A Universal Configuration Format for Avionics

Philipp Chrysalidis [1], Frank Thielecke[1]

Abstract: Avionics module configuration, especially in the face of advancing technologies, will become more complex as computational demands rise. This requires a robust and automated approach while adhering to industry standards. However, state-of-the-art configuration is still highly error-prone and suffers from various stakeholders working with unsynchronized and decentralized data. This causes unnecessary iterations, leading to delays in development. The Universal Configuration Format for Avionics (UCoF), integrated into the AvioNET framework, presents a forward-looking solution. UCoF, built upon a meta-model approach, strives to enhance the configuration process through model-based methods. It meets essential configuration management requirements and offers versatility by supporting the configuration of diverse avionic platforms. Combining essential data for configuring real avionics device families, implementation targets and network design grants users access to a comprehensive data set throughout the configuration process.

Keywords: Avionics; Configuration; MBSE

1 Introduction

The configuration of avionics modules is a complex and still highly error-prone process, especially with new technologies on the horizon (e.g., single-pilot cockpit, smart cabin) that demand higher computational capabilities. Moreover, a more robust and automated configuration process is essential for efficiently implementing these new technologies.
However, aviation standards provide a well-defined framework for development, including the configuration of Integrated Modular Avionics (IMA) modules as defined in ARINC 653 [Ae12]. Since IMA is integral to aviation, any configuration process must adhere to these standards.
Furthermore, module and application development and testing are distributed and often collaborative processes. These trends are likely to intensify, with advances in model-based systems engineering (MBSE) leading to more virtual and hybrid testing. To ensure smooth operations and minimize delays in development under these circumstances, transparency and consistency of information are of utmost importance. Without these, faults may propagate to later development phases, leading to time-consuming iteration loops. Meanwhile, there is currently no common industry standard for avionics configuration, and stakeholders often rely on proprietary formats.
To address this challenge, the Institute of Aircraft Systems Engineering (FST) at the Hamburg

[1] Hamburg University of Technology, Institute of Aircraft Systems Engineering, Nesspriel 5, 21129 Hamburg, Germany, firstname.lastname@tuhh.de

doi:10.18420/sw2024-ws_04

University of Technology (TUHH) has developed the Avionics Next-Gen Engineering Tools (AvioNET) framework [HT21]. AvioNET is a seamless end-to-end toolchain based on generic MBSE methods, offering solutions for a more efficient design process throughout all development phases. Key elements of AvioNET include Architecture, Configuration, Simulation, Testing, Verification & Validation, Visualization, and Avionics Data Management. Developing methodologies within this framework ensures immediate consideration of all necessary interfaces for seamless information transfer during development. Utilizing the AvioNET framework, an approach for automating the configuration process, has already been implemented at FST. The model-based method outlined in [CHT23] automatically derives the configuration for an avionics module from a set of requirements. However, AvioNET's scope extends to platform solutions, necessitating a more generic approach that includes data management for data consistency.

The Universal Configuration Format for Avionics (UCoF), introduced in this paper, is integrated into AvioNET. This approach enables model-based configuration not only for avionics hardware but also for virtual and testing environments, facilitating Software-in-the-Loop (SIL) and Hardware-in-the-Loop (HIL) testing for continuous integration and validation.

This paper will contextualize current research in section 2. Section 3 will lay out the requirements regarding configuration management. Section 4 will present the UCoF approach. Finally, the paper will be summarized in section 5, with an outlook for future research provided.

2 Related Work

In 2010, Horváth et al. [HVS10] presented a framework for systematically designing ARINC 653 configuration tables for the Wind River VxWorks Real-Time Operating System (RTOS). Using meta-modeling as proposed by the ARINC 653 standard, they created a data model from which specific design instances could be derived. To align the meta-model with IMA development roles, the model was divided into four subgroups: Applications, Health Monitoring, Communication, and Interface Control Document. This concept was implemented in the Eclipse Modeling Framework (EMF) [Fo23], enabling the use of the model in their overall design approach and toolchain. However, this approach heavily focused on ARINC 653 implementation and lacked universal applicability.

Darif et al. [Da22] took a similar approach to embed a model-based approach into a tooling environment for an RTOS. Their goal was to support multiple RTOS to reduce certification costs by introducing a high degree of reusability with their concept. Additionally, their tooling could generate certification data for ARINC, such as configuration tables and test data.

In contrast to the previous works, Annighoefer [An19] employed a different approach to configuration. Instead of replicating the ARINC 653 standard in a meta-model, they used a generic avionics architecting model. This model aimed to perform on all levels of detail during the design of avionics components, allowing for a wide range of modeling targets.

Therefore, the model could be used for configuring proper ARINC 653 certified hardware [An20].
Furthermore, a query language for meta-models expanded the generic capabilities and enabled easy access to the model. By combining both the query language and the generic avionics model, interfaces for data transfer from and to other formats could be easily created. These methods facilitated highly automated processing of configuration data in tooling environments, as seen in Mueller et al. [Mu23]. However, it's worth noting that the very generic approach has a significant disadvantage, since participants could interpret the model differently, leading to compatibility issues.
Halle; Thielecke [HT09] presented the challenges faced during the configuration process due to the involvement of various stakeholders while needing to maintain data consistency. They introduced a meta-model approach, combining it with data management, to support validation, continuous integration, extensibility of the format, and interfaces to other formats or tools. UCoF builds strongly upon these works, further expanding on the presented solutions and broadening the defined scope.

3 Model-Based Avionics Configuration Management

Considering more recent developments described by Martinen et al. [Ma17] and Uludağ et al. [Ul23], information flow between stakeholders will become even more complex than what was described in [HT09]. Test rigs will gain even more prominence in development and may be locally distributed, communicating through long-range internet connections. Examples of these concepts can be found in the works of Chrysalidis et al. [CHT22] and Martinez et al. [MGG22]. In the latter case, the newly created standard by EUROCAE [EU20] for distributed testing was successfully utilized, indicating the path toward future developments. Information must be readily available across different locations with a guarantee of uncompromised data.
The advances in virtual integration also allow for more reliable Software-in-the-Loop (SIL) and Hardware-in-the-Loop (HIL) testing earlier in the process. However, the setup of the required virtual or hybrid environments introduces an added layer of configuration complexity. Increasing the efficiency of the configuration process, therefore, includes solutions for these environments. The scope of UCoF encompasses the formalization and standardization of the configuration of testing environments while still needing to guarantee uncompromised data between all participating stakeholders.
Moreover, as standards evolve through updates and expansions, and new standards continue to emerge, UCoF's sustainability relies on its ability to readily accommodate these evolving specifications.

3.1 Requirements for Configuration Management

Based on the outlined scope, we present the requirements for a configuration management format capable of meeting future needs in Table 1.

Requirement	Description
Satisfy Standards	The configuration format has to satisfy the most common and most important standards and process requirements in avionics development.
Modularity	Modularity is needed, so that different parts of the configuration can be completed independently. It is also a strong basis for the reusability requirement.
Reusability	An efficient process greatly profits from a high degree of reusability, ensuring less redundant tasks.
Traceability	Changes to the configuration need to be traced to both a time and a stakeholder. This way, faults can be identified more clearly.
Accessibility	Accessibility means both easy access to the model (i.e. open source) and intuitive use. This means, that the usage domain must be clearly defined.
Generality	The model must be as generic as possible, as to incorporate the maximum amount of information for the maximum amount of systems, with the least amount of redundant data.
Automatibility	The format must be machine-readable and accessible through parsing.
Adaptability	The configuration must be open to new technologies, while still maintaining backwards compatibility, as to ensure smooth integration of new configuration targets.
Information Propagation	Information must flow bidirectionally throughout the configuration process, meaning that both a source and a sink for information must be provided, including corresponding interfaces.
Scalability	Configurations need to be stored efficiently and accessible for all project sizes.

Tab. 1: Requirements for Configuration Management

4 Universal Configuration Format for Avionics

UCoF is built upon the requirements outlined in subsection 3.1 and aims to be a future-proof solution for avionics configuration. UCoF is based on a meta-model which is implemented using EMF [Fo23]. The process of creating a graphical user interface is streamlined through the automation of code generation, allowing for the rapid assessment of the meta-model's effectiveness in specific instances.
The primary objective of UCoF is to expand the platform definition and incorporate test systems in both hardware and software, providing swift and easy access to SIL and HIL testing. By integrating these test systems as target platforms within the model, data consistency is assured, with no information concealed within transformation scripts (see Figure 1). This approach to transparency makes working with different hardware intuitive and also allows for direct information links, aiding in the debugging processes.
Moreover, changes made to target configurations can be easily traced, enabling the reuse of these configurations for different platform components. Thus, UCoF supports continuous integration of avionics configurations throughout all stages of development.
Configuration data in UCoF is categorized into three primary groups: “Devices”, “Testing” and “Network” (see Figure 2). The “Devices” group encompasses all configurations for individual modules included in the platform. Module configuration is generic and independent of the target, while implementation-specific configurations for real hardware, test benches, and virtual environments are separated. Real hardware configuration is

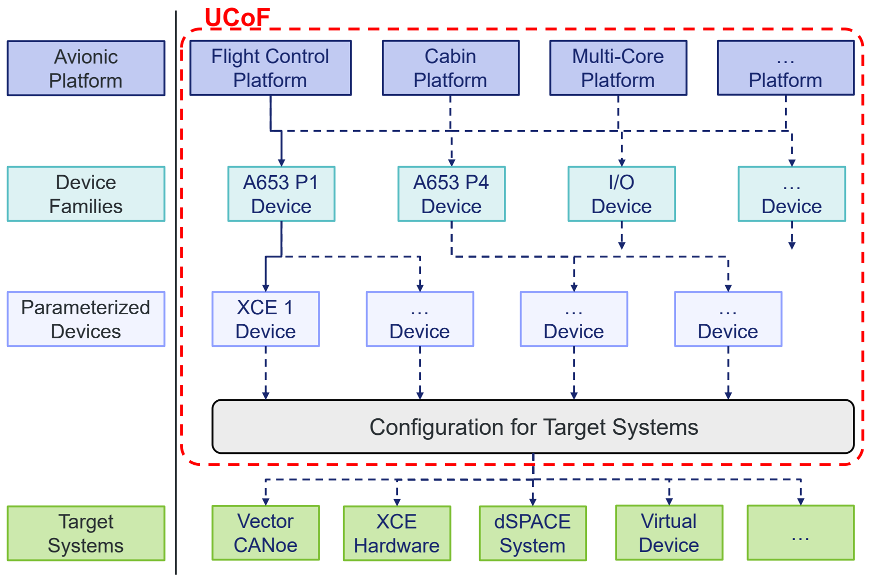

Fig. 1: UCoF Approach

also included in the "Device" group, while configurations for test benches and virtual environments are found in the "Testing" group. The "Testing" group additionally includes the definition of basic testing procedures. This allows for test bench configurations to be included in the model through references, while maintaining a clear separation from the proper avionics hardware configuration. Additionally, the definition of basic testing procedures facilitates quick test case execution by providing a central information hub through UCoF. Within the "Network"group, configurations for device communication, such as switches and gateways, are specified. The generic configuration of gateways presents an innovative strategy for tackling the challenges associated with test virtualization, as outlined in section 3.

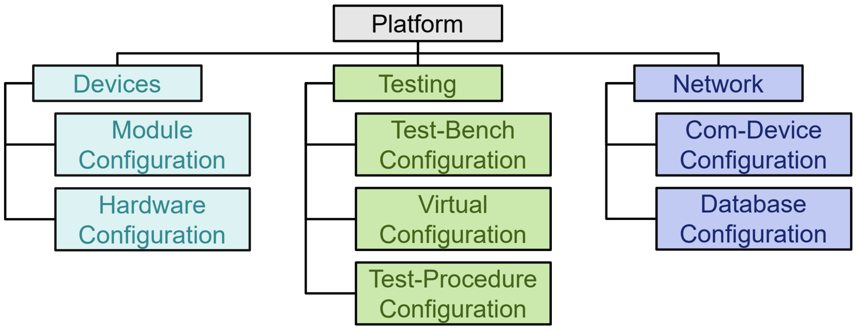

Fig. 2: UCoF Structure

4.1 Data Categorization in UCoF

Figure 3 demonstrates the categorization of data within UCoF through a simplified example. As previously explained, data is categorized into master and target specific information. Both datasets are stored separately but interconnected via references. Consequently, target-specific configurations solely extend the provided master information, allowing for independent addition of new targets. Most of the information is reused across all targets, minimizing redundancy even when accommodating various implementations. Modifications to target-specific data are isolated, while adjustments to master data resulting from tests can be proposed and accepted by multiple stakeholders.
In this example, CAN bus data is defined for a test system and an ARINC 653-compliant avionics device. General data comprises attributes like the identifier (CAN ID), data length code (DLC), and baud rate, which remain constant across all implementations and are shared among specific configurations. Target-specific configurations might involve assigning channel sets and in-device termination for test systems or drivers and hardware ports for the ARINC 653 device.
Alterations to the ARINC 653 driver or test system termination are isolated to their targets and don't affect the master configuration. However, should a too high baud rate be detected during HIL testing with the test system, this change can be automatically applied to the master data and verified. UCoF's integrated data connectivity facilitates subsequent reconfiguration of the ARINC 653 device without necessitating additional user input, since the updated data is referenced within the model. Consequently, this workflow significantly reduces the likelihood of errors and ensures consistent behavior across different targets.

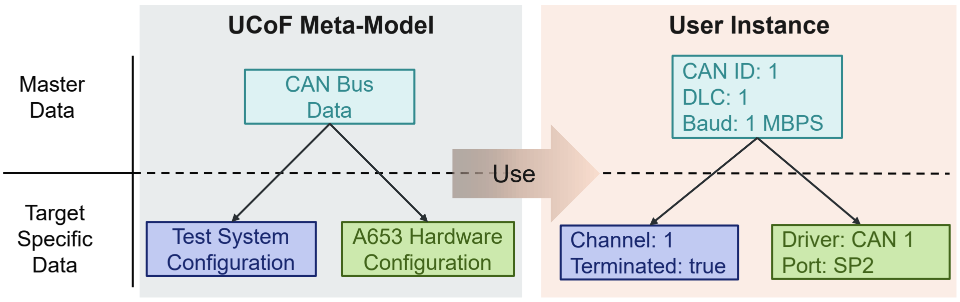

Fig. 3: UCoF Data Categorization

The implementation of this approach in UCoF is demonstrated in Figure 4 in a simplified excerpt of the meta-model. The UML classes are categorized into two main groups: master and target-specific data, further segmented into "Devices", "Network", and "Testing". Focused on CAN communication, relevant classes in the model are consolidated as "classes" to improve clarity
The modeled master device represents an ARINC 653 compliant module. In these, sampling ports can be defined for communication with other modules. Configuring a sampling port entails defining the respective communication protocol and payload. In UCoF, this

data is stored separately in a database linked to the device, enabling a clear overview of communication and facilitating data reusability.
However, to meet implementation requirements, information within the “Devices” and “Network” sections needs expansion. To achieve this, the master device configurations reference potential implementations. For instance, in the context of a test system, the CAN configuration undergoes further specification through additional attributes. The generation of the executable target configuration can only be automated with this specific, target-related information.

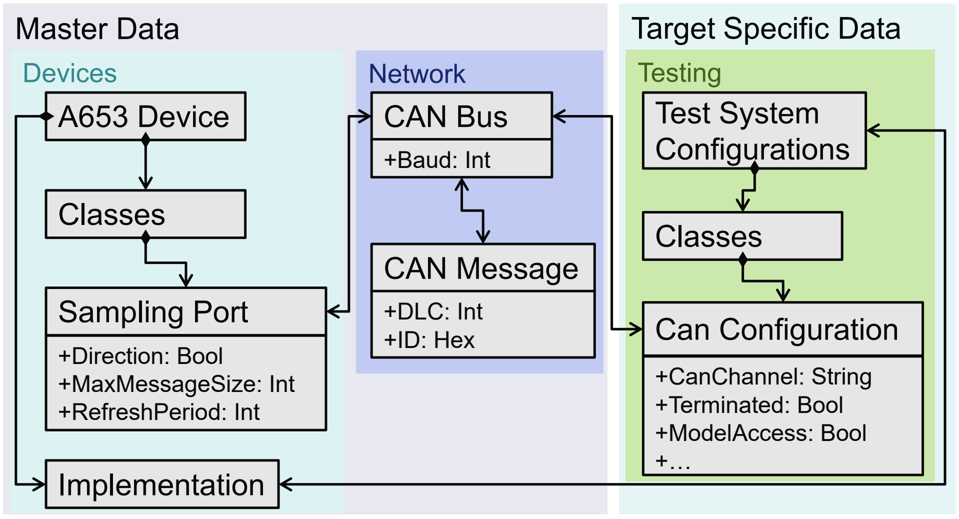

Fig. 4: Simplified UCoF UML Excerpt

Additionally, Figure 4 illustrates the benefits of UCoF’s modular approach. By linking information through references, new configuration data can be seamlessly incorporated while preserving the fundamental structure of the core elements. For instance, the configuration of an ARINC 653 compliant device maintains an adherence to the established standard, ensuring minimal structural alterations over time.
Conversely, configuring test systems proves more dynamic due to frequent updates within proprietary configuration environments. This volatility necessitates a more flexible approach to accommodate changes effectively. Moreover, the format’s capability to easily integrate new, emerging implementation targets, underlines the required future-proof architecture of the meta-model.

4.2 Working with UCoF

UCoF seamlessly integrates configuration information throughout the development cycle, supporting not only individual devices, but also facilitating configuration for the entire

avionics platform. To start, input for platform configuration, sourced either from a singular or multiple sources (such as a platform architecture), is required.
This input is further specified by relevant stakeholders, for example the system integrator defined in ARINC 653. As depicted in Figure 5, the master configuration data is derived from this input and serves as the baseline for all target configurations.
Depending on the development stage, single devices or the already defined platform can be configured for the respective targets. Early in development, when hardware might not yet be available, the master configuration can be expanded to encompass configuration information for a simulation environment. This enables a SIL approach early in the development, adding further iteration cycles for the configuration data, without any redundant information. Updates stemming from SIL testing can seamlessly be integrated into the model via interfaces that enable information flow to the UCoF model.
Advancing through the development stages, the SIL approach may transition to HIL testing, where available hardware and test systems come into play. Target specific configurations for real hardware or test systems expand the master configuration data, greatly reducing the configuration workload during this stage. The creation of executable code or other configuration artifacts, such as proprietary project files for test systems, are created through automation interfaces. These interfaces are set up bidirectional, allowing for changes to be propagated back to the UCoF model. Therefore, the platform configuration can be adapted dynamically with a minimal risk for errors. In instances where the platform remains partially virtualized, the inclusion of gateways becomes essential to enable comprehensive communication among all modules. With UCoF, the configurations of said gateways are included in the same model and are part of the overall platform configuration, thereby ensuring the holistic platform approach for all modules.
Upon the availability of actual hardware for the complete platform, expanding the master configuration data simply involves specifying configurations for the finalized modules. With automatic data updates and transfer, information continuity is preserved between testing phases and the final implementation. The model's traceability attributes ensure comprehensive tracking of changes, guaranteeing data integrity throughout the process.

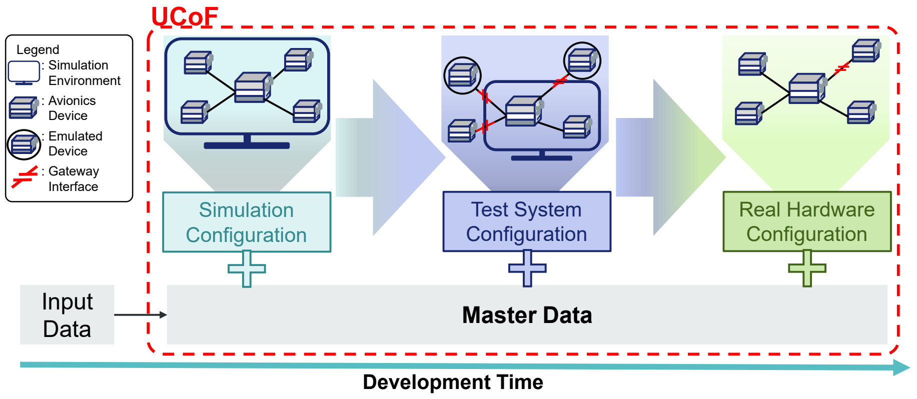

Fig. 5: UCoF during Development

5 Conlusion and Outlook

Avionics module configuration is a complex process, exacerbated by higher demand in computational power due to emerging technologies. This paper introduces the Universal Configuration Format for Avionics (UCoF) within the AvioNET framework, aiming to provide a future-proof solution. UCoF is based on a meta-model and enhances platform configuration by unifying device, network and testing configurations within a single format, ensuring data consistency and transparency through all development steps.
UCoF's design adheres to key configuration management requirements such as adherence to standards, modularity, reusability, traceability, accessibility, generality, automatibility, adaptability, and bidirectional information flow. Moreover, UCoF's segregation of master and target-specific information ensures interconnectedness while offering a highly adaptive model, enabling continuous integration. This categorization minimizes redundancy, enhancing efficiency and reliability across avionics development.
Future research will expand UCoF to encompass additional implementation targets. It will also delve into developing generic automation methods to simplify the creation of interfaces for input and output data. Furthermore, UCoF's open-source project release is planned, once a well-defined beta version is available, fostering collaborative development and broader adoption.

Acknowledgement

This work was funded by the German Federal Ministry of Economic Affairs and Climate Action (BMWK) within the TALIA project. Their support and the cooperation of the partners is greatly appreciated. The authors of this publication are responsible for its content.

Supported by:

Federal Ministry for Economic Affairs and Climate Action

on the basis of a decision by the German Bundestag

References

[Ae12] Aeronautical Radio, Inc.: Avionics Application Software Standard Interface: ARINC Specification 653P1-4, Standard, 2012.

[An19] Annighoefer, B.: An Open Source Domain-Specific Avionics System Architecture Model for the Design Phase and Self-Organizing Avionics. SAE Int. J. Adv. &. Curr. Prac. in Mobility 1/2, pp. 429–446, 2019.

[An20] Annighoefer, B.; Brunner, M.; Schoepf, J.; Luettig, B.; Merckling, M.; Mueller, P.: Holistic IMA Platform Configuration using Webtechnologies and a Domain-specific Model Query Language. In: 2020 AIAA/IEEE 39th Digital Avionics Systems Conference. Virtual Conference, pp. 1–10, 2020.

[CHT22] Chrysalidis, P.; Halle, M.; Thielecke, F.: Testing of High-Lift System Functions with a Distributed Avionics Test Bench. In: 2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC). Portsmouth, VA, USA, pp. 1–8, 2022.

[CHT23] Chrysalidis, P.; Hoeber, H.; Thielecke, F.: A semi-automated approach for requirement-based early validation of flight control platforms. CEAS Aeronaut. J. 14/1, pp. 271–280, 2023, ISSN: 1869-5590.

[Da22] Darif, I.; Politowski, C.; El-Boussaidi, G.; Kpodjedo, S.: A Domain Specific Language for the ARINC 653 Specification. In: IEEE International Symposium on Software Reliability Engineering Workshops. Charlotte, NC, USA, pp. 238–245, 2022.

[EU20] EUROCAE: Technical Standard of Virtual Interoperable Simulation for Tests of Aircraft Systems in Virtual or Hybrid Bench, EUROCAE ED-247A, Standard, 2020.

[Fo23] Foundation, E.: Eclipse Modeling Framework, [Online; accessed 5. Oct. 2023], 2023, URL: `https://eclipse.dev/modeling/emf`.

[HT09] Halle, M.; Thielecke, F.: Next Generation Konfigurationsmanagement for IMA, FlyING 2009, Oct. 2009.

[HT21] Halle, M.; Thielecke, F.: Avionics Engineering Tool Network (AvioNET):Experiences With Highly Automised and Digital Processes for Avionics Platform Development. In: 2021 AIAA/IEEE 40th Digital Avionics Systems Conference (DASC. San Antonio, TX, USA, 2021.

[HVS10] Horváth, A.; Varró, D.; Schoofs, T.: Model-Driven Development of ARINC 653 Configuration Tables. In: 29th Digital Avionics Systems Conference. Salt Lake City, UT, USA, 6.E.3-1-6.E.3–15, 2010.

[Ma17] Martinen, D. H.; Lagalaye, M.; Pfefferkorn, J.; Casteres, J.: Modular and Open Test Bench Architecture for Distributed Testing, [Online; accessed 4. Oct. 2023], 2017.

[MGG22] Martinez, R.; Gaurel, C.; Gruber, M.: Distributed Testing using VISTAS (ED-247 RevA). In: European Test and Telemetry Conference. Nuremberg, Germany, pp. 227–231, 2022.

[Mu23] Mueller, P.; Merckling, M.; Tietz, V.; Frey, C.; Luettig, B.; Waldvogel, A.; Annighoefer, B.; Abdo, K.; Halle, M.; Thielecke, F.: Introduction of a Dedicated Platform Level for IMA Systems Development With an Extensive Automation Tool Support. In: 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC). Barcelona, Spain, 2023.

[Ul23] Uludağ, Y.; Bayoğlu, Ö.; Candan, B.; Yılmaze, H.: Model-Based IMA Platform Development and Certification Ecosystem. In: 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC). Barcelona, Spain, 2023.