The METASAT Model-Based Engineering Workflow and Digital Twin Concept

Alejandro J. Calderón[1], Irune Yarza[1], Stefano Sinisi[2], Lorenzo Lazzara[2], Valerio Di Valerio[2], Giulia Stazi[2], Leonidas Kosmidis[3], Matina M. Trompouki[3], Alessandro Ulisse[2], Aitor Amonarriz[1], Peio Onaindia[1]
[1]Ikerlan Technology Research Centre [2]Collins Aerospace [3]Barcelona Supercomputing Center (BSC)

***Abstract*—Considering the complexity in the design of future satellites and the need for compliance with ECSS standards, the METASAT project proposes a novel design methodology based on model-based engineering and supported by open architecture hardware. The initiative highlights the potential of software virtualisation layers, like hypervisors, to meet standards compliance on high-performance computing platforms. Our focus is on the development of a toolchain tailored for these advanced hardware/software layers. In our view, without such innovation, the satellite industry may face high and unsustainable development costs and timelines, which could affect its competitiveness and reliability. This paper provides an overview of the model-based engineering toolchain, the workflow, and the digital twin concept proposed for the METASAT project. We present the purpose of each component and how they work together in the broader METASAT vision, with the objective of showing how this approach can make the development process more efficient and enhance dependability in the sector.**

***Index Terms*—Model-Based Design, Digital Twin, Hypervisor, Satellite, High-Performance, Modularity, Reusability**

I. Introduction

The space domain, as many other sectors, is actively considering novel methods and tools based on artificial intelligence, digital twins, virtual design and testing, and other Industry 4.0 concepts, in order to manage the increased complexity of the design of upcoming satellites. Nevertheless, especially from the satellite on-board software engineering point of view, these technologies require a solid ground to be built upon. First of all, the computational power of the hardware platform must meet the needs of the advanced algorithms running on top of it. The software layer too must both allow an efficient use of the hardware resources and at the same time guarantee non-functional properties such as dependability in compliance with ECSS standards. Finally, design methods need to adapt to the specific challenges posed by both the increased complexity of hardware/software (HW/SW) layers and Industry 4.0 concepts.

The METASAT Horizon Europe project, which started in January 2023, will address these challenges. The METASAT vision is that a design methodology based on Model-Based Engineering (MBE), jointly with the use of open architecture hardware, constitutes that solid ground. To reach its vision, METASAT will leverage existing software virtualisation layers (e.g., hypervisors), which already provide guarantees in terms of standards compliance, on top of high-performance computing platforms based on open hardware architectures. The project focuses on the development of a MBE workflow and a toolchain to design software modules for this HW/SW layer. Without such measures, the time and cost of developing new systems could become prohibitive as the complexity of the system increases, reducing competitiveness, innovation, and potentially dependability across the industry. A more detailed description of the project can be found in [1].

In this paper, we present a summary of the work done on the definition of the METASAT toolchain, the MBE workflow, the digital twin concept at its core, and the METASAT integration framework. This latter enables a tool-independent solution to Digital Twins building, supporting multiple virtual integration and interoperability standards. This poses the basis for a vendor-independent, modular, interoperable and collaborative framework avoiding technology discontinuity while promoting model reuse across the development cycle. All these ingredients together represent the METASAT contribution. The remainder of this paper is organized as follows. Section II provides background information on the METASAT hardware and software platform, model-based engineering, and digital twin technology. Section III presents the tools constituting the METASAT toolchain and their integration. Section IV presents the METASAT workflow and digital twin concept. Section V presents the METASAT integration framework. Finally, Section VI presents the conclusions of the paper.

II. Background

A. The METASAT Hardware and Software Platform

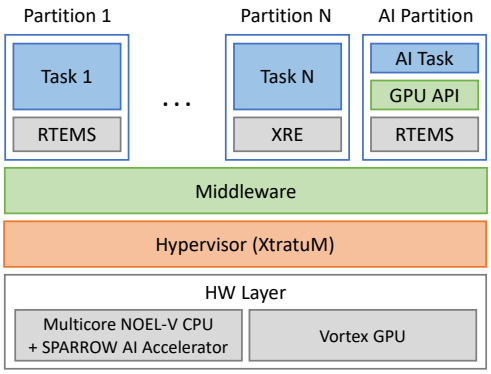

Fig. 1. The METASAT HW and SW Platform

The research presented throughout this paper has received funding from the European Commission's Horizon Europe programme under the METASAT project (grant agreement 101082622).

979-8-3503-4859-0/DATE24/© 2024 EDAA

In terms of hardware, the METASAT platform will employ a multicore processor, based on the NOEL-V space processor from Frontgrade Gaisler. To enhance the AI processing capabilities of NOEL-V, the METASAT CPUs will be integrated with the SPARROW AI SIMD accelerator [2]. The METASAT platform will also include a Vortex general purpose Graphics Processing Unit (GPU). The Vortex GPU will be connected to the CPU cluster using an AXI interface. Regarding the software platform, METASAT use case applications run on partitions on top of the XtratuM hypervisor over the RISC-V architecture. Each partition could either run RTEMS or be a baremetal partition based on XRE (XtratuM Runtime Environment). The different partitions will be able to share the SPARROW AI accelerator and the GPU among them. Moreover, they will require access to serial and Ethernet devices. Figure 1 shows a high-level overview of the METASAT HW and SW platform.

B. Model-Based Engineering (MBE)

MBE is a contemporary approach adopted in METASAT to manage the complex system development of the METASAT platform leveraging modeling and simulation to support all lifecycle stages from design, to validation and to verification. MBE centralises models in system engineering, capturing requirements, behaviours, architecture, and interfaces visually and structurally. These models ensure consistency, completeness, and traceability throughout development and are pivotal in integration, validation, and verification activities, including digital twin-based analyses. MBE incorporates various simulation and analysis methods to evaluate system performance, design alternatives, and optimise parameters.

C. Digital Twin Technology

Digital Twin technology offers a virtual representation of physical systems or processes, enabling engineers to simulate, monitor, and analyse system behaviour and performance of the system digitally. It integrates real-time data from the field with physics-based models and advanced analytics to create a digital replica that mimicks the behaviour and functionality of its physical counterpart. Typically, there are different definitions of digital twins depending on the lifecycle stage where they are employed: as-designed digital twins (the ones employed in METASAT for validation and integration verification analyses), as-built digital twins, and as-operated digital twins. The digital twin technology provides several benefits such as, e.g., virtual prototyping, what-if scenario evaluations, monitoring and predictive maintenance, and human decision making support.

III. MBE Toolchain

This section provides an overview of the multiple tools comprising the METASAT MBE toolchain.

A. TASTE

TASTE (The ASSERT Set of Tools for Engineering) is a set of tools dedicated to the development of embedded real-time systems. This open-source toolchain addresses the modelling, automatic code generation and deployment of distributed systems composed of heterogeneous software and hardware components [3] [4] [5]. TASTE allows importing modules of heterogeneous application code produced manually (in C or Ada) or automatically using external modelling tools such as MATLAB/Simulink and SDL (Specification and Description Language). The tool ensures consistency in the integration of application code modules using the formal languages AADL (Architecture Analysis and Design Language) [6] and ASN.1 (Abstract Syntax Notation One) [7], which allow a precise and complete description of the system architecture and data. AADL is a modelling language that is used in the development of real-time, safety-critical systems. It allows modelling the HW and SW components, as well as their interaction. To this end, it provides a standardised and formal notation for describing the architecture and non-functional requirements of complex systems, focusing on aspects such as system structure, communication, timing, and performance. Whereas ASN.1 is an ISO/IEC and ITU-T standard that allows for specification of data structures, both from the semantic as well as the encoding point of view. For automatic generation of code and build scripts, TASTE integrates the use of the Kazoo tool [8]. Kazoo builds the executable application taking as input the functional blocks, the glue code generated by the tool itself to handle communication between functional blocks, and the AADL models defining the hardware and software interactions of the system. For AADL parsing, Kazoo uses the Ocarina tool [9].

B. MathWorks Tools

MathWorks offers a set of tools specialised in mathematical computing, data analysis, and algorithm development. Among those, MATLAB and Simulink serve as high-level programming language and graphical modelling and simulation environment, respectively. Additionally, MathWorks provides various complementary toolboxes that are used in the METASAT project, such as, e.g., *Simulink Requirements*, to manage requirement formalization and enable traceability; *Simulink Test*, to automate validation and testing of Simulink models and generated code; *Embedded Coder*, to assist engineers and developers in the process of generating efficient and optimised C and C++ code for embedded systems; and *GPU Coder*, to allow the generation of CUDA (i.e., a parallel computing platform and programming model developed by NVIDIA for GPU programming) code from MATLAB or Simulink. the Vortex GPU of the METASAT platform, based on RISC-V, mainly supports the Khronos OpenCL standard. Regarding the latter, a recent work by Vortex developers demonstrated the possibility to execute CUDA code on the Vortex GPU [10] of the METASAT platform, an option that will be explored in the METASAT project.

C. QEMU Instruction Set Simulator

The QEMU Instruction Set Simulator [11] is an open-source machine simulator and virtualizer supporting the execution of target SW binaries on a host computer. The QEMU ecosystem enables the emulation of a wide variety of platforms including RISC-V (i.e., the targeted HW for METASAT).

D. Verilator

Verilator is an open-source software tool that takes as input the Verilog HDL code of an RTL IP and creates an equivalent cycle-accurate behavioral C++ or SystemC model. This language conversion keeps the abstraction level of the converted model equal to the one of the original RTL IP. Verilator compiles the generated model into a much faster optimized model which provides typically higher performance than the more widely used event-driven simulators.

IV. The METASAT MBE Workflow and Digital Twin concept

The METASAT Model-Based Engineering (MBE) workflow consists of a model-centric approach that leverages models throughout all the different stages of the entire system development lifecycle, taking into account the phases suggested by the ARP-4754 [12] and ECSS-E-ST-40C standards [13]: from requirements formalization, to system design and implementation, to system integration validation and verification (V&V) analysis. Explicits links between the artifacts produced at each stage ensure coverage and traceability from SW artifacts up to system requirements.

The METASAT MBE Workflow consists of seven main stages, positioned in the V-model as shown in Figure 2. To support the different MBE stages and the integration analyses of interest, METASAT exploits the concept of Digital Twin as a virtual system that closely models an entire complex system. This is increasingly employed in various industries such as manufacturing and it is very valuable also for space systems, which traditionally rely building exact space system twins on ground for the operational phase. METASAT exploits Digital Twins starting from the left side of the V-model, to define a digital twin that facilitates, through representative simulations, the validation of system-level requirements and the evaluation of different design alternatives. Moving to the right side of the V-model, the Digital Twin will evolve to support integration verification, first at item-level, to verify the implementation of each SW component, and then at system level, to verify the full system integration.

The following subsections describe with more details the METASAT MBE process in terms of the different stages, the artifacts that shall be exchanged among such stages, the employed model-based toolchains, and the role of the digital twins.

A. System Requirement Specification

The first step in the workflow is the system requirement specification. In this stage, system requirements are collected, analysed and formalised, and test procedures defined to test given requirements to cover a wide range of scenarios and conditions. These cases specify inputs, expected outputs, and the desired system behaviour. Collected system requirements are linked to test cases to ensure traceability and coverage during validation and verification analysis. Similarly, in the following stages of the METASAT workflow, system requirements will be linked to design artifacts and generated code, to facilitate the identification of design modules or tests directly impacted by any changes in the requirements. In this stage, within the METASAT workflow, we propose to use the Simulink Requirements Toolbox. If an external tool is utilised, like IBM DOORS or Microsoft Excel, once all system specification requirements are gathered, they shall be transferred to Simulink Requirements.

B. System Architecture Design and Validation

Driven by system requirements, the next step in the MBE workflow involves the design and validation of the system architecture. To achieve this, it is necessary to define and integrate: (i) a functional architecture model, by decomposing the system into functional components and specifying the interfaces for their interconnection; (ii) a physical architecture model, by decomposing the system into a set of interconnected hardware components. Once validated, such a system architecture model will be used to drive the composition of the different design and SW artifacts for system-level integration validation and verification analyses.

During the design, the TASTE toolchain and the AADL standard are used to capture the functional and physical architecture. By means of the TASTE Interface View the functional components and their relationship are defined in terms of provided and required interfaces. Then, the TASTE Data View is used for defining data types and function parameters. This is achieved by means of the ASN.1 standard including all the semantic information about the data carried over the invocation of the interface, as well as the limitations on the allowed values. Similarly, leveraging the TASTE Deployment View, the physical architecture model is defined and the functional blocks are mapped to the elements of the selected target hardware platform (i.e., functional and physical architecture integration).

This system architecture is validated at several points during the architecture design phase. Requirement coverage is assessed using the Simulink Requirements toolbox, where model skeletons generated from TASTE's architecture design are matched with mapped requirements to ensure comprehensive coverage. Additionally, architecture integration is validated through the TASTE Deployment View, wherein functional blocks from the Interface View are mapped to target platform elements and interfaces. This allocation process ensures validation of integration in the design of the system architecture.

C. System Design

Once the system architecture design has been completed and validated, the workflow proceeds with the detailed system design defining executable behavioural models for the functional elements of the system architecture. To this end, using TASTE we automatically generate the design model skeletons (compliant to the interfaces defined in the system architecture model) as Simulink blocks. The behavior of these blocks is provided then by the user leveraging system requirements. These models include portions of the application model that will be running on the satellite (application models), and will be integrated with the plant model and the operating environment model (physical models) developed directly in Simulink. As

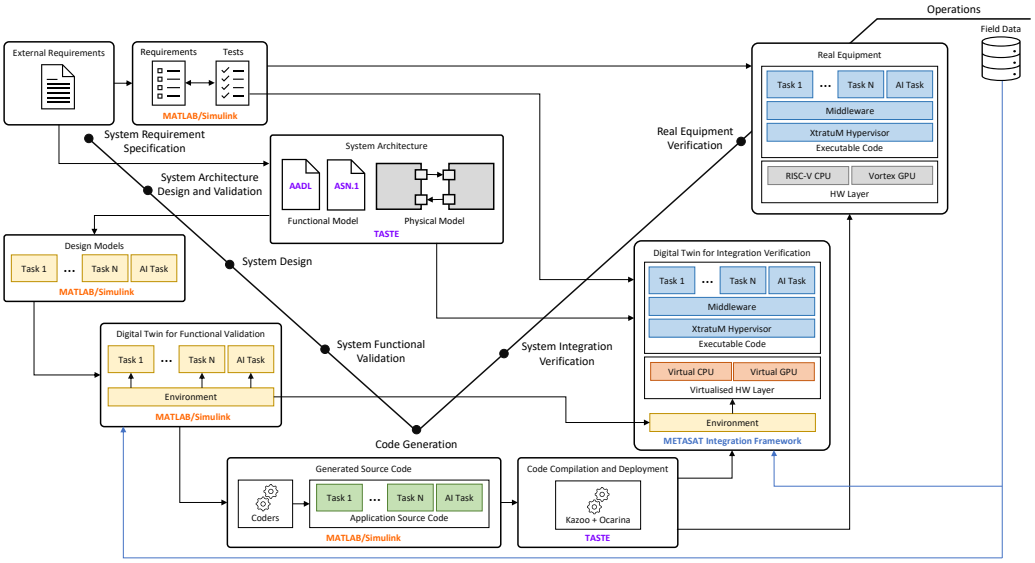

Fig. 2. The METASAT MBE Workflow

a result of the system design stage, we obtain the necessary models to create a digital twin for system functional validation, which is used in the next step.

D. System Functional Validation

The system functional validation stage focuses on assessing the system-level integration of the behavioural design models defined in previous stage and aligned with on-field data by coupling these models with the data-driven environment model (developed Simulink) and a real test bench. The so obtained system model defines a functional *Digital Twin* of the system and it is used for functional validation analysis executing the test procedures defined in the System requirements stage. We remark that this functional digital twin provides the means for validating the system-level behaviour against requirements specifications at an early design stage. Indeed, individual components are developed and tested independently, but also in a full system simulation. The Model in the Loop (MiL) validation strategy is at the foundation of such a digital twin enabling the system-level validation and testing of functional requirements.

This stage is executed in MATLAB/Simulink (as all the design artifacts are defined in Simulink) and provides as output a report of the system behavioral analysis. Simulink allows design traceability by linking the system model to the system requirements using the Simulink Requirements Toolbox. This allows the validation of the design model behaviour by means of requirement implementation.

E. Model-Based Design (MBD) and Code Generation

In this stage, a MBD workflow is employed focusing on the automatic SW code generation from the behavioural design models previously defined and validated. Automatic SW code generation is the cornerstone of any MBD methodology and can help to significantly decrease the effort of the SW verification stage. In general, if the code generation tool is qualified, there is no need to re-run tests already executed on the models. Vice versa, there is still the need to verify the code, but the troubleshooting effort is usually significantly lower because tests have been already successfully executed on models. This automatic SW code generation activity is perfomed by combining MATLAB/Simulink and TASTE. In particular MathWorks Embedded Coder generates C/C++ code, and GPU Coder generates CUDA code. Both tools have the ability to generate code for AI inference for different frameworks (e.g., TensorFlowLite, cuDNN and TensorRT). To conclude this code generation process, the automatically generated C/C++ or CUDA code is imported into TASTE, where the Kazoo and Ocarina tools are used for glue code generation and for the compilation process. This last step includes the configuration needed for the correct deployment of the software on the system, such as configuration related to the use of high-performance features like OpenMP, the SPARROW AI accelerator, the GPU on RTEMS and the hypervisor configuration for these features, as well as the partition configuration.

F. System Integration Verification

This stage focuses on the verification of system-level integration against the system requirements and executing the test procedures defined in the System requirements stage. To achieve this, a (as-designed) *Digital Twin* for integration verification is built. The Virtual Processor in the Loop (VPiL) approach is at the foundation of such a digital twin. Indeed, this digital twin integrates sw artifacts (i.e., SW binary objects, from application to operating system and device drivers) with models of the target METASAT virtual HW platform (i.e., emulated models of target processor, memories and HW peripherals) fed with data from the field by coupling the data-driven environment model and a real test bench. VPiL simulations allow testing the target EOC (source compiled for the target machine) without the need of a physical hardware platform. VPiL simulations allow to verify different test objectives such as, e.g., early

verification of the EOC generated for the target machine; SW/SW integration verification testing: integration testing of the developed application software, middleware, and real-time operating system; HW/SW integration testing: to verify the correctness of the operations executed by the SW on the target virtualized platform; and verification of communication between different control units leveraging the different virtualized communication interfaces (e.g., AFDX, CAN or Ethernet). In addition, VPiL supports code coverage during testing, enabling the analysis of the detailed execution traces provided by the virtual platform, avoiding the use of code instrumentation.

As for the previous integration analyses, this composition is driven by the AADL model of the functional and physical architecture (Section IV-B). The METASAT virtual HW platform consists of QEMU models of the RISC-V virtual board, emulating the riscv64 ISA, that will be suitable modified to model the NOEL-V space processor and to add support for the SPARROW AI accelerator; and a model of the RISC-V based GPU, Vortex, derived directly from its Verilog implementation. The data-driven environment model is the same already employed in the previous system functional validation analysis.

To build such a digital twin, the METASAT Integration Framework of Section V will be employed relying on open standards for model exchange and co-simulation, to guarantee interoperability between the different model-based toolchains.

G. Real Equipment Verification

This last stage consists of testing the METASAT SW and HW platform prototype coupled with the environment model by means of Hardware-in-the-Loop (HiL) simulations. The HiL approach tests the interaction of the physical prototype (i.e., the METASAT platform) with the simulated environment model, evaluating the interplay between hardware and software and verifying the overall system functional and non functional behavior.

V. The METASAT Integration Framework

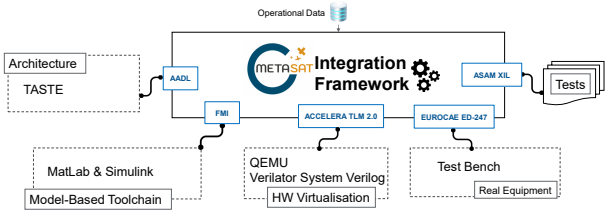

Fig. 3. The METASAT Integration Framework

METASAT proposes an integration framework (namely, the METASAT Integration Framework, see Figure 3) to support multi-domain simulation of digital twins and their evolution during the V-cycle. As described in Section IV, METASAT Digital Twins are made by the integration of different components defining the METASAT HW/SW Platform (Section II-A). These are cyber domain components (such as, e.g., communication, computational, application SW at different level of abstraction) and multi-physics plant components in the physical domain (e.g., the environment model). Usually each of these heterogeneous domains is modelled according to a suitable simulation semantic, i.e., Model of Computations (MoCs): the Discrete Event MoC for, e.g., communication models; the Discrete Time MoC for, e.g., HW models; and the Continuous Time MoC for, e.g., physical plant models. The intent of the METASAT Integration Framework is to enable the integration and the simulation of such heterogeneous domains by properly orchestrating the different MoCs. In this setting, engineers can use the same modelling tools they are used to (i.e., the ones identified in the MBE workflow of Section IV), and that are specialized for the specific subsystem domain, and then integrate their models using the METASAT Integration Framework. To achieve this, a key enabler feature is the support of multiple virtual integration and interoperability standards to propose a tool-independent solution to the Digital Twins building. This is crucial to counteract interoperability problems due to the lack of a unique and universal model-exchange standard and the need of a different and specialized modelling toolset to cope with the heterogeneous nature of space applications. This poses the basis for a vendor-independent, modular, interoperable and collaborative framework avoiding technology discontinuity while promoting model reuse. This framework will be based on the DESYRE multi-domain simulation framework described in Section V-B. DESYRE supports a continuous validation and verification of a system behaviour and performance as the system matures and evolves during its V-cycle.

The remainder of this Section describes the supported interoperability standards and the DESYRE technology baseline.

A. Interoperabilty standards

- The TLM 2.0 standard to enable the model-exchange of components based on memory-mapped buses and on-chip communications [14] and co-simulation with Instruction Set Simulators (ISS). This standard eases the hardware/software co-simulation and it allows to cover both the functional and the architectural part of a system. In METASAT this standard is used to integrate models of HW processors and peripherals.
- The Functional Mock-up Interface (FMI) 2.0 standard [15] to enable the co-simulation of models (e.g., physical plants) developed and exported as Functional Mock-up Unit (FMU) in a different modelling environment. In METASAT this standard is used reuse and integrate the environment model developed in Simulink by exporting it as FMU.
- The ED-247 standard to enable a virtual or hybrid (real and virtual equipment) integration in a local or geographically distributed setup by means of different virtualisation rules for the different signals and communication bus [16]. In METASAT this standard is explored to support the integration of the Digital Twin for System Integration Verification (Section IV-F) with a real test bench.
- The ASAM XIL standard [17] to enable test reuse across different integration verification analyses (e.g., MiL, SiL, VPiL, HiL) and among different test systems. In METASAT this standard is explored to reuse, from left

to right of the V-cycle, the test procedures defined in the very first stage of the MBE workflow.

B. DESYRE: multi-domain simulation framework

DESYRE is a multi-domain simulation framework for system virtual integration and testing, based on state-of-the-art open standards (such as SystemC TLM 2.0, FMI 2.0, ED-247), and developed by Collins Aerospace (see [18] for more details). DESYRE is the technology baseline adopted in METASAT to build the METASAT Integration Framework. DESYRE supports simulation-based system-level analyses of Cyber-Physical Systems at different stages of the V-cycle. By means of virtual system prototypes, DESYRE enables a broad set of benefits, from facilitating the evaluation of different design alternatives to early requirements validation and system-level integration V&V. In particular, the following three capabilities, enabled by DESYRE, are of relevance for the METASAT MBE integration analyses described in Section IV.

1) SW Virtual Testing & Certification: This capability enables the integration of the application SW on top of a model-based execution platform consisting of models of the communication (e.g., network bus) and computational (e.g., OS, HW peripherals) platforms. The application SW can be integrated at different abstraction levels (e.g., as a retargeted SW application or as a rehosted binary compiled for the target platform [13]) according to the preferred verification strategy (e.g., SiL or VPiL, respectively) to verify SW/SW or SW/HW integration. This capability scales-up and speeds-up the SW verification activities by anticipating and migrating them from real HW prototypes to virtual model-based execution platforms. This in turn has the potential to pave the road to SW certification by simulation or to just a single certification run.

2) System Virtual & Hybrid Integration for V&V: This capability leverages a fully virtual or a hybrid integration (i.e., where virtual and real equipment can be seamlessly integrated together) to enable: i) the co-design of system parts that are usually developed by heterogeneous and multi-domain teams; ii) a fast prototyping of explorative system solutions without creating real prototypes; and iii) the discovery of system integration and performance issues without the need of setting up real integration laboratories.

3) Collaborative Virtual & Hybrid Integration for V&V: This capability builds on the previous one but focuses more on the collaborative aspect of the integration analyses that can be performed in an extended enterprise setting (e.g., customer-supplier integration). This includes also preserving the Intellectual Properties when sharing and integrating other party component models.

VI. Conclusions

In this paper we presented the METASAT toolchain, an MBE workflow which covers the entire system design cycle for satellite applications, the digital twin concept at its core, and the METASAT integration framework. The proposed approach takes advantage of the specification, modelling, simulation, and code generation capabilities of TASTE and MathWorks toolsets, as well as the system virtual integration and testing capabilities of DESYRE. This latter provides the basis for the METASAT integration framework, and enables a tool-independent solution to Digital Twin building, thanks to the use of interoperability standards. The defined solution allows the system design process to be accelerated and ensures accuracy and reliability. It also allows the seamless development and testing of complex systems that involve the integration between different components in the cyberphysical domain, which can be leveraged for complex satellite applications development.

References

[1] L. Kosmidis, A. J. Calderón, A. Álvarez, S. Sinisi, E. Göhler, P. Gómez, A. Hönle, A. Jover, L. Lazzara, M. Masmano, P. Onaindia, T. Poggi, I. Rodríguez, M. Solé, G. Stazi, M. M. Trompouki, A. Ulisse, V. Di Valerio, J. Wolf, and I. Yarza, "METASAT: Modular Model-Based Design and Testing for Applications in Satellites," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*. Springer Nature Switzerland, 2023, pp. 347–362.

[2] M. Solé and L. Kosmidis, "SPARROW: A Low-Cost Hardware/Software Co-Designed SIMD Microarchitecture for AI Operations in Space Processors," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022.

[3] M. Perrotin, E. Conquet, P. Dissaux, T. Tsiodras, and J. Hugues, "The TASTE Toolset: Turning Human Designed Heterogeneous Systems into Computer Built Homogeneous Software," in *ERTS2 2010, Embedded Real Time Software & Systems*, 2010.

[4] M. Perrotin, E. Conquet, J. Delange, A. Schiele, and T. Tsiodras, "TASTE: A Real-Time Software Engineering Tool-Chain Overview, Status, and Future," in *SDL 2011: Integrating System and Software Modeling*, 2012.

[5] J. Delange, C. Honvault, and J. Windsor, "Model-Based Engineering Approach for System Architecture Exploration," in *Embedded Real Time Software and Systems (ERTS)*, 2012.

[6] P. H. Feiler, D. P. Gluch, and J. J. Hudak, "The Architecture Analysis & Design Language (AADL): An Introduction," Carnegie Mellon University, Software Engineering Institute, 2006.

[7] International Telecomunication Union, "Information Technology - Abstract Syntax Notation One (ASN.1): Specification of Basic Notation," *International Standard X.680*, 2002.

[8] TASTE, "Kazoo," https://taste.tuxfamily.org/wiki/index.php?title=Kazoo, accessed November 2023.

[9] J. Hugues and J. Delange, "Model-Based Design and Automated Validation of ARINC653 Architectures Using the AADL," in *Cyber-Physical System Design from an Architecture Analysis Viewpoint: Communications of NII Shonan Meetings*. Springer, 2017, pp. 33–52.

[10] R. Han, B. Tine, J. Lee, J. Sim, and H. Kim, "Supporting CUDA for an Extended RISC-V GPU Architecture," in *Workshop on Computer Architecture Research with RISC-V (CARRV)*, 2021.

[11] F. Bellard, "QEMU, A Fast and Portable Dynamic Translator," in *USENIX annual technical conference, FREENIX Track*, vol. 41, 2005.

[12] SAE International, "ARP4754A - Guidelines for Development of Civil Aircraft and Systems," https://www.sae.org/standards/content/arp4754a/, accessed December 2023.

[13] European Cooperation for Space Standarization (ECSS), "ECSS-E-ST-40C - Software," https://ecss.nl/standard/ecss-e-st-40c-software-general-requirements/, accessed November 2023.

[14] IEEE Standards Association, "IEEE 1666-2011 - IEEE Standard for Standard SystemC Language Reference Manual," https://standards.ieee.org/ieee/1666/4814/, accessed November 2023.

[15] Modelica Association, "Functional Mock-Up Interface," https://fmi-standard.org/, accessed November 2023.

[16] Y. Hildenbrand, "ED-247 (VISTAS) Gateway for Hybrid Test Systems," SAE Technical Paper, 2018.

[17] Association for Standarization of Automation and Measuring Systems, "ASAM XIL," https://www.asam.net/standards/detail/xil/, accessed November 2023.

[18] G. Stazi, S. Sinisi, V. Di Valerio, C. Liu, D. Uttberg, L. Yapi, L. Lazzara, A. Ulisse, M. Baleani, and A. Mignogna, "Agile Model-Based Integration Framework for Advanced Software Validation and Verification," in *AIAA SCITECH 2023 Forum*, 2023.