often referred to as pointing modes [2].

ACSs need to be engineered in compliance with ECSS standards and need to ensure a certain degree of dependability. Thus, dependability analysis is conducted at various levels and by using different techniques. Fault Tree Analysis (FTA) is one of these techniques. This analysis technique is being automated within various Model Driven Engineering (MDE)-based methodologies. The tool supported CHESS-methodology [3] is one of them. This tool-supported methodology incorporates ConcertoFLA [4], a dependability analysis technique enabling failure behavior and propagation analysis and thus FTA-results generation. ConcertoFLA, however, similarly to other techniques, belongs to the academic research niche. To promote this technique within the space industry, extensive explorations are needed. In this paper, we contribute to perform such explorations. More specifically, we exploit the tool-supported CHESS methodology to model and analyze (via ConcertoFLA) an ACS. We also discuss how analysis results can be used to generate ECSS-FTA compliant analysis results.

The rest of the paper is organized as follows. In Section 2, we provide background information. In Section 3, we provide an overview of the entire approach. In Section 4, we focus on the modelling of our ACS. In Section 5, we perform the analysis. In Section 6, we elaborate on FTA generation, based on ConcertoFLA analysis results. In Section 7, we discuss related work. Finally, in Section 8, we present our concluding remarks and future work.

2. Background

In this section, we present the background information on which we base our work.

2.1. ACSs in Sun Pointing mode

As mentioned in the introduction, ACSs control the satellite's attitude. This control-related functionality is relative to a frame of reference depending upon the pointing requirement, which is either mission mode or safehold mode [2]. The former refers to the main objective of the satellite and latter contributes to fail safe. For instance, a sun pointing mode could be a safehold mode for a satellite pointing its solar arrays towards the sun to power the critical parts of the satellite. In sun pointing mode, the attitude of a satellite, relative to the sun, is maintained by controlling the torques applied to the satellite by actuator thrusters. A sun sensor measures the angle of the sun beam, in sensor's reference frame, when the beam hits the sensor. The direction of the sun is acquired from this angle and fed into the control system as an input. The control system calculates the torque based on minimizing the pointing error function.

Typically, an ACS (focus on software) is a composite component composed of the following four software components: (1) PDController takes the Sun direction and angular velocity of the satellite in three axes as an input, computes the pointing error and calculates proportional and derivative torques. (2) SteerController also computes the proportional torque by minimizing the pointing error. But, the objective is to compute relatively greater torques for faster convergence to target pointing position. This is important in cases when the satellite is significantly diverging from the desired position e.g., when it is upside down. (3) FeedforwController computes additional torques to achieve the equilibrium state and stabilize the satellite body. A satellite in the space is subject to the disturbance torques coming from outside of the boundary of the satellite. For example, the Sun radiation pressure, gravitational pull of the celestial bodies and the earth atmospheric pressure etc. FeedforwController takes these torques as an input and calculates a complementary torque to counteract the disturbances. Typically, these disturbances torque are calculated by the angular rates of the satellite measured a gyroscope sensor. (4) TorqueSelector takes the torques computed by the above-mentioned controllers, and, based on the current attitude of the satellite, allows a specific torque to be applied on the satellite through the thrusters. If the satellite is oriented 180 degrees opposite to the desired pointing direction, a rapid change in attitude is required. Hence, relatively greater torques, which are computed by SteerController, are applied. On the other hand, when the satellite is closer to the desired pointing direction, rather a seamless convergence and stabilization is of interest. Therefore, smaller torques and stabilization torques, computed by PDController and FeedforwController respectively, are applied to satellite.

2.2. ECSS-compliant FTA

ECSS-Q-ST-30 [5], the ECSS standard targeting product assurance, lists the methods for performing dependability analysis, required at all levels of the space system. Fault Tree Analysis (FTA) is a top down approach to assess the reliability of a system, by analyzing the failure state and constructing a probable cause tree. The objective of the analysis is to "ensure that the design conforms to the failure tolerance requirements for combinations of failures" [5]. According to ECSS-Q-ST-30, a system/sub-system level analysis is required to identify the combination of the events that lead to a failure e.g. loss of mission [5]. The end consequence of a failure is defined as the severity, and could be catastrophic, critical, major and minor. The severity level of a failure together with the probability of its occurrence defines the criticality of the event. The severity level of the system in consideration i.e., ACS in Sun pointing mode is classified as critical, since the consequence is loss of the mission.

ECSS-Q-ST-40-12C [6] provides the guidelines for performing ECSS-compliant FTA, required for all the systems/sub-systems with severity level catastrophic, critical or major consequences. For procedural guidelines of performing FTA, ECSS-Q-ST-40-12C refers to IEC-61025 [7]. Additionally, it is suggested that Fault Tree Handbook [8] can be used to complement the standard. IEC-61025 uses general terms to describe FTA analysis, steps to perform the analysis, list of assumptions and standardized shapes and symbols. IEC-61025 requirements are generic and are not limited to space domain. A compliance artifact, as per IEC-61025, is required to include the analysis, data, used

symbols, and common cause failures & minimal cut sets; where appropriate. Section 4 discusses these requirements in ConcertoFLA context and their conformance.

2.3. SafeConcert and ConcertoFLA

SafeConcert [9] is a meta-model implemented within the CHESS toolset [3] enabling dependability architects to model dependability's information necessary to conduct dependability analysis. SafeConcert is a subset of CHESS-ML (which in turn is an extension of SysML [10]), the meta-model used in CHESS toolset to enable component-based systems design.

ConcertoFLA [4], which builds on top of CHESSFLA [11], allows users (system architects and dependability engineers) to decorate component-based architectural models (specified using CHESS-ML) with dependability-related information, execute Failure Logic Analysis (FLA) techniques, and get the results back-propagated onto the original model. Different FLA techniques are available in the literature [12], and can be used at the early stages of the design phase to achieve a robust architecture with respect to linear relationships. ConcertoFLA builds on top of Failure Propagation Transform Logic (FPTC) [13]. Similar to FPTC, ConcertoFLA is a compositional technique to qualitatively assess the dependability of component-based systems, and partially combines and automatize traditional safety analysis techniques (i.e., FMEA and FTA). ConcertoFLA allows users to calculate the behavior at system-level, based on the specification of the behavior of individual components.

In ConcertoFLA terms, a component can act in four different possible ways (1) source of the failure thus generating a failure due to internal fault, (2) sink of the failure thus avoiding the propagation of the external fault (failure in input) through fault tolerance, (3) propagator of the failure, and (4) transformer of the failure into a different type. ConcertoFLA rules are logical expressions, which specify the component's behavior by describing the input/output relationship. In [14], the syntax for writing the logical expressions is given (subset of the FlaMM meta model). This syntax represents a further refinement of what was presented in [4].

3. Approach overview

Figure 1 shows a high-level view of the overall approach. This approach builds on top of the tool-supported technique presented in [4] and then further developed in [14]. In this paper, we use such approach and customize it in the context of ECSS standards to enable architects and dependability managers to calculate the failure behavior at sub/system level, as well as generating FTA based on the causality paths traced by the ConcertoFLA plugin.

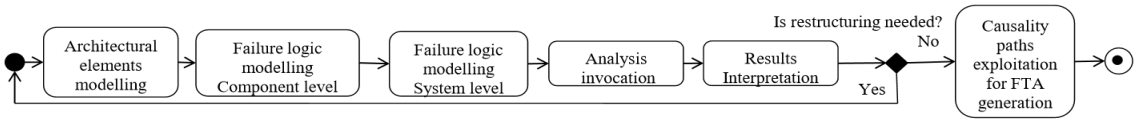

Figure 1. Approach overview, adapted from [4].

4. Modelling

In this section, first, we explain the system level assumptions, then, we model the ACS system (textually described in Section 2).

4.1. System level assumptions

Assumption 1. The system starts computing torques when valid estimates of the input sensor readings are available. An external component checks the estimated values of the input sensors and, in response to the valid estimates, activates the ACS system. This component is assumed to be analyzed already as per ECSS standards [6][7] and is out of the scope of this paper. The validity check in the component determines that the sensors are activated and registering their perceptions, and does not reason on the value itself being registered. Therefore, this assumption rules out the following failure types/modes on the input ports (1) Omission, (2) Commission, (3) Early, and (4) Late.

Assumption 2. Each of the components has been internally verified as per ECSS standard verification guidelines. Thus, none of the components in the system could act as the source of the failure. Rather, a component may only propagate, transform or sink failures.

4.2. Modelling of ACS in SafeConcert

The ACS described in Section 2 is modeled using CHESS-ML/SafeConcert, which enables for representing the system architecture utilizing the component based paradigm. ACS itself is represented as a composite component, which is composed of all the other components, collectively defining the function of the system. Figure 2 illustrates the component based design of the software system with injected failures on the input ports. ACSComposite component has three inputs and four output ports providing the interfaced sensors readings and actuators commands respectively. More specifically, the three input ports sunEstVec, angVelocity and gyroDistTorques represent Sun direction, angular velocity of the satellite and measured disturbance torques, respectively. The output ports ctrlTorque, propTorque, derTorque and feedforwardTorque corresponds to the control torque selected to be applied to the satellite for achieving the desired orientation, the torque computed through proportional controller, the torque computed through derivative controller and compensating cross coupling torque respectively.

According to the assumptions listed in Section 4.1, on the input ports, value failure could be experienced. As shown in figure 2, this value failure, which is of two types i.e. valueSubtle and valueCoarse, is injected on each input port by using the appropriate CHESS stereotype, called *FPTCSpecification*. Moreover, the failure behavior of each component is given by using the *FLABehavior* stereotype. This behavior is described as follows.

PDController component propagates and sinks the failures exposed on its input ports. If there is a valueCoarse failure on any of the input ports, the produced torques will also have a valueCoarse failure. The valueSubtle failure on sunEstVec input port is propagated to the output ports. Whereas, a valueSubtle failure occurs on the angVelocity input port, the component detects it via a low pass filter and sinks it and produces noFailure on the output ports. Similarly, FeedforwController component also detects a valueSubtle failure on its input port using a low pass filter and sinks it and produces no failure on output port. FeedforwController, however, propagates the valueCoarse failure to FeedforwardTorque output port. SteerController component propagates both the valueSubtle and valueCoarse failure on the sunestVec input port to the output torque. Finally, the TorqueSelector component also propagates both the valueSubtle and valueCoarse failure on the input ports to the ctrlTorque output port.

5. Analysis of ACS via ConcertoFLA

To apply ConcertoFLA, all the three input ports of the system shall be injected with possible failures. Considering the assumptions in Section 4.1, the possible failures are limited to only value failures i.e. valueSubtle and/or valueCoarse. This suggests two possible failures for each port. However, in this paper, for space reasons, we only consider one input port i.e., sunEstVec. This port receives faulty data. To compute the torques, PDController and SteerController rely on the data received on sunEstVec. Thus, the computed torques are also faulty (valueCoarse or valueSubtle failure). The control torque selected by TorqueSelector, based on the context i.e., satellite current attitude opposite to pointing object or closer to the pointing object (see Section 2), will also be faulty. The resulting torque, in case of valueSubtle failure on the output port, when applied on the satellite, could result in loss of mission. The valueCoarse failure could result in major mission degradation due to not pointing accurately to the object of interest. Figure 3 shows the propagated failures on the output ports.

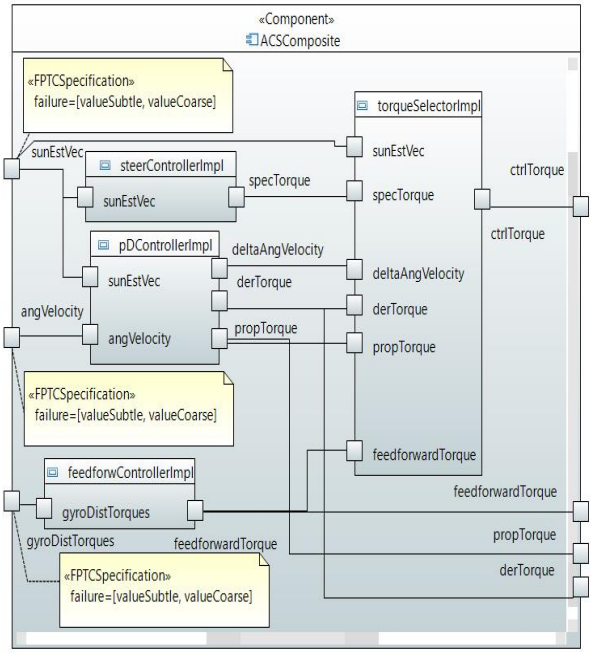

Figure 2. ACS system before the analysis.

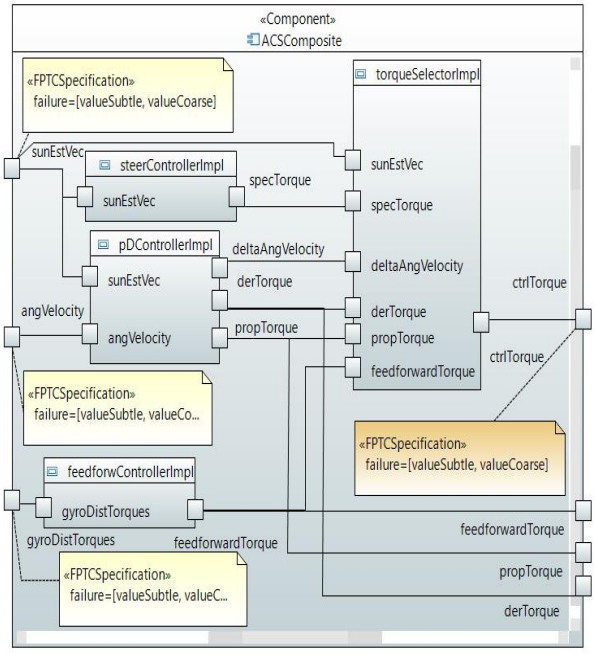

Figure 3. ACS system with analysis results.

6. ECSS-compliant FTA via ConcertoFLA

ConcertoFLA also calculates the failure propagation paths and produces their representation according to the specifications of FlaMM meta model (see [14] for FlaMM structure and corresponding XML Schema). Figure 4 shows some XML snippets representing the failure propagation paths. The yellow highlighted text illustrates the failure behavior on the CtrlTorque output port along with the information on previous failures. Figure 5 displays the same failure propagation for ACS composite component in a tree-like view. The component exhibits valueCoarse failure on the ctrlTorque output port. This is a system level failure and could be described as a top event of the FTA. The valueCoarse failure could be collapsed in the tree and the contributing previous failures could be traced, as depicted in figure 5. The ports exhibiting previous failures could be used as the contributing events of top level event, in the FTA. The information regarding the owning component of the port could also be retrieved from the tree, to uniquely identify ports with same label e.g., in the ACS case. This also conforms to one of the requirement from IEC 61025, which specifies that the FTA events shall be uniquely identified. Moreover, all the listed components in the tree could also be collapsed further to view the failure behavior displayed as tree nodes. This information is crucial to identify the relationship between an event and its descendants, otherwise depicted using logical gate symbols in FTA. Thus, the FlaMM model contains all the required information to construct an ECSS conformant FTA. Figure 6 illustrates ECSS-compliant partial FTA manually produced from the results stored in FlaMM model upon a successful execution of ConcertoFLA analysis. For the sake of clarity, it should be noted that given the qualitative nature of the ConcertoFLA analysis the fault tree cannot contain quantitative information.

7. Related work

To develop dependable systems many researchers are utilizing model based methods. In [9] and in [4], some of these works were already discussed (focus on dependability modelling and analysis). Thus, in this paper, we limit our attention to those works that have considered automatic generation of analysis artifacts based on requirements coming from standards. To enhance the V-model process of the avionics product development, in [15], authors propose a model based tool chain which embraces the end-to-end process covering all phases of software development lifecycle and focusing explicitly on dependability aspects. They also use a SysML-metamodel extension and consider FTA analysis. However, the integration of dependability analysis is still in progress and no generation is mentioned.

In [16], a model based approach is proposed to augment the structure and behavior models of railway domain system with fault related information. Effects of the presence of faults are analyzed in system and are used to complement the preliminary Failure Modes, Effects and Analysis (FMEA).

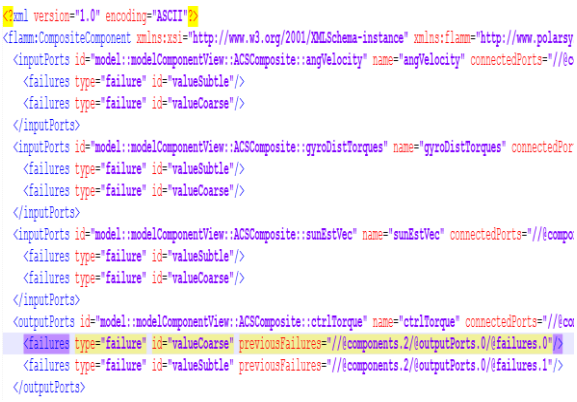

Figure 4. XML based ACS FlaMM model.

Figure 5. Tree-like ACS FlaMM model.

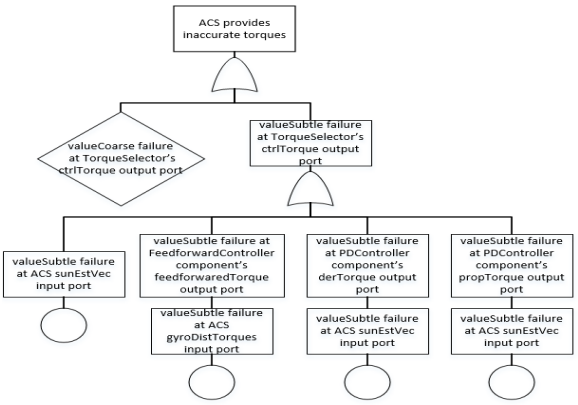

Figure 6. ECSS-compliant Partial FTA of ACS generated manually.

8. Conclusion and future work

In this paper, we have explored the usage of the tool-supported CHESS-methodology in the context of Attitude Control Systems Engineering. More specifically, we have used ConcertoFLA, the dependability analysis technique incorporated within the CHESS methodology. Via ConcertoFLA, we have modeled an Attitude Control System, its failure behavior, and performed analysis. Based on the analysis results, we could elaborate on FTA-results generation in the context of ECSS standards and more generally in the context of IEC 61025.

In the future, we aim at implementing a plugin to support a proper generation of FTA, in compliance with the largely accepted graphical concrete syntax. Based on ongoing work in the context of AMASS [17] and previous work [18] [19], we also aim at generating product and process-based argument fragments. This generation is also expected to connect ECSS-compliant product artifacts with ECSS-compliant process artifacts, since dependability analysis results constitute expected work products in a software development process model.

9. References

[1] Fortescue P and Stark J 1998 *Spacecraft Systems Engineering* (John Wiley & Sons ltd West Sussex England)
[2] Markley F L and Crassidis J L 2014 *Fundamentals of Spacecraft Attitude Determination and Control* (Springer New York/Heidelberg/Dordrecht/London)
[3] PolarSys CHESS, https://www.polarsys.org/chess/ (last accessed 03/08/2017)

[4] Gallina B, Sefer E and Refsdal A 2014 Towards safety risk assessment of socio-technical systems via failure logic Analysis IEEE Int., Symp., on Software Reliability Engineering Workshops (ISSREW) Naples Italy pp 287-292
[5] European Cooperation for Space Standardization ECSS-Q-ST-30C Space product assurance - Dependability 06/03/2009
[6] European Cooperation for Space Standardization ECSS-Q-ST-40-12C Space product assurance - Fault tree analysis – Adoption notice ECSS/IEC 61025 06/03/2009
[7] Internationl Electrotechnical Commission IEC 61025 fault tree analysis (FTA) IEC 61205:2008
[8] U.S. Nuclear Regulatory Commission NUREG 0492 1991 Fault Tree Handbook
[9] Montecchi L and Gallina B SafeConcert: a metamodel for a concerted safety modeling of socio-technical systems 5th Int., Symp., on Model-Based Safety and Assessment (IMBSA) Trento Italy September 2017
[10] SysML v1.4 Specification Release September 2015 http://www.omgsysml.org/specifications.htm
[11] Gallina B, Javed M A, Muram F and Punnekkat S 2012 Model-driven dependability analysis method for component-based architectures IEEE Proc., of the Euromicro Software Engineering and Advanced Applications (SEAA) Conf., Cesme Izmir Turkey
[12] Grunske L and Han J 2008 A comparative study into architecture-based safety evaluation methodologies using AADL's error annex and failure propagation models 11th IEEE High Assurance Systems Engineering (HASE) Symp., Nanjing China pp 283–292
[13] Wallace M 2005 *Modular architectural representation and analysis of fault propagation and transformation* Electronic Notes in Theoretical Computer Science vol, 141 issue 3 pp 53–71
[14] CONCERTO Deliverable D3.3 November 2015 Design and implementation of analysis methods for nonfunctional properties – Final version
[15] Hovsepyan A, Landuyt D V, Beeck S, Michiels S, Joosen W, Rangel G, Briones J F and Depauw J 2014 Model-driven software development of safety-critical avionics systems: an experience report, in Proc., Int., Workshop on Model-Driven Development Processes and Practices (MD^2P^2) CEUR Workshop Proceedings pp 28–37
[16] Olmo J, Poza J, Garramiola F, Nieva T and Aldasoro L 2017 Model driven hardware-in-the-loop fault analysis of railway traction systems IEEE Int., Workshop of Electronics, Control Measurement, Signals and their Application to Mechatronics (ECMSM) Donostia-San Sebastian pp 1-6
[17] AMASS (Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems) http://www.amass- ecsel.eu
[18] Alajrami S, Gallina B, Sljivo I, Romanovsky A, Isberg P 2016 Towards cloud-based enactment of safety-related processes Proc., of the 35th Int., Conf., on Computer Safety, Reliability and Security (SAFECOMP) Trondheim Norway Lecture Notes in Computer Science vol, 9922 pp 309-321
[19] Gallina B 2014 A model-driven safety certification method for process compliance IEEE Proc., of Int., Symp., on Software Reliability Engineering Workshops (ISSREW) pp 204-209

Acknowledgement
This work has been partially financially supported by the AMASS project [17]. We thank S. Puri and P. Isberg for their valuable explanation regarding CHESS tool-set and ConcertoFLA implementation.