Deriving Verification-related Means of Compliance for a Model-based Testing Process

Barbara Gallina
Mälardalen University
Västerås, Sweden
barbara.gallina@mdh.se

Anneliese Andrews
Denver University
Denver, Colorado, U.S.A.
andrews@cs.du.edu

***Abstract*—DO-331 is the supplement of DO-178C for model-based development. DO-331 is an objective-based guidance, which defines a set of objectives that have to be achieved for the model-based development of aeronautical software. The guidance also recommends the evidence in terms of activities and work products that should respectively be carried out and produced to meet the objectives. To explain why the evidence collected supports the claims concerning objectives achievement, manufacturers could adopt a safety case-based approach. Fail-SafeMBT is an academic, recently proposed, and potentially innovative model-based testing process, which needs compelling arguments to be adopted for the development of aeronautical software. To reduce the gap between industrial settings and academic settings, in this paper, we adopt the safety case-based approach and we explain how to semi-automatically derive means for compliance, aimed at arguing Fail-SafeMBT's compliance. Our focus is limited to the Verification Planning Process and we contribute to partially justify the adequacy of Fail-SafeMBT to act as process evidence by creating fragments of compelling arguments. To do that, we first manually check if Fail-SafeMBT includes DO-178C/DO-331-compliant process elements, then we model Fail-SafeMBT in compliance with Software Process Engineering Meta-model 2.0, then, we derive process-based arguments from the Fail-SafeMBT process model by using MDSafeCer, the recently introduced Model Driven Safety Certification method. By doing so, we provide a threefold contribution: we pioneer the interpretation of DO-331 in academic settings, we validate MDSafeCer in the avionics domain and we strengthen Fail-SafeMBT by providing suggestions aimed at increasing its maturity level.**

Keywords—DO-178C; DO-331; Model-based Testing; Safety cases; Certification; Means of Compliance.

I. INTRODUCTION

Manufacturers of aircraft shall justify via an implicit safety case [1] that the software performs its intended function with a level of confidence in safety that complies with airworthiness requirements. Such implicit safety case is known as Plan for software aspects of certification during the planning phase (preliminary safety case) and as Software accomplishment summary containing compliance substantiation via software life cycle data during the implementation phase (operational safety case).

The software life-cycle data used for means for compliance derive from the planning of the all the needed processes, including the verification process. According to DO-178C [2]/ DO-331 [3], this process consists of a set of activities that are selected and ordered in order to meet the expected objectives. Besides a set of ordered activities, the verification process also consists of other relevant process elements: the verification methods, the verification results, the organizational responsibilities, and the qualification of the tools used, if any.

To make the implicit safety case explicit and thus to accelerate the certification process, it is of utmost importance to be able to explain why, a certain verification process is deemed adequate to meet the expected objectives.

MDSafeCer [4] is a recently introduced Model Driven Safety Certification method, which adopts model-driven engineering principles to automate the generation of process certification artefacts. More specifically, via MDSafeCer, process models compliant with SPEM (Software Process Engineering Meta-model) 2.0 [5] are transformed into argumentation models compliant with SACM (Structured Assurance Case Metamodel) [6] and presented via GSN (Goal Structuring Notation) [7]-goal structures.

The recently proposed systematic fail-safe model-based testing process (see [8] and [9]), called in this paper Fail-SafeMBT, is a potentially innovative end-to-end testing process targeting testing of mitigation behaviour. In the presence of compelling arguments concerning its adequacy as verification process evidence, this end-to-end testing process could support the safety claims within the required safety case. In this paper, we contribute to partially justify the adequacy of Fail-SafeMBT to act as process evidence. To do that, we first manually check if this testing process includes DO-178 C/DO-331 compliant process elements, then we model it in compliance with SPEM 2.0, then, we derive means for compliance from the Fail-SafeMBT-related process model by using MDSafeCer. By doing so, we provide a threefold contribution: we pioneer the interpretation of DO-331 in academic settings, we validate MDSafeCer in the avionics domain and we strengthen Fail-SafeMBT by providing suggestions aimed at increasing its maturity level.

II. BACKGROUND

This section recalls the essential background on which the presented work is based.

A. Fail-SafeMBT

Model-based testing (MBT) is a testing technique that uses a model that describes a system under test (SUT) to produce test

cases. Fail-SafeMBT is an end-to-end MBT process aimed at increasing robustness. Fail-SafeMBT consists of two-phases.

The first phase uses fault trees (FTs) and a behavioral model (BM), specified using Communicating Extended Finite State Machines (CEFSM) to ascertain and test that failures, as specified in the fault tree, can actually occur and what tests steps and procedures must be created to make these failures happen. This, first phase, in essence, integrates FT and BM into an Integrated Communicating Extended Finite State Machine and then uses this integrated model to generate test cases based on test criteria (IC) that lead to failures. It also determines in which behavioral states they can occur. The result is a failure applicability matrix.

The second phase takes in input this information and the required safety mitigation requirements. It builds mitigation models and creates test cases for them. It determines failure scenarios based on systematic scenario failure coverage criteria and builds failure mitigation tests for them to systematically determine which failures needs to be tested and what the expected mitigation results are to be. These tests can be created early in the development lifecycle as they are based on functional (black box) models.

For sake of clarity, it should be noted that Fail-SafeMBT has not been applied in the avionics domain yet. Concerning its level of maturity, it should be observed that in terms of motivation and definition, Fail-SafeMBT is fairly mature. However, in terms of validation as well as measurement, it is still in its embryo stage. A true validation in industrial settings has not yet been carried out. Thus, no compelling arguments can be derived from process models pertaining its execution in compliance with DO-178C/DO-331.

B. DO-178C & DO-331

DO-178C provides guidance for the development of software for airborne systems and equipment. Associated to DO-178C, there are specific supplements which have to be used in case of, for instance, specific development methods are used. DO-331 is to be used in case model-driven engineering approaches are used. DO-331 indicates the modifications and additions to DO-178C.

The purpose of DO-178C together with its supplements is to guarantee a level of confidence in the correct functioning of the software developed in compliance with airworthiness requirements. To do that, it provides a series of processes characterized by a set of objectives, activities and expected deliverables. These processes address the different phases of the entire software life-cycle, including the planning. The planning entails the provision of a series of plans, which have to be approved by the certification body during the first interaction between the applicant and the certification body itself. Once the plans are approved, the applicant can start the real development consisting of the execution of the plans.

Among the expected deliverables of the process planning we have: software development plan (SDP), software verification plan (SVP), and Plan for Software Aspects of Certification (PSAC). In the context of this paper we will focus on these three deliverables. SDP is a plan, which provides a detailed description concerning how the software should be developed. More specifically, SDP includes: a) the identification of software model standards, whose definition be provided according to MB.11.23 of DO-331; b) the software life-cycles; and c) the software development environment.

SVP is a plan that provides detailed description concerning what the applicant intends to do to satisfy the verification process planning objectives. Section MB11.3 of DO-331, indicates that SVP shall include information concerning: a) Organizational responsibilities; b) Independence; c) Verification methods (testing methods, including the method for selecting test cases, the test procedures to be used, and the test data to be produced.).

SDP and SVP provide additional information with respect to the one already included within PSAC, which, as stated in [7], serves as the primary means for communicating the proposed development methods to the certification authority for agreement and defines the means of compliance with DO-178C/DO-331. PSAC includes the software life- cycle, the software life-cycle data, plus various other items that are not in focus in this paper. PSAC, together with other plans (SDP and SVP), is aimed at gaining agreement on the means of compliance through approval of the plans.

C. Process Modeling

Based on our previous work on process modeling [10] and process-based model-based certification [10-13], this background section recalls basic information on SPEM 2.0. SPEM2.0 is the OMG's standard for systems and software process modelling. SPEM 2.0 supports the definition of reusable process content, i.e., work definition elements (e.g., tasks, etc.) as well as elements representing: who is responsible for the work (roles), how the work should be performed (guidance), what should be expected as in/output (work-products) and which tool should be used to perform the work. In Fig. 1, we recall a subset of SPEM 2.0 modelling elements, which are interrelated in this paper to model static process structures.

Fig. 1. Subset of SPEM2.0 Icons

D. MDSafeCer

Based on our previous work (see [4] and [14]), this background section recalls basic information on MDSafeCer. MDSafeCer is a method that adopts MDE principles to enable the semi-automatic generation of composable process-based argument-fragments within safety cases. Via MDSafeCer, process models compliant with a process modeling language meta-model (e.g., SPEM 2.0) are transformed into argumentation models compliant with SACM and presented via for instance GSN-goal structures. MDSafeCer generates process arguments based on the argumentation pattern given in [14]. The pattern can be instantiated either during the planning phase (preliminary safety case) of during the execution phase (operational safety case). The instantiated pattern at the planning phase is constituted of a top level claim stating that the planned

process is in compliance with the required standard-level. This claim can be decomposed by showing that all the process activities have been planned and that in turn for each activity all the tasks have been planned and so on until an atomic process-related work-definition unit is reached.

E. GSN

Based on our previous work on safety case building [15], this background section recalls basic information on GSN. GSN is a graphical notation, which permits users to structure their safety case-argumentation into flat or hierarchically nested graphs (constituted of a set of nodes and a set of edges), called goal structures. To make the paper self-contained, in Fig. 2, we recall a subset of the GSN concrete syntax used in this paper. As Fig. 2, borrowed from [15] shows, all the nodes are characterized by an identifier (ID) and a statement, which is supposed to be written in natural language.

We recall that a *Goal* represents a claim about the system; a *Strategy* represents a method that is used to decompose a goal into sub goals; a *Solution* represents the evidence that a particular goal has been achieved; a *Context* represents the domain or scope in which a goal, evidence or strategy is given; *Supported by* represents an inferential (inference between goals) or evidential (link between a goal and the evidence used to substantiate it) relationship. Finally, *In context of* represents a contextual relationship. Note that an *undeveloped goal*, which is intentionally left undeveloped in the argument, is depicted as a goal decorated with a hollow-diamond 'undeveloped entity' symbol at the centre-bottom [7].

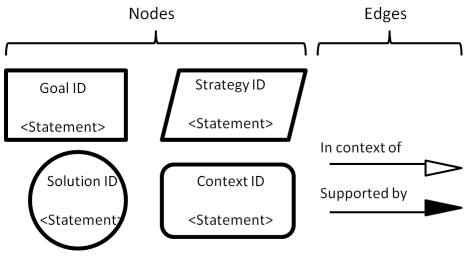

Fig. 2. Partial Concrete Syntax of GSN

III. Collecting and Modeling Elements of Compliance

In this section we proceed in a similar way as we did in a previous work (see [12] and [14]) related to the exploitation of MDSafeCer for deriving safety case arguments in the railways domain. Thus, first of all, we collect and model elements of compliance since to partly act as means for compliance, needed for process verification planning, Fail-SafeMBT must be the result of the selection and composition of process elements that can be considered compliant with respect to Objective 1 ("The activities of the software life cycle processes are defined."), listed within Table MB.A-1 "Software Planning Process", in DO-331.

Fail-SafeMBT is a process to be used during the verification process for testing mitigation behavior. Thus, first, its definition should be aligned with the Software Verification Planning Process. As recalled in the background, according to DO-331 (Section 11-3), the SVP should define the appropriate roles, the appropriate work to be executed (broken down in terms of activities/tasks), the independence of the work, the specific guidelines, the expected work products to be consumed and produced. Moreover, according to DO-331, the model standards used in th SVP, should be defined in the SDP. Since, as recalled in background, Fail-SafeMBT contains some of the required elements, its compliance can be partially argued about. More specifically, the following list highlights the process elements that meet to some extent the DO-331 requirements: the phases that compose Fail-SafeMBT can be considered aligned with the Software Verification Planning Process. However, not all the required tasks are included in Fail-SafeMBT. This means that a company should be aware about what else should be performed.

Similarly to what we identified in a previous work [14] concerning possible compliance of academic-presented processes/methods, also in the case of Fail-SafeMBT, we identify that to enable its usage in real settings, its presentation should be enhanced and its alignment clearly made explicit. More specifically, all input/output work products should be specified and aligned with DO-331.
Roles and their responsibility should be defined. Independence related to the expected activities should be defined. Tools (and more generally the verification environment) should be indicated as well as the rational and adequate justification in terms of their quality should be provided. Concerning stringency, which varies according to the criticality level, it should be emphasized.

Concerning the model standards, their definition and motivation is given (for details see [7-8]); however, the given definition and motivation does not meet all the requirements provided in MB.11.23. For instance, it is not clear at which abstraction level Fail-SafeMBT is applied. In Fig. 3 and Fig. 4, we illustrate the SPEM 2.0 models representing the augmented Fail-SafeMBT tasks.

By construction, these augmented Fail-SafeMBT tasks contain process elements that are to some extent in compliance with DO-331. To explain this compliance, in the following section we derive process-based arguments and we document them in GSN. Besides the enhancement of the presentation, to satisfy all the DO-331, Fail-SafeMBT should, however, be further developed or combined with another methodology offering complementary support. Thus, given the awareness developed thanks to the performed gap analysis, we also indicate the undeveloped goals.

Fig. 3. Fail-SafeMBT's Phase-1 in SPEM2.0

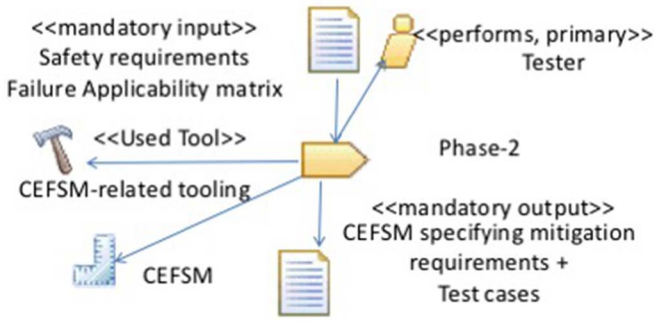

Fig. 4. Fail-SafeMBT's Phase-2 in SPEM2.0

IV. Deriving Verification-related Means of Compliance

In this section, we derive means for compliance. More specifically, we derive a process-based argument for arguing about Fail-SafeMBT compliance with DO-331. Our derived argument given in GSN argues that Fail-SafeMBT is partially compliant with the DO-331 requirements related to the software development and verification planning processes. As previously done, to derive such argument, based on the pattern incorporated by MDSafeCer, we proceed compositionally and from the process models given in Fig. 3 and Fig. 4, by using MDSafeCer, we first derive sub-arguments that argue about compliance at phase level (see Fig. 5 and Fig. 6). Then, such arguments could be further developed to indicate the missing evidence (e.g., the missing work products). Finally, to argue about compliance at the process level, we compose sub-arguments as indicated in Fig. 7 and Fig.8.

As Fig. 5 and Fig. 6 show, several goals require to be further developed. The generation of undeveloped goals is due to the presence of "undeveloped" in the text-field description of the process elements depicted in Fig. 3 and Fig. 4. Given the unsatisfactory provision of information pertaining to the definition and the rationale of important process elements, the arguments remain unfounded. The generation of unfounded arguments does not mean that Fail-SafeMBT is unfounded. It simply means that given its still low maturity in terms of documentation, its usage in the context of industrial safety-critical systems is discouraged. At the same time, further development is encouraged. Fail-SafeMBT needs to be better documented and motivated as well as further applied to produce the needed evidence.

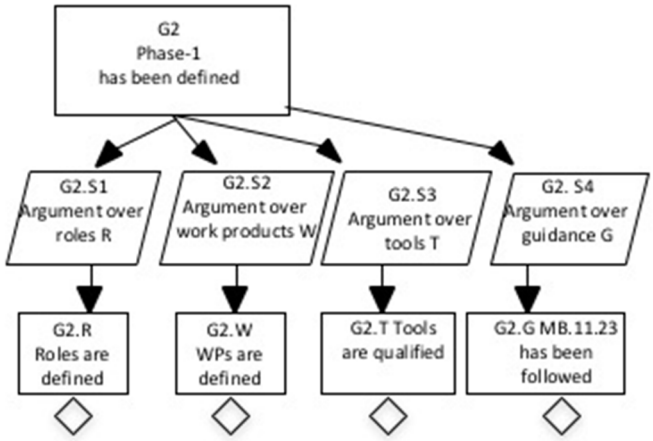

Fig. 5. Sub-argument in GSN related to Phase-1

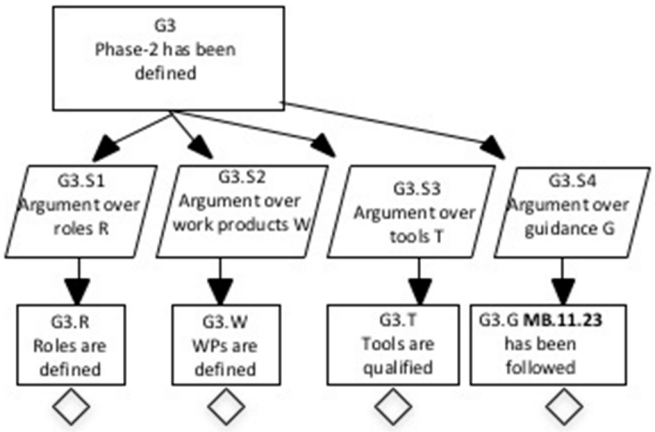

Fig. 6. Sub-argument in GSN related to Phase-2

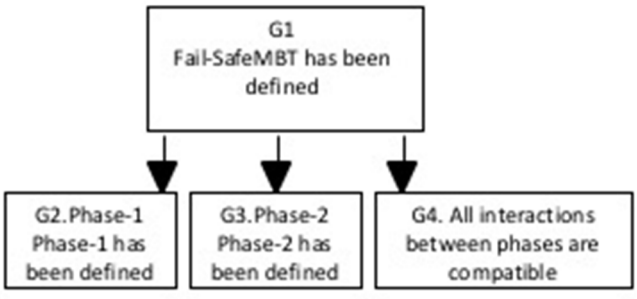

Fig. 7. Argument about Fail-SafeMBT Compliance

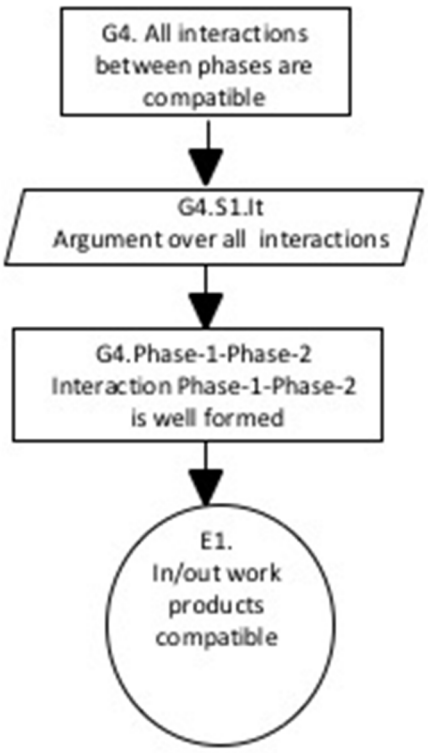

Fig. 8. Argument about Interactions Compliance

V. Related Work

The current certification framework is traversing a crisis's phase due to the growing complexity associated to the safety justifications that are required by the standards [16] and [17]. Moreover, technology and standardization are evolving at different speed and usage of technological solutions risk to be hindered because of conservative requirements coming from standards. To contribute to the solution of the crisis, methods to speed up the creation of safety justifications are beneficial [4] and their application to the emerging technological solutions in order to test their robustness against the standards [14] or in order to challenge the standards [18] is crucial. In its effort aimed at strengthening via process-based evidence an existing model-based testing process, our work represents a novelty and an effort to contribute to the reduction of the gap between academy and industry. Paz et al. [19] have also explored compliance requirements concerning DO-178C in case of model-based development. Their investigation also contributes in reducing the gap between industry and academia. Their approach, however, is not safety case-based. Stallbaum et al. [20] have explored compliance requirements for model-based testing in the avionics domain in order to facilitate its application. However, their work focuses on DO-178B.

VI. Conclusion and Future Work

Newly proposed and potentially innovative engineering methods suffer of low acceptance in avionics industrial settings due to the requirements of the certification process. Methods aimed at speeding up the provision of process-based arguments can be beneficial. In this paper, we have used MDSafeCer to show that Fail-SafeMBT can be partly used as testing-planning-related evidence within a safety case. More specifically, we have focused on specific portions of the DO-178C/D331 documents related to verification process compliance and we have argued by using GSN about compliance with DO-331-related verification process planning.

In the future, to achieve a full compelling process argument, we will further develop Fail-SafeMBT according to the findings. Ideally, all undeveloped goals should be replaced by well-founded and explained goals. Once the Fail-SafeMBT's maturity in terms of documentation is increased, we will apply it to test the mitigation behavior of aeronautical software. Then, based on the data coming from the execution of Fail-SafeMBT, we will derive also compliance substantiation data, i.e. we will derive arguments to be included in the operational safety case showing that the objectives related to the Verification of the Verification Process Results are met.

Acknowledgment

This work has been partially supported by the ECSEL AMASS (No 692474) [21] project.

References

[1] Holloway, C. M., 2013, Making the Implicit Explicit: Towards an Assurance Case for DO-178C, Proc ISSC, Boston, MA, ISSS.

[2] RTCA DO-178C (EUROCAE ED-12C), 2013, Software Considerations in Airborne Systems and Equipment Certification, RTCA Inc, Washington DC.

[3] RTCA DO-331, 2011, Model-Based Development and Verification Supplement to DO-178C and DO-278A, RTCA Inc, Washington DC.

[4] Gallina, B, November 3-6, 2014, A Model- driven Safety Certification Method for Process Compliance. 2nd IEEE International Workshop on Assurance Cases for Software-intensive Systems (ASSURE), joint event of ISSRE, Naples, Italy, doi: 10.1109/ISSREW.2014.30, pp. 204-209.

[5] OMG, 2008, Software & systems Process Engineering Meta-model (SPEM), v 2.0. Full Specification formal/08-04-01, Object Management Group.

[6] SACM:http://www.omg.org/spec/sacm/1.0.

[7] GSN: Community Standard Version 1, 2011.

[8] A. Gario, A. Andrews and S. Hagerman, 2014, Testing of safety-critical systems: An aerospace launch application, IEEE Aerospace Conference, Big Sky, MT, pp. 1-17.

[9] A. Andrews, S. Elakeili and S. Boukhris, 2014, Fail-Safe Test Generation in Safety Critical Systems, IEEE 15th International Symposium on High-Assurance Systems Engineering (HASE), Miami Beach, FL, pp. 49-56.

[10] Gallina, B., and K. R. Pitchai and K. Lundqvist, 2014, S-TunExSPEM: Towards an Extension of SPEM 2.0 to Model and Exchange Tuneable Safety-oriented Processes. 11th International Conference on Software Engineering Research, Management and Applications (SERA), SCI 496, Springer, ISBN 978-3-319-00947-6, Prague, Czech Republic, August 7-9, 2013.

[11] Gallina, B., K. Lundqvist and K. Forsberg, October 5-9, 2014, THRUST: A Method for Speeding Up the Creation of Process-related Deliverables. IEEE 33rd Digital Avionics Systems Conference (DASC-33), doi:10.1109/DASC.2014.6979489, Colorado Springs, CO, USA.

[12] Gallina, B., L. Provenzano, June, 2015, Deriving Reusable Process-based Arguments from Process Models in the Context of Railway Safety Standards. 20th International Conference on Reliable Software Technologies-Industrial Presentation- (Ada-Europe), Madrid, Spain.

[13] B. Gallina, L. Fabre. Benefits of Security-informed Safety-oriented Process Line Engineering. IEEE 34th Digital Avionics Systems

Conference (DASC-34), Prague, Czech Republic, September 13-17, 2015.

[14] B. Gallina, E. Gomez-Martınez, and C. Benac Earle. Deriving Safety Case Fragments for Assessing MBASafe's Compliance with EN 50128. 16th International SPICE Conference on Process Improvement and Capability dEtermination (SPICE), Dublin, Ireland, Vol. 609, Communications in Computer and Information Science series, Springer, 2016.

[15] Dardar, R., B. Gallina, A. Johnsen, K. Lundqvist, M. Nyberg, 2012, Industrial Experiences of Building a Safety Case in Compliance with ISO 26262, in: 2nd International Workshop on Software Certification, International Symposium on Software Reliability Engineering Workshops (ISSREW), IEEE, 349–354.

[16] Gallina, B., 2015, How to increase efficiency with the certification of process compliance. In the 3rd Scandinavian Conference on SYSTEM & SOFTWARE SAFETY, Stockholm, March 24-25

[17] Ferrell, T., U., Ferrell, 2014, Assuring Avionics - Updating the Approach for the 21st Century, Proceedings of SafeComp-Workshops, the International Conference on. Computer Safety, Reliability and Security (SafeComp), Lecture Notes in Computer Science, Vol. 8696, pp. 375-383, Springer, Florence Italy.

[18] McDermid, J. A., 2014, Nothing is Certain but Doubt and Tests, CoRR, arXiv:1404.6801 [cs.SE].

[19] A. Paz and G. El Boussaidi, 2016, Preliminary analysis of model-based support for DO-178C-compliant avionics software development and certification, Université du Québec, École de Technologie Supérieure, Montreal, Canada, Tech. Rep. [Online]. Available: http: //dx.doi.org/10.13140/RG.2.1.1241.4960.

[20] Stallbaum, H., M. Rzepka, 2010, Toward DO-178B-compliant Test Models, Model-Driven Engineering, Workshop on Verification, and Validation (MoDeVVa), Oslo, pp. 25-30.

[21] AMASS (Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems). http://www.amass-ecsel.eu.