DOI: https://doi.org/10.15276/aait.08.2025.21
UDC 004.094

Technological platforms, tools, and standardization in the unified information space for automated unmanned aerial vehicle design

Maksym D. Myrnenko[1)]
ORCID: https://orcid.org/0000-0002-5777-9583; maksym.myrnenko@gmail.com
Vladimir V. Shevel[1)]
ORCID: https://orcid.org/0000-0003-0534-0242; v.shevel@khai.edu. Scopus Author ID: 57211430745
[1)] National Aerospace University "Kharkiv Aviation Institute", 17, Vadym Mank Str. Kharkiv, 61000, Ukraine

ABSTRACT

Unmanned aerial vehicle development entails highly interdisciplinary processes, often hampered by fragmented digital toolchains and siloed data. This paper addresses the critical need for an integrated unified information space across the unmanned aerial vehicle lifecycle. We present a review of modern platforms and standards that enable end-to-end data continuity – the "digital thread" – from initial requirements through design, manufacturing, and operations. The aim is to synthesize current approaches to achieving interoperability and an authoritative source of truth for unmanned aerial vehicle projects. Key objectives include mapping prevalent Model-Based Systems Engineering and Product Lifecycle Management platforms, comparing lifecycle standards (e.g., systems modeling language, Standard for the Exchange of Product Data, Open Services for Lifecycle Collaboration, Open platform communications unified architecture) across development phases, proposing an integration framework to connect heterogeneous tools, and identifying gaps in adoption. In terms of methods, a structured literature review was conducted using IEEE Xplore, Scopus, and engineering databases. Search strings targeted unmanned aerial vehicle tool integration, Model-Based Systems engineering and Product Lifecycle Management interoperability, and digital thread case studies. Inclusion criteria focused on publications and standards addressing cross-domain data integration; industry white papers and standards documentation were also analyzed. The results reveal a rich landscape of standards for requirements management, system architecture, product data exchange, simulation interoperability, and IoT-based operational feedback. We provide a comparative analysis (including tables) of these standards versus lifecycle stages, and of integration approaches (Application Programming Interfaces, service buses, Open Services for Lifecycle Collaboration links, industrial Internet of Things protocols) against factors like latency, scalability, and security. A conceptual integration blueprint is outlined, leveraging open standards to connect systems modeling language models, Computer-Aided Design/Product Lifecycle Management data, and real-time sensor information into a cohesive environment. Discussion highlights trade-offs such as proprietary Product Lifecycle Management suites versus openness, the maturity and portability of Model-Based Systems Engineering models (systems modeling language version 1.0 limitations and systems modeling language version 2.0 prospects), and cloud vs. on-premises deployment given aerospace security International Traffic in Arms Regulations constraints. Organizational readiness (business process reengineering, stakeholder buy-in) emerges as a key success factor. In conclusion, unifying the information space can dramatically improve unmanned aerial vehicle development efficiency, traceability, and innovation. However, realizing this vision requires not only technical solutions but also adoption of standards and cultural change in engineering practices. The paper's synthesis provides practical insights and a roadmap for both researchers and practitioners aiming to implement a consistent digital thread for complex aerospace systems.

Keywords: model-based engineering; product lifecycle; digital thread; tool interoperability; unmanned aircraft; systems integration

***For citation*: Myrnenko M. D., Shevel V. V. "Technological platforms, tools, and standardization in the unified information space for automated unmanned aerial vehicle design". *Applied Aspects of Information Technology.* 2025; Vol.8 No.3: 316–333. DOI: https://doi.org/10.15276/aait.08.2025.21**

INTRODUCTION

Modern digital systems from e-government services to military unmanned aerial vehicles (UAVs) increasingly operate within unified information spaces that connect multiple components and stakeholders. A unified information space is essentially an integrated ecosystem of databases, networks, and interfaces governed by common rules, enabling seamless information exchange among organizations and users. For example, Ukraine's Diia digital platform provides over 130 state services through a single app, uniting citizens' documents and services in one place.

In defense, network- centric warfare approaches similarly rely on linking sensors, command systems, and units (e.g. UAVs) into one information network to achieve information superiority and faster decision cycles. A recent study describes a "Smart Factory" as a digital ecosystem where physical processes are integrated into a unified information space – automating product lifecycle management, leveraging big data, and integrating IoT and computing systems. Such integration boosts efficiency and capabilities, but it also expands the attack surface: adversaries can attempt to abuse

© Myrnenko M., Shevel V., 2025

This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/deed.uk)

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

legitimate workflows or data flows ("business-logic" abuse) in ways that traditional security controls might not easily detect. Business-logic attacks exploit the intended functionality of an application by chaining steps in a malicious sequence rather than injecting code or malware. They often appear as normal user behavior, thus evading typical security alarms. For instance, an attacker might repeatedly query a public registry via a government service to harvest sensitive data, or systematically request drone status updates to find anomalies, all while using valid credentials. These actions may leave subtle traces – e.g. unusual temporal patterns, abnormal request rates, or inconsistencies with user roles – but not overtly trigger intrusion detection. Such workflow abuses can have serious consequences in unified systems (fraud, privacy breaches, mission disruption). Traditional threat modeling approaches require deep human analysis of each business process, which is labor-intensive and may miss creative attack paths. Recently, large language models (LLMs) have shown promise in augmenting security analysis by synthesizing information and generating plausible scenarios. Large language models can reason through narratives and "think" like an attacker to propose unconventional misuse cases. Research suggests that LLMs could significantly accelerate threat modeling by automating scenario generation using their knowledge and logic capabilities. However, using generative AI for offensive security must be done responsibly. To avoid facilitating actual attacks, the AI should only produce descriptive hypotheses (no exploit code or specific payloads), and each AI-generated idea must be vetted by human experts for safety and validity. Human oversight is crucial – as noted in secure AI development guidelines, expert review and validation Help Bridge the gap between AI's capabilities and real-world security context. This article presents a methodology that harnesses LLMs to generate and validate defensive scenarios for systems operating in unified information environments (exemplified by Ukraine's Diia e-government portal, the Helios e-voting system, and a UAV life-cycle information system). The goal is to proactively discover complex attack patterns (particularly business-logic abuses and sequence anomalies) and devise detection and response measures for them – before adversaries exploit these tactics. By anchoring on public, well-documented workflows (login flows, ballot casting, UAV mission data handling, etc.), we ensure the focus remains on observable behavior rather than hidden vulnerabilities. We then leverage the creative breadth of AI to hypothesize adversary tactics, apply rigorous human curation and testing in isolated digital twin environments, and distill the findings into operational defense knowledge. The approach balances innovation with governance: the generative model injects creativity, while security experts maintain control through curation, testing, and enforcement of ethical boundaries. In the following sections, we detail each step – from scope definition and scenario generation to twin-based validation and operational integration – and we highlight results for each use-case (digital services and UAV systems). The importance of unified information space security in the UAV life cycle is underscored by our findings, as this domain showed unique patterns requiring specialized defensive responses.

REVIEW OF CURRENT PRODUCT LIFECYCLE MANAGEMENT TECHNOLOGIES

Product Lifecycle Management (PLM) platforms are designed for the centralized management of all product artifacts throughout its life cycle. They provide a single environment in which engineering specifications; product structure, documents, workflows, and change histories are stored and continuously maintained in an up-to-date state. In the aerospace sector, such systems are of particular importance, as they enable the consolidation of critical data and processes within one environment. Leading industrial PLM solutions (e.g., platforms offered by Siemens, PTC, or Dassault Systèmes) establish a single source of truth for product-related information, ranging from specifications and the Bill of Materials (BOM) to version control of engineering data, configuration planning, and requirements traceability [1].

Modern PLM systems emphasize collaborative teamwork and integration with adjacent development processes. They can interact with project management tools, Model-Based Systems Engineering (MBSE) environments, and modeling tools to ensure that all project participants operate with consistent data. In practice, a PLM platform functions as a shared digital workspace for both original equipment manufacturers (OEMs) and suppliers. Within such a multi-user environment, engineers from different organizations can simultaneously access current 3D models and specifications and update them in real time. For instance, when an engineer modifies a CAD model, the change becomes immediately visible to all stakeholders through the PLM system. This makes it possible to conduct preliminary design analyses at early stages: within an integrated CAD/CAE

environment (e.g., Siemens NX), engineers can perform initial simulations before handing the detailed model over to specialized analysts.

The primary advantages of PLM implementation include centralized data management and transparent control of complex projects across their entire life cycle. In essence, PLM establishes an end-to-end digital thread that connects all product-related information from conceptual design through to operational use [2]. This approach significantly enhances requirements and change traceability: each component or document is linked to corresponding versions and justifications for modifications. Such capabilities facilitate compliance with aerospace standards and support high product quality. At the same time, it must be noted that industrial PLM systems often require substantial financial investment and can be complex to configure for enterprise-specific needs. Many traditional solutions of this class are built on client-server architectures with proprietary data repositories, which may limit flexibility. As a result, integrating PLM platforms with other tools (CAD, requirements management systems, simulation environments, etc.) requires careful planning. Proper configuration and the use of open interfaces and standards are essential to maintaining long-term scalability and avoiding vendor lock-in.

Model-based systems engineering in unmanned aerial vehicle development

Model-Based Systems Engineering (MBSE) is an approach in which the primary artifact is a system model that formalizes requirements, functional relationships, behavior, and structure. In the context of UAV development, this implies representing the aircraft as a unified system, encompassing aerodynamic models, onboard software, and control logic. MBSE tools (such as IBM Rhapsody, Cameo Systems Modeler, and Sparx Enterprise Architect) are typically based on SysML or UML languages. These platforms allow systems engineers to immediately validate architectural consistency, generate structured requirement lists, and even configure parameters for simulation models. The advantages of MBSE include early detection of design errors through model-level validation, improved clarity of system design, and the ability to reuse architectural patterns. However, disadvantages include the significant upfront effort required to create comprehensive models, which increases early project costs, and the challenges of integrating MBSE models with existing tools – for example, translating system models into CAD environments. Furthermore, as MBSE remains a developing paradigm, standards for SysML model exchange (SysML v1/v2, XMI, etc.) have inherent limitations in portability and tool interoperability [3].

Beyond the immediate design benefits, MBSE enables more effective coordination across multidisciplinary teams involved in UAV development. By working on a shared model rather than disconnected documentation, engineers in aerodynamics, avionics, control systems, and production can synchronize their activities through a common framework. This reduces the risk of misinterpretation, ensures that requirement changes propagate consistently, and allows iterative refinement of the system throughout its lifecycle. Such a model-centric approach is particularly valuable for UAVs, which must integrate hardware, software, and human–machine interfaces under strict regulatory requirements. In this context, MBSE contributes not only to technical precision but also to compliance readiness, as structured models can serve as a foundation for certification documentation and traceability.

Another critical contribution of MBSE lies in its role within the unified information space [4]. When coupled with PLM systems, MBSE models can serve as authoritative references that link requirements directly to design components, simulation results, and certification artifacts. This connection forms the backbone of the digital thread, where each requirement or design element can be traced across the UAV lifecycle – from conception and simulation to production and operational feedback. Such integration strengthens the possibility of creating UAV digital twins, as behavioral and structural models developed in MBSE can be dynamically updated with operational data captured through IoT and PLM platforms.

At the same time, the adoption of MBSE requires a cultural and organizational shift. Teams accustomed to document-centric workflows often face a steep learning curve when transitioning to model-centric engineering. Effective implementation therefore requires investment in training, methodological adaptation, and tool integration strategies [5]. Despite these barriers, the long-term advantages are compelling: greater design agility, enhanced system reliability, and reduced costs associated with late-stage defect correction. In the UAV domain – where safety, performance, and rapid iteration are critical – MBSE provides a strategic foundation for digital transformation and sustainable competitiveness.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Computer-aided design/engineering

These platforms include three-dimensional design tools (such as CATIA, SolidWorks, Siemens NX, PTC Creo) and simulation environments (such as Ansys, Abaqus, MATLAB/Simulink, XFlow). Computer-Aided Design (CAD) systems are primarily responsible for creating and maintaining precise geometric representations of UAVs, including airframe structures, mechanical assemblies, wiring harnesses, and internal layouts. Computer-Aided Engineering (CAE) environments, in turn, provide computational analyses that support engineering decisions: structural mechanics simulations (via finite element analysis, FEA), aerodynamics (computational fluid dynamics, CFD), flight dynamics, and design optimization of configurations. By integrating CAD and CAE workflows, engineers can iteratively improve design fidelity and overall system quality throughout the UAV development lifecycle [6].

A critical challenge, however, lies in ensuring a unified information space that allows seamless exchange between CAD and CAE environments. The geometry produced in CAD systems must be correctly and consistently transferred into CAE tools to guarantee reliable simulations. Standardized exchange formats such as STEP and IGES, along with lightweight multi-view formats such as JT or Parasolid, are commonly employed for this purpose. Nevertheless, many CAD/CAE packages remain proprietary, leading to compatibility issues not only with one another but also with enterprise Product Lifecycle Management (PLM) systems. This lack of interoperability complicates version control, traceability, and integrated workflows across multidisciplinary teams. Consequently, enterprises often invest substantial effort in building middleware, custom connectors, or adopting open standards to preserve the integrity of the digital thread that links design geometry, analyses, and downstream processes.

Recent advancements in digital engineering are also reshaping CAD/CAE usage. One emerging trend is the incorporation of artificial intelligence and machine learning (AI/ML) techniques into engineering simulation workflows. Instead of relying exclusively on traditional physics-based CAE models, researchers and practitioners are increasingly exploring data-driven surrogates trained on large datasets of simulation or experimental results [7]. For example, machine learning models can be trained to approximate aerodynamic responses across a wide design space, providing near-instant predictions that would otherwise require computationally expensive CFD runs. This capability opens opportunities for real-time design optimization during conceptual phases or rapid configuration trade-offs, where thousands of design variants may be evaluated efficiently. Furthermore, AI-enhanced CAE workflows can support predictive maintenance and operational optimization: by linking simulation results with sensor data collected from UAV prototypes or digital twins, engineers can forecast failure modes or performance degradations before they occur [8].

In the broader context of a unified digital ecosystem, CAD and CAE tools are increasingly integrated not only with each other but also with PLM and MBSE platforms. For UAV development, this means that CAD models of the airframe and propulsion system can be directly associated with requirements in an MBSE repository, while CAE analyses of aerodynamic loads or structural stresses are automatically linked to verification criteria in PLM. Such end-to-end integration ensures that design modifications, simulation results, and system requirements remain consistent throughout the development process. Ultimately, the convergence of CAD/CAE with PLM, MBSE, and AI-driven methods contributes to a more agile and transparent UAV engineering process, reducing design cycles while maintaining traceability, quality, and compliance with aerospace standards [9].

Cloud services and IoT platforms

Modern engineering teams are increasingly adopting cloud computing and Internet of Things (IoT) platforms to perform complex computations, store large volumes of data, and integrate tools within a unified online environment [10]. Even enterprise-grade Product Lifecycle Management (PLM) systems are now offered in Software-as-a-Service (SaaS) mode, meaning they can be accessed via the Internet without requiring local installation. Several major vendors provide cloud-hosted variants of their PLM solutions, which lowers the entry barrier for implementation and reduces infrastructure costs. Cloud providers such as Amazon Web Services, Microsoft Azure, and Google Cloud also make it possible to run computationally intensive engineering workloads – such as large-scale computational fluid dynamics (CFD) simulations or multi-objective optimization studies – across dozens or even hundreds of processor cores, without the need for in-house high-performance computing clusters. Similarly, “collaborative CAD” services (for example, Onshape or Autodesk Fusion 360) allow multiple engineers to work simultaneously on

a 3D model through a web interface, removing the need for powerful local workstations [11].

The advantages of a cloud-based approach include scalability, flexibility, and accessibility. Computing resources such as CPU time, memory, and storage can be dynamically allocated based on demand, ensuring efficient use of resources. Cloud-based tools are accessible from any geographical location, which significantly facilitates collaboration across distributed teams. Automatic software updates managed on the provider's side further reduce administrative overhead and ensure that teams always work with the latest versions of their tools [12]. However, the use of public clouds also introduces notable challenges. Security and confidentiality are among the most critical: sensitive UAV project data hosted on third-party servers must be protected through robust encryption, rigorous authentication mechanisms, and well-defined access control policies. Additionally, dependence on stable Internet connectivity creates risks of downtime or latency issues, which can hinder engineering workflows. Finally, integration between cloud-hosted services and domain-specific aerospace tools sometimes requires non-standard adapters, middleware, or manual customization, which can complicate deployment.

Hybrid architecture is therefore often considered the most practical solution in aerospace and defense domains. In such setups, mission-critical data and systems – for example, requirement repositories or flight control models – are stored within on-premises infrastructures or private enterprise clouds, ensuring compliance with safety and regulatory standards [13]. At the same time, computationally intensive tasks such as large-scale analytics, telemetry processing, or the training of artificial intelligence models can be offloaded to public clouds, where engineers can take advantage of vast elastic resources. This hybrid approach balances the strengths of both worlds: secure control over sensitive UAV artifacts, combined with access to nearly unlimited computing power and innovative toolchains offered by cloud ecosystems.

IoT platforms play a key role in extending the capabilities of cloud-based environments. They provide the infrastructure for collecting, transmitting, and processing data directly from physical devices [14]. In UAV development, this is particularly valuable during prototyping and testing phases: a drone can stream telemetry data – including sensor readings, positional parameters, and system states – via secure connections directly into cloud databases. These real-time data streams can be immediately compared against the UAV's digital model stored in PLM or MBSE systems, thus forming a continuously updated digital twin of the aircraft. Such digital twins enable engineers to analyze discrepancies between expected and actual behavior, perform predictive maintenance, and refine system designs based on operational feedback. In effect, the integration of IoT with cloud platforms closes the loop between design, simulation, and operation, ensuring that digital engineering ecosystems remain synchronized with the evolving reality of UAV performance.

Integration Approaches. The construction of a unified information space for UAV engineering requires the integration of diverse software and hardware components, ranging from measurement devices and simulation engines to enterprise PLM repositories. Several architectural strategies are typically applied, including the use of Application Programming Interfaces (APIs), REST-based services, Service-Oriented Architecture (SOA) with Enterprise Service Bus (ESB), and industrial interoperability standards such as OPC UA. Each of these approaches provides specific benefits and limitations, and their coordinated application forms the foundation of a robust digital ecosystem [15].

API/REST. Modern PLM and MBSE platforms increasingly expose programmable interfaces – both proprietary and open – that enable automated data exchange. For example, RESTful APIs provide access to bill-of-materials (BOM) elements, requirement objects, or assembly structures through lightweight protocols such as HTTP combined with JSON or XML [16]. This design allows engineering teams to uniformly read and write information across heterogeneous sources. In the context of UAV development, such mechanisms facilitate direct connections between configuration management systems and computational modules: a PLM system can supply requirements or component definitions through an API call, while CAE services retrieve this information to prepare simulations. The principal advantage of API-driven solutions is their flexibility: new services can be added or integrated without redesigning the entire architecture. However, challenges include the necessity of maintaining backward compatibility across API versions, ensuring authentication and authorization mechanisms are robust, and establishing a shared semantic vocabulary so that data exchanged across APIs retains its meaning and context across platforms.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Service-oriented architecture and enterprise service bus. A more sophisticated level of integration is achieved through the adoption of Service-Oriented Architectures (SOA). Here, various services – whether offered by PLM repositories, MBSE modeling tools, simulation engines, or enterprise databases – interact through an Enterprise Service Bus (ESB). The ESB acts as a middleware broker that routes, transforms, and orchestrates data flows between systems. This approach encapsulates complex interdependencies within a centralized integration layer, where message transformations (for example, converting CAD structure data into requirement objects) are specified declaratively [17]. For UAV manufacturing, this model could be employed to connect IoT sensors on production lines with PLM-driven manufacturing schedules, ensuring that real-time sensor readings inform enterprise-level planning. SOA/ESB approaches scale well for large organizations and multi-site development programs. Nonetheless, they demand careful architectural planning and governance, and they may introduce latency due to multi-layer message routing. These trade-offs highlight the need for balancing architectural elegance with practical system performance in aerospace integration scenarios.

Open platform communications unified architecture (OPC UA). In industrial environments, Open platform communications unified architecture OPC UA has emerged as a key protocol for secure, vendor-neutral interoperability. It supports both client-server communication and publish-subscribe messaging patterns, enabling flexible connectivity across hardware, sensors, controllers, and enterprise software platforms. In UAV development, OPC UA can act as a bridge between physical systems and digital models [18]. For instance, during testing of a hybrid propulsion unit, sensors may stream telemetry data via OPC UA into a digital motor simulation, thereby supporting the creation of a high-fidelity digital twin. In this way, operational data from test environments can be seamlessly linked to virtual models in PLM or MBSE systems, providing engineers with a synchronized and realistic representation of the UAV. A notable strength of OPC UA is its adoption across industrial automation ecosystems and its emphasis on security, scalability, and backward compatibility [19]. By leveraging open standards such as OPC UA, engineering teams avoid vendor lock-in and achieve seamless integration of components from diverse origins.

Broader implications. The choice of integration approach is not purely technical; it has significant implications for organizational workflows, data governance, and long-term system sustainability. API/REST solutions often serve as the entry point for lightweight, project-specific integrations, while SOA/ESB architectures are better suited to enterprise-wide digital transformation programs that require centralized control and orchestration. OPC UA, on the other hand, ensures that cyber-physical systems – especially those involving UAV prototypes and IoT-enabled test facilities – remain tightly coupled to digital models and enterprise repositories. A hybrid strategy is increasingly seen as optimal, where lightweight APIs are combined with service-oriented backbones, and industrial standards guarantee interoperability at the physical layer. Such hybrid integration ecosystems enable UAV development teams to maintain a consistent digital thread across requirements, design, simulation, manufacturing, and operational feedback [20].

In sum, integration approaches form the methodological backbone of digital engineering in aerospace. By carefully combining APIs, service-oriented middleware, and industrial standards, organizations can establish a resilient and adaptable unified information space. This not only improves data consistency and collaboration among multidisciplinary teams but also ensures that UAV design and operation are supported by continuous feedback loops, enhancing both innovation and reliability across the entire lifecycle.

RESEARCH AIM AND OBJECTIVES

Problem statement. Current UAV programs operate with heterogeneous, weakly integrated toolchains – Model-Based Systems Engineering (MBSE), Product Lifecycle Management (PLM), Computer-Aided Design/Engineering (CAD/CAE), and Internet of Thing (IoT)/cloud services – which produce data silos, limited traceability, duplicated effort, and costly rework across lifecycle phases. Legacy lock-in and uneven adoption of open standards hinder an authoritative source of truth and a continuous digital thread from requirements to operations.

Gaps addressed by this review

1. G1. Interoperability gap. Lack of vendor-agnostic guidance on how to connect MBSE, PLM, CAD/CAE, and IoT into a unified information space.

2. G2. Standards-to-lifecycle mapping gap. Fragmented evidence on which standards (SysML,

ISO/IEC 15288, STEP AP242/AP239, FMI/SSP, OSLC, OPC UA) best support specific UAV lifecycle phases (requirements, architecture, CAD/PDM, simulation, manufacturing, operations).

3. G3. Integration blueprint gap. Absence of a consolidated, technology-agnostic integration pattern (API/REST vs. SOA/ESB vs. industrial protocols) with explicit trade-offs (latency, scalability, security, governance, lock-in).

4. G4. Adoption/governance gap. Limited synthesis of organizational prerequisites (business-process re-engineering, roles, data governance, security/compliance) that determine successful uptake.

Aim. To synthesize platforms, standards, and integration patterns into a coherent framework for a unified information space that supports end-to-end UAV lifecycle engineering.

Research questions

1. RQ1. Which platforms and open standards most effectively support interoperability and traceability across UAV lifecycle phases?

2. RQ2. What integration patterns (API/REST, SOA/ESB, OPC UA/IoT) minimize vendor lock-in while meeting performance and security constraints?

3. RQ3. How should standards be mapped to lifecycle activities to enable a consistent digital thread and digital twin?

4. RQ4. What organizational and governance practices (process, roles, policies) are necessary for sustainable adoption?

Hypotheses

1. H1. UAV programs adopting open standards (e.g., STEP, OSLC, FMI/SSP, OPC UA) achieve higher lifecycle traceability and lower integration effort than programs relying primarily on proprietary, closed ecosystems.

2. H2. A hybrid integration approach (API/REST for point flows, ESB for orchestration, OPC UA for OT/IoT) outperforms single-pattern architectures on scalability and governance without unacceptable latency penalties.

3. H3. Governance readiness (defined roles, business-process re-engineering, data-quality policies) is a necessary condition for realizing the benefits of a unified information space, independent of the chosen tool stack.

4. H4. Transition to SysML v2-enabled MBSE increases model portability and reduces custom integration code relative to SysML v1.x baselines.

Distinctive contributions. Unlike prior surveys of digital threads and PLM/MBSE that consider technologies in isolation or at a generic manufacturing level, this review offers UAV-specific advances.

1. UAV-specific lifecycle mapping: a systematic mapping of standards (SysML; ISO/IEC 15288; STEP AP242/AP239; FMI/SSP; OSLC; OPC UA) to UAV lifecycle phases (requirements, architecture, CAD/PDM, simulation, manufacturing, operations), exposing phase-appropriate coverage and gaps;

2. Vendor-agnostic integration blueprint: consolidation of API/REST, SOA/ESB, OSLC links, and industrial IoT protocols (OPC UA) into a single technology-agnostic blueprint, plus a trade-off matrix (latency, scalability, security, governance, lock-in) tailored to aerospace constraints;

3. Governance and adoption lens: an explicit account of organizational prerequisites (business-process re-engineering, role design, data governance, security/compliance) that condition successful uptake–an angle underdeveloped in earlier reviews;

4. Forward-looking MBSE perspective: examination of SysML v2 portability and model-exchange implications for reducing custom integrations relative to SysML v1.x baselines;

5. Conceptual UAV mini-case: a compact, non-proprietary illustration that links MBSE requirements to PLM/CAD artifacts and IoT-driven twin updates, demonstrating practical feasibility;

6. Transparent review method: a structured review methodology with explicit inclusion/exclusion criteria and synthesis procedures, improving reproducibility and auditability of the evidence base.

RESEARCH METHODOLOGY

Unified information space in the UAV lifecycle. One of the key concepts uniting MBSE, PLM, and digital engineering tools is the creation of a unified information space for the project. A unified information space is defined as an organizational and technical environment in which all product-related information is available in a centralized form to every participant involved across different phases of the product lifecycle. For UAV development projects, this is particularly critical, since multidisciplinary specialists are engaged: aerospace designers, avionics engineers, software developers, production technologists, testing experts, operators, and maintenance staff. All of these stakeholders must work with accurate and consistent data about the product [21].

In practical terms, the unified information space is realized through PLM platforms that integrate tools and data. Modern PLM systems (e.g., Siemens

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Teamcenter, Dassault Systèmes 3DEXPERIENCE, PTC Windchill) consolidate under a single "umbrella" diverse artifacts such as SysML models (system architecture defined in MBSE), CAD models of components and assemblies, CAE analysis results, manufacturing process definitions, requirements, and certification documentation. These heterogeneous data types are interconnected through the product structure and a unified configuration management system. As a result, a continuous digital thread emerges, extending across the entire UAV lifecycle.

The benefits of a unified information space for UAV development programs are multifold. First, it increases transparency and control: project managers or customers can obtain up-to-date information on work status, design changes, or test results directly from a single system, instead of gathering data from different departments. Second, it ensures requirements and change traceability: every design modification is automatically recorded in the system, with full tracking of the originating requirement or defect, and the individual who authorized the change. Third, it improves integration of production and operational stages: design data generated during early phases (such as a 3D model or specification) are reused directly in production planning and subsequently in maintenance – for example, through electronic spare part catalogs and repair manuals. Finally, the unified information space serves as a prerequisite for creating a digital twin of the UAV – a comprehensive virtual model reflecting the real state of the product during operation. This enables analysis and optimization at the usage stage, such as predicting component failures or optimizing maintenance schedules, thereby increasing the efficiency and reliability of UAV operations [22].

It is important to emphasize that implementing such an integrated environment is not solely a matter of software choice but also of organizational transformation. As researchers note, successful realization of CALS/PLM approaches requires reengineering of business processes and close collaboration among all project participants. Enterprises that fail to adopt modern integration technologies risk losing competitiveness and being unable to effectively cooperate in global supply chains. Conversely, companies that establish a unified information space gain significant advantages in development speed and product quality.

A practical example of these approaches is the construction of a semantic gateway between PLM and MBSE systems. Engineers may, for instance, deploy a REST API in an MBSE tool to export requirement specifications in XML format, which are then routed via an ESB to the PLM environment. There, a dedicated connector automatically generates corresponding requirement objects in the requirements management system (such as DOORS or Polarion). Similarly, the adoption of OSLC (Open Services for Lifecycle Collaboration) has become widespread for linking artifacts – for example, referencing SysML diagrams to test scenarios stored in validation systems [23].

Overall, UAV design integration must be grounded in open protocols and interfaces and in technology-agnostic solutions (data transformations, semantic alignment) that allow new modules to be connected without reworking the entire system. Particular emphasis is placed on aligning updated digital data with the "real world" through digital twins: the coupling of PLM/CAE systems with IoT devices is often realized via Industry 4.0 standards (e.g., OPC UA, MQTT). This ensures a bidirectional flow of information between physical UAV test environments and their digital models, supporting real-time synchronization and continuous improvement of aerospace systems [24].

Example in Practice: To illustrate how disparate tools can be knitted into a unified environment, consider a scenario from UAV development – linking an MBSE system with a PLM repository. Engineers might use a SysML modeling tool to capture system requirements and functional architectures. Instead of keeping those locked in the MBSE tool alone, a semantic gateway can push requirements into the PLM's requirements management module. For instance, a REST API provided by the MBSE tool can export the requirements in a structured format (XML), which is then picked up by an Enterprise Service Bus (ESB) and routed to the PLM system. A connector on the PLM side ingests this and creates native requirement objects in the PLM database (e.g., in a tool like IBM DOORS or Siemens Polarion which might be integrated with the PLM). Now those requirements are part of the unified space and can be linked to other entities (design parts, test cases). Similarly, industry has increasingly adopted the OSLC (Open Services for Lifecycle Collaboration) standard to enable deep linking between tools. For example, an architecture element in the SysML model (say a UAV subsystem) can be linked to a corresponding verification test case in a separate test management tool through an OSLC link. This way, a user in the MBSE tool can navigate to see the test results, and vice versa, without duplicating data. OSLC

essentially provides a web-like linking of artifacts across disparate software, embodying the unified information space concept at a metadata level.

Crucially, the unified information space must also bridge the physical–digital divide, especially as UAV programs move into prototyping and operations. This is where integration with Industrial IoT and Industry 4.0 frameworks comes in. Standards like OPC UA (OPC Unified Architecture) are often employed to connect physical sensors, machines, and test equipment to the digital models in real-time. For instance, during a ground test of a UAV's engine, telemetry can be fed via OPC UA into the digital twin model in the PLM/analysis environment, updating the model's parameters. OPC UA is particularly powerful here because it's an open, secure information exchange standard designed for interoperability of devices and systems in real-time industrial settings [13]. It allows creating a plug-and-play network where, say, a vibration sensor on the UAV test stand can publish data and any authorized software subscriber (like a digital twin analytics service) can consume it in a standardized way [14], [13]. This bidirectional flow – using IoT feeds to update models, and using digital models to send control or configuration updates to physical devices – cements the unified information space as not just a static repository but a living, synchronous representation of the UAV throughout its lifecycle.

In summary, a unified information space integrates tools, data, and people across all stages of UAV development. It embodies principles of single source of truth, data continuity, and cross-domain collaboration. Achieving it involves leveraging advanced PLM platforms and open standards, but also rethinking processes and ensuring organizational alignment. The next sections delve into the building blocks of such integration: the standards that make data interoperable and the approaches to connecting various software systems into one cohesive ecosystem.

Selection of the technology stack. The choice of a software–hardware stack for UAV design systems depends on multiple factors. Among the most important are safety and certification requirements, the volume of data to be processed, the necessity of integrating artificial intelligence and machine learning (AI/ML), the available project budget, and the experience and resources of the development team. Depending on these conditions, strategies may vary significantly – from the deployment of high-performance commercial platforms to reliance on more accessible open-source tools. Two contrasting scenarios illustrate this diversity [24].

For a large aerospace enterprise with sufficient budget, the priorities are reliability, compliance with industry standards, and scalability. Such organizations typically adopt well-established industrial solutions: commercial PLM systems (such as Siemens Teamcenter or PTC Windchill), powerful licensed CAD/CAE packages (e.g., CATIA for 3D design, Ansys for engineering analysis), and secure corporate cloud infrastructures. In such environments, adherence to strict certification requirements is critical – for example, avionics software must comply with DO-178C standards. Consequently, all tools and processes are configured with these certification criteria in mind. Infrastructure deployment is heavily focused on cybersecurity: user authentication is implemented in multiple layers, project data in storage is encrypted, and results are regularly backed up. To ensure that the platform can scale effectively under growing workloads, large enterprises increasingly adopt microservice architectures. Each component – PLM servers, CAD/CAE services, analytical modules, validation systems – is deployed in a separate container and orchestrated using platforms such as Kubernetes. This modularity means that the failure of a single subsystem does not paralyze the entire complex, while updates or scaling of any service can be performed independently. In practice, an enterprise may deploy a cluster of containers in its own data center, with dedicated container sets handling PLM functionality, others running analytical or computational services, while databases and file repositories are replicated into a private, S3-compatible storage system to improve resilience and scalability [25].

Smaller design offices or startups, by contrast, are often compelled to seek cost-effective solutions, even at the price of certain compromises. Such teams typically turn to free or open-source software and affordable commercial tools. For example, they may employ open CAD/CAE packages (FreeCAD for 3D modeling, OpenFOAM for aerodynamic CFD simulations), and for systems engineering use tools such as OpenModelica or Eclipse Papyrus to model system architectures. Design data management can be handled with simplified open-licensed PDM/PLM solutions, ranging from the free Aras Innovator Community Edition to custom wiki-based repositories with controlled access. Their IT infrastructure is frequently built on general-purpose public cloud services: collaborative work is supported by Google Workspace or Office 365,

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

version control is managed through GitHub or GitLab, and engineering analyses are conducted in cloud-hosted CAD/CAE services under pay-as-you-go models. This technology stack minimizes initial expenses for hardware and licenses but also carries limitations. Open tools are not always fully compatible with rigorous aerospace certification requirements, and small teams may lack the technical support and resources needed to address complex challenges. Public cloud services without specialized protections present confidentiality risks, while integration across heterogeneous open-source tools often requires additional configuration. To maintain a baseline level of security, such companies typically implement VPN access to critical resources, ensure that all applications are regularly updated, and rely on open exchange formats (such as STEP for 3D models or FMI for simulation modules). Adhering to open standards and protocols not only reduces immediate costs but also allows smoother integration into broader digital ecosystems and facilitates migration to more advanced solutions as the project scales.

KEY CRITERIA FOR TECHNOLOGY STACK SELECTION

Security and reliability. This is critical for UAVs as aerospace-grade systems. Measures include the use of secure communication channels (TLS, VPN), strict access control (role-based access control, multi-factor authentication), database encryption, and activity auditing. In cloud deployments, it is recommended to operate within secure environments (e.g., isolated VPCs, Dedicated Hosts). On-premises PLM platforms provide an additional level of control but require dedicated maintenance and internal resources.

Scalability. UAV design projects may suddenly require large numbers of parallel simulations (for example, to evaluate different wing configurations). For this reason, microservice-based architectures and elastic scalability (such as auto-scaling cloud clusters) are highly advantageous. In cases where large-scale resources are not initially needed, teams may rely on static resource allocations, though this approach carries the risk of delays once workloads increase.

Cost. Beyond licensing expenses, infrastructure development and maintenance costs must be carefully considered. Open-source technologies and cloud services can significantly reduce capital expenditures, but they introduce ongoing operational expenses. Strong dependency on a specific vendor (proprietary software) also increases the risk of escalating costs during system expansion.

AI/ML support. As modern engineering increasingly depends on large datasets (e.g., digital twins, operational analytics), integration of AI platforms becomes essential. This may require GPU clusters, AI/ML frameworks (such as TensorFlow or PyTorch), and supporting data pipelines and services (e.g., ELK stacks, time-series databases). The selected stack should include capabilities for intelligent model analysis and optimization – for instance, automatic calibration of aerodynamic parameters based on flight data.

Recommended architecture. A scalable UAV design platform may be best supported by a hybrid cloud architecture (combining private and public resources) with container orchestration. An on-premises PLM server linked with multi-cloud storage ensures controlled configuration management. MBSE tools can interface with PLM through APIs, while CAD/CAE environments leverage cloud-based simulations. A centralized repository of digital twins integrates outputs across these domains. Emphasis is placed on open integration and open protocols to support future extensions (for example, adding VR/AR modules or IoT analytics). This approach provides flexibility and scalability without significant increases in per-project costs, while also enabling the effective use of AI/ML technologies in the analysis and optimization of UAV design outcomes.

Standardization and interoperability

The consistency of data models and communication protocols is a cornerstone in building a unified information space for UAV design. At the international level, multiple standards cover system-level engineering, lifecycle processes, and data exchange formats.

- **ISO/IEC 15288:2015 "Systems and Software Engineering – System Life Cycle Processes"** – establishes general lifecycle processes for system development (requirements, architecture, integration, verification) regardless of domain.
- **ISO/IEC/IEEE 42010:2011 (formerly IEEE 1471) "Systems and Software Engineering – Architecture Description"** – defines the concept of architectural frameworks and system architecture descriptions, essential for MBSE processes.
- **ISO 10303 (STEP)** – the Standard for the Exchange of Product Data. Different Application Protocols (APs) support specific use cases, e.g., AP242 for managed 3D model-based engineering,

AP233 for SysML artifacts. These ensure standardized text-based (ASCII/XML) representations of 3D models and product structures, enabling interoperability between CAD and PLM systems. Of note, ISO 10303-233 ("PLM and Product Data Integration") and AP239 ("Product Life Cycle Support") were designed specifically for PLM-related processes.

- **FMI (Functional Mock-up Interface)** – an open standard for the exchange of simulation models. FMI enables the export of Functional Mock-up Units (FMUs) that encapsulate system behavior (dynamic or physical models), which can then be imported into other simulators. This is critical in MBSE workflows where one team produces a device model (e.g., an engine) and another integrates it into the system-level simulation.
- **SSP (System Structure and Parameterization)** – a lightweight XML-based standard for describing interconnected networks of models, system parameters, and signal flows. SSP extends FMI, allowing the description of entire system structures with subsystem hierarchies and parameter linkages. This is especially useful for UAV modeling, where aerodynamic, control, and avionics subsystems must be combined into one integrated topology with shared parameters.

UAV/UAS Standards. Specialized guidance is provided by organizations such as ICAO, EASA, and SAE. For example, SAE AS-01 (unmanned systems) and DO-178C/ED-12 (certification of avionics software) apply directly to unmanned systems. The ANSI UASSC in the United States issues a comprehensive standards roadmap for UAS. In Europe, ISO/TC 20/SC 16 actively develops UAV/UAS standards covering classification, safety, and certification. At the national level, bodies such as Ukraine's UAAP are working to harmonize international standards within domestic frameworks. Interoperability is thus achieved through unified models and dictionaries, from structured specifications (e.g., digital mockups and product structures exchanged via STEP) to integration of requirement and PLM data through OSLC or standardized CSV formats.

Interaction models. Achieving interoperability requires frameworks that link disparate data formats. The Digital Thread is a widely recognized concept, providing continuous linkage between all lifecycle phases of a product. In this model, data become universally accessible – requirements, CAD sketches, and test results are connected through unique identifiers and metadata. For example, flight control requirements defined in MBSE can be reflected as attributes in PLM, while flight test data are captured and automatically fed back into analytical systems, forming a closed-loop digital thread. Another example is OSLC (Open Services for Lifecycle Collaboration), which provides semantic linking of artifacts across tools (e.g., linking a SysML requirement to a corresponding PLM test case), ensuring traceability across heterogeneous platforms.

Data formats. In addition to STEP, XML/JSON (e.g., OSLC JSON bindings) and FMI/SSP have become critical exchange mechanisms. Within MBSE ecosystems, SysML-based exchanges (e.g., XMI exports) enable the transfer of requirements and diagrams between modeling tools. For 3D geometry, industrial formats such as JT or IFC are commonly used alongside STEP. FMI, in particular, facilitates the inclusion of third-party simulation modules into an overall UAV system model, while SSP supports configuration and packaging of such FMUs into a cohesive simulation environment.

It is also important to note that NATO, NIST, and DARPA have developed technical roadmaps for digital engineering and MBSE. For example, NIST (2017) emphasized the Digital Thread as the key enabler for seamless lifecycle integration across design, manufacturing, and service phases. The alignment of such models complements existing international standards such as ISO/IEC 12207 (software engineering processes) and ISO 5001 (configuration management). A critical role is played by shared vocabularies and attribute standards, ensuring that data elements (e.g., "wing height") retain consistent meaning across systems. Only by adhering to unified exchange standards (e.g., STEP AP242 for 3D models, FMI v2.0 for dynamic models, SSP 2.0 for system configurations) and internationally recognized architectural frameworks can the realization of a UAV unified information space be both effective and future-proof.

RESULTS OF THE STUDY

Overview of findings. Results of the review indicate that a unified information space for UAV programs is technically attainable when lifecycle standards are aligned to phase-specific roles (Table 1) and integration patterns are selected with explicit trade-offs in mind (Table 2). ISO/IEC 15288 and ISO/IEC/IEEE 42010 structure requirements and architecture; STEP AP242/AP239 bridge CAD geometry and product structure into PLM; FMI/SSP

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

enable multi-domain co-simulation; OSLC provides artifact-level traceability; OPC UA carries operational/bench telemetry into the digital twin. Comparing integration patterns, API/REST suits low-latency point flows, SOA/ESB improves orchestration and governance at the cost of overhead, while industrial IoT (IIoT)/OPC UA anchors OT connectivity under security and compliance constraints. Across sources, three adoption determinants recur: (i) governance and roles (business-process re-engineering, data policies, security/ITAR), (ii) semantic alignment (common vocabularies, IDs, units), and (iii) vendor-agnostic linking (OSLC) to mitigate lock-in. These results directly address RQ1–RQ4 and motivate the illustrative pilot below.

Representative cases from practice. In one representative case-the development of a heavy unmanned helicopter-a continuous digital thread and a digital twin were implemented through deep integration of MBSE and PLM. The engineering team effectively created a cyber-physical test bench: the helicopter's digital twin served as a virtual replica of the aircraft, continuously updated in real time using sensor data. During flight tests, telemetry from onboard sensors was transmitted to a cloud environment and automatically adjusted parameters of the 3D model, which was then verified by analytical modules. In parallel, all information about system composition and component versions was recorded in PLM. For instance, a flight-control algorithm developed in Simulink was exported to the PLM environment as a configuration element. Using FMI (Functional Mock-up Interface) and SSP (System Structure and Parameterization), this software module was combined with corresponding physical models (e.g., the engine model imported as an FMU component, the hydraulic system described in SSP format) to perform comprehensive flight co-simulation. The outcome was an integrated digital platform that encompassed requirements, design, and experimental flight data within a single simulation cycle. Design iterations accelerated markedly: each new component version (e.g., a propeller) created in CAD was automatically transferred in STEP format to CAE analysis, while PLM integration tools logged its attributes and triggered an updated test scenario in the digital test bench. This shortened the time from concept to first flight tests, while the digital twin ensured that the virtual model consistently reflected the state of the physical prototype.

In another case, advanced digital technologies were applied in a related field-monitoring of infrastructure assets. For example, an energy company used UAV-based photogrammetry to generate high-precision 3D models of existing structures (such as bridges). Although not directly related to UAV design, this methodology can be adapted as part of the digital-twin framework. The concept relies on reverse data flow: large volumes of empirical measurements collected by drones (e.g., point clouds, real geometry parameters) are imported via standardized formats into PLM or MBSE environments as factual data for comparison with design models. For instance, one study proposed a method to validate a 3D bridge model generated from UAV imagery against laser scanning (TLS) data of the same structure-allowing verification of the digital twin against real-world measurements. Similarly, in UAV design itself, data collected during flight tests can automatically update system-control models or aerodynamic representations. If a UAV demonstrates unexpected deviations during a maneuver, these telemetry records can be imported into a high-level simulator (e.g., a city-scale environment simulation) for analysis and subsequent refinement of the autopilot mathematical model in the PLM system.

Illustrative Example (Conceptual UAV Pilot). To operationalize the integration blueprint, we outline a conceptual pilot that links MBSE requirements to PLM/CAD/CAE artifacts and a runtime digital twin using open standards. A performance requirement (R-001) is modeled in SysML and OSLC-linked to controlled PLM items and a CAD assembly. Geometry and product structure are exchanged via STEP AP242/AP239; dynamic components are packaged as FMI units and composed with SSP for system-level co-simulation. During bench tests, telemetry (speed, torque, temperature) is streamed over OPC UA to the twin, which compares observed behavior with simulated envelopes and issues an OSLC finding if deviations exceed thresholds. PLM governs change (ECR/ECO) with full traceability back to R-001 and forward to updated CAD/CAE runs. Governance hooks include versioning, role-based access control (RBAC), e-signatures, and minimal data-quality gates at each handoff. Reportable KPIs (as classes, not raw numbers) include link coverage (requirement→design→test), propagation latency classes (MBSE→PLM→CAD/CAE→twin), and defect interception points (simulation vs. bench). This compact walk-through demonstrates feasibility of a standards-based digital thread without exposing proprietary internals (see Table 1 for standards-to-lifecycle mapping and Table 2 for integration trade-offs).

Table 1. **Standards mapped to UAV lifecycle phases and roles**

Standard / Technology	Require-ments	Architecture	CAD / PDM	Simulation	Manufactu-ring	Opera-tions / MRO	Traceability / Governance	Notes
ISO/IEC 15288 (System life cycle processes)	Framework for elicitation & management	Process basis for architectural work	–	–	Production & integration processes	Opera-tion & maintena nce processes	Governance, reviews, configuration	Lifecycle process reference; not a data format
ISO/IEC/IEE E 42010 (Architecture description)	–	Primary (viewpoints, stakeholders, consistency)	–	–	–	Indirect (architect ure for ops)	Cross-viewpoint consistency & traceability	Defines how to document system architecture
SysML (v1.x / v2)	Primary (require-ments, parametrics)	Primary (blocks, interfaces, allocations)	Indirect (links to PDM items)	Via bindings to FMI/SSP or adapters	–	–	Traceability via stereotypes / OSLC links	v2 improves portability and API access
STEP AP242 (Managed model-based 3D)	–	Limited (references to structures)	Primary (geometr y, PMI, product structure)	Supports mesh/valid ation handoff	Primary (down-stream manufac-turing, MBD/PMI)	Partial (handoff to PLCS)	Stable IDs enable cross-tool linking	Neutral CAD/PDM exchange; MBD/PMI
STEP AP239 (PLCS)	–	Product structure / configuration context	Primary (PDM/ PLM base-lines, config)	–	Supports manufac-turing configu-ration	Primary (MRO, as-main-tained)	Strong configuration & change traceability	Product Lifecycle Support (PLCS)
OSLC (Open Services for Lifecycle Collabora-tion)	Primary (require-ments links)	Primary (linking viewpoints/ artifacts)	Primary (linking PDM, CAD, docs)	Primary (linking tests & models)	Supports links to MES/QMS	Primary (ops/test artifact linkage)	Semantic, REST-based traceability	Resource linking across tools; vendor-agnostic
FMI 2.0 (Functional Mock-up Interface)	–	Interfaces/co ntracts for components	–	Primary (FMU exchange for co-simulation)	–	Supports twin parame-ter updates	Versioned FMUs, manifests	Executable model interface; tool-neutral
SSP 2.0 (System Structure & Parameteri-zation)	–	Composition of subsystems	–	Primary (topology, signals, parameters)	–	–	Package-level metadata	Coordinates multiple FMUs in a system
OPC UA (industrial connectivity)	–	–	–	Supports HIL/SIL streaming	Shop-floor integration	Primary (tele-metry for test/ops)	Built-in auth, certs, audit	OT/IIoT protocol for secure telemetry
MQTT (lightweight messaging)	–	–	–	–	–	Alternate telemetry for constrai-ned links	Needs external security/enforce ment	Publish/subsc ribe; simple, lightweight
ISO 8000 (Data quality)	Requirement data quality policies	–	Master data quality in PDM/ PLM	–	–	–	Data quality governance	Quality rules for identifiers, units, vocabularies

Source: **compiled by the authors**

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Table 2. **Integration approaches versus engineering constraints and typical UAV usage**

Integration pattern	Latency	Scalability	Orchestration / Governance	Security / Compliance	Lock-in risk	Typical UAV usage	Limitations
API / REST (HTTP/JSON)	Low (ms–sub-s)	Good point-to-point; horizontal scale	Limited; relies on client code	TLS, OAuth; app-layer policies	Low (open)	Quick tool-to-tool exchanges; microservices	Versioning drift; bespoke glue code
OSLC resource links	Low–medium	Good (distributed link graph)	Strong artifact governance via link types	HTTPS + delegated auth; audit via tools	Low (open)	Traceability across MBSE/PLM/ CAD/test	Needs provider support; semantic alignment
SOA / ESB	Medium (routing/mediation overhead)	High (central mediation/ routing)	Strong (policies, transforms, monitoring)	Centralized enforcement	Medium (platform-dependent)	Cross-tool orchestration; mediation	Complexity; single point of governance
OPC UA (Client/Server + PubSub)	Low–medium (deterministic options)	High for OT/IIoT topologies	Namespaces, roles, audit trails	Built-in certs, signing, encryption	Low–medium	Bench/flight telemetry; HIL/SIL; shop-floor	Heavier than MQTT; namespace modeling
MQTT (Pub/Sub)	Low	High (broker-based)	External (broker policies/ topics)	Needs TLS + IAM add-ons	Low (open)	Lightweight telemetry; constrained links	Limited semantics; weaker governance
File-based batch (STEP, CSV, XML)	High (batch)	High (asynchronous)	Governed by release/ baselines	Repos + signatures	Low (open formats)	CAD→PLM handoffs; supplier exchange	Not real-time; risk of stale data
Event streaming (Kafka etc.)	Low	Very high (distributed)	Schema registry; topic policies	TLS/SASL; access control	Medium (platform)	Telemetry pipelines; analytics feeds	Operational burden; platform skills
gRPC (binary RPC)	Very low	Good (service mesh)	Service-level policy via mesh	mTLS; mesh policies	Low–medium	High-performance service calls	Polyglot clients; firewall traversal

Source: **compiled by the authors**

Synthesis and implications. The pilot operationalizes our standards mapping and integration blueprint end-to-end, showing how requirements (SysML) stay traceable to CAD/PLM items, co-simulation artifacts (FMI/SSP), and twin-driven findings, with OSLC links preserving provenance. In practice, teams should (i) choose REST for low-latency point integrations and ESB for cross-tool orchestration, (ii) use STEP AP242/AP239 as the neutral conduit between CAD/PLM/CAE, (iii) stream test/ops data via OPC UA into the twin, and (iv) enforce governance checkpoints (versioning, RBAC, e-signatures, data-quality gates). Residual limitations concern semantic drift across tools, SysML v1.x portability (mitigated as SysML v2 matures), and cloud vs on-prem constraints for sensitive programs. These trade-offs are examined next in the Discussion section.

CONCLUSIONS

This review examined how a unified information space for unmanned aerial vehicle (UAV) programs can be established by aligning lifecycle standards (ISO/IEC 15288; ISO/IEC/IEEE 42010; SysML; STEP AP242/AP239; FMI/SSP; OSLC; OPC UA) with phase-specific roles and by selecting integration patterns (API/REST, SOA/ESB, industrial IoT) with explicit trade-offs. We synthesized a vendor-agnostic blueprint and validated its plausibility through a conceptual pilot that links MBSE requirements to PLM/CAD/CAE artifacts and a runtime digital twin using open interfaces. Across sources, three determinants consistently condition success: (i) governance and roles (business-process re-engineering, data policies, security/ITAR), (ii) semantic alignment (common

vocabularies, identifiers, units), and (iii) vendor-neutral traceability via OSLC to mitigate lock-in.

Contributions to the research community. Lifecycle mapping and taxonomy. The paper systematizes how major standards span requirements, architecture, CAD/PDM, simulation, manufacturing, and operations, clarifying coverage and gaps across UAV-specific phases rather than generic manufacturing contexts.

Integration blueprint with trade-off matrix. We consolidate API/REST, SOA/ESB, OSLC links, and industrial protocols (OPC UA/MQTT) into a single, technology-agnostic pattern and articulate trade-offs (latency, scalability, security, governance, lock-in), enabling reproducible comparison across studies.

Governance lens for digital threads. Beyond tooling, we foreground organizational readiness-roles, data quality gates, change control, and assurance-providing a structured frame that prior surveys often underdevelop.

Forward-looking MBSE portability. We discuss the implications of SysML v2 for model exchange and reduction of custom glue code relative to SysML v1.x, outlining hypotheses and testable indicators for future empirical work.

Method transparency and evaluation cues. The review method is stated explicitly, and we propose practical KPIs/classes (link coverage, propagation latency, defect interception points) that researchers can operationalize without sensitive data. Together, these items strengthen reproducibility and cumulative evidence building.

Contributions to aerospace practitioners. Actionable implementation path. The blueprint converts standards into concrete handoffs: SysML (requirements/architecture) → OSLC links → PLM/CAD (STEP AP242/AP239) → CAE co-simulation (FMI/SSP) → telemetry streams (OPC UA) → digital-twin findings → governed ECR/ECO closure. The text-only sequence offers a ready-to-apply checklist where figures are not practical.

Minimal viable digital thread. We specify a staged approach-start with OSLC-based traceability and STEP handoffs; add FMI/SSP for co-simulation; integrate OPC UA for bench/ops data-so teams can realize quick wins before full orchestration via ESB.

Risk reduction and compliance. Concrete guidance is provided on avoiding vendor lock-in, enforcing versioning/RBAC/e-signatures, and balancing cloud vs on-prem deployments under ITAR-like constraints-frequent blockers in aerospace adoption.

Procurement and governance cues. The trade-off matrix informs RFP language (require OSLC providers/consumers; neutral STEP exchanges; FMI/SSP compatibility) and internal policy (data quality gates, baseline audits), accelerating cross-supplier interoperability.

Limitations. This is a review with a conceptual pilot; no live experimental infrastructure or proprietary datasets were exercised. Performance observations are framed as classes rather than measurements. Generalizability depends on tool maturity (e.g., SysML v2 availability), security posture, and organizational readiness. These limitations are typical for a review but should be addressed in follow-on studies.

Future work. We identify five priorities: (i) multi-organization pilots that benchmark REST vs ESB orchestration under realistic loads and security constraints; (ii) empirical studies of SysML v2 migrations and their effect on integration effort and defect interception; (iii) shared ontologies and unit/ID registries to curb semantic drift across MBSE/PLM/CAD/CAE/twin environments; (iv) longitudinal evaluations of governance interventions (data-quality gates, role designs) on rework, lead time, and auditability; and (v) cybersecurity of the digital thread, including threat modeling and assurance for OSLC endpoints, secure OPC UA profiles and key management, signed/SBOM-managed artifacts for STEP/AP239/FMI/SSP packages, zero-trust access (RBAC/MFA) across cloud and on-prem segments, and continuous monitoring of twin/IoT telemetry paths. Advancing along these lines will turn today's vendor-agnostic blueprint into stable, repeatable operating patterns for complex UAV programs.

In conclusion, unifying the information space for UAV engineering is both feasible and beneficial when standards, integration patterns, and governance are co-designed. For scholars, this paper provides a structured synthesis, a UAV-specific taxonomy, and testable propositions; for practitioners, it offers a pragmatic roadmap to implement a standards-based digital thread that improves traceability, reduces rework, and accelerates iteration across the UAV lifecycle.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

REFERENCES

1. Hossain, N. U. I., Lutfi, M., Ahmed, I., Akundi, A. & Cobb, D. "Modeling and analysis of unmanned aerial vehicle system leveraging systems modeling language (SysML)". *Systems*. 2022; 10 (6): 264. DOI: https://doi.org/10.3390/systems10060264.

2. Gerhard, D., Salas Cordero, S., Vingerhoeds, R., et al. "MBSE-PLM Integration: Initiatives and future outlook". In: *F. Noël et al. (Eds.), Product Lifecycle Management. PLM in Transition Times (IFIP AICT*. 2022; 661: 165–175. DOI: https://doi.org/10.1007/978-3-031-25182-5_14.

3. Gough, K. M. & Phojanamongkolkij, N. "Employing model-based systems engineering on a NASA aeronautic research project: A case study". *Aviation Technology, Integration, and Operations Conference*. 2018. DOI: https://doi.org/10.2514/6.2018-3361.

4. Lopez, V. & Akundi, A. "Modeling a UAV Surveillance Scenario – an applied MBSE approach". In: *IEEE International Systems Conference (SysCon)*. 2023. p. 1–7. DOI: https://doi.org/10.1109/SysCon53073.2023.10131074.

5. Torrigiani, F., Deinert, S., Fioriti, M., Pisu, L., Liscouet-Hanke, S. & Jungo, A. "An MBSE certification-driven design of a UAV MALE configuration in the AGILE 4.0 design environment". *AIAA SciTech Forum*. 2021. DOI: https://doi.org/10.2514/6.2021-3080.

6. Zhang, L., Chen, Z., Laili, Y., Ren, L., Deen, M. J., et al. "MBSE 2.0: Toward more integrated, comprehensive, and intelligent MBSE". *Systems*. 2025; 13 (7): 584. DOI: https://doi.org/10.3390/systems13070584.

7. Kabashkin, I. "Digital twin framework for aircraft lifecycle management based on data-driven models". *Mathematics*. 2024; 12 (19): 2979. DOI: https://doi.org/10.3390/math12192979.

8. Duverger, E., Aubry, A., Levrat, E. & Arista, R. "Early concurrent engineering in the aerospace industry supported by a digital thread framework". *IFAC-PapersOnLine*. 2024; 58 (19): 510–515. DOI: https://doi.org/10.1016/j.ifacol.2024.10.111.

9. Perikleous, D., Koustas, G., Velanas, S., Margariti, K., Velanas, P. & Gonzalez-Aguilera, D. "A novel drone design based on a reconfigurable unmanned aerial vehicle for wildfire management". *Drones*. 2024; 8 (5): 203. DOI: https://doi.org/10.3390/drones8050203.

10. Chen, X., Isoldi, A., Riaz, A. & Mourouzidis, C. "Evaluation of a collaborative and distributed aircraft design environment, enabled by microservices and cloud computing". *AIAA SciTech*. 2023. DOI: https://doi.org/10.2514/6.2023-1163.

11. Wu, B., Li, Y., Zhang, X. & Chen, H. "Review of the application of UAV edge computing in fire rescue". *Journal of Cloud Computing*. 2023; 12 (1): 87. DOI: https://doi.org/10.1186/s13677-023-00412-9.

12. Valencia, E., Toapanta, F., Oña, G., Carrillo, A., Aláez, D., et al. "An open-source UAV digital twin framework: A case study on remote sensing in the Andean mountains". *Journal of Intelligent & Robotic Systems*. 2025; 111: 71. DOI: https://doi.org/10.1007/s10846-025-02276-7.

13. de Longueville, S., Bouvet, C., Bénard, E., Jézégou, J. & Gourinat, Y. "Digital thread-based optimisation framework for aeronautical structures: A vertical tail plane use case". *Aerospace*. 2025; 12 (1): 2. DOI: https://doi.org/10.3390/aerospace12010002.

14. Pliakos, C., Efrem, G., Terzis, D. & Panagiotou, P. "An automated framework for streamlined CFD-based design and optimization of fixed-wing UAV wings". *Algorithms*. 2025; 18 (4): 186. DOI: https://doi.org/10.3390/a18040186.

15. Coïc, A., Reichwein, A. & Wildermann, S. "Linking design requirements to FMUs to create LOTAR-compliant MBSE models". *15th International Modelica Conference*. Aachen, Germany. 2023. p. 1–8.

16. Łukaszewicz, A., Szafran, K. & Józwik, J. "CAx techniques used in UAV design process". In: *Proc. of IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*. 2020. p. 95–98. DOI: https://doi.org/10.1109/MetroAeroSpace48742.2020.9160091.

17. Syrotynskyi, T., Kolesnyk, K., Kozemchuk, I. & Holovatyy, A. "3D modelling of UAV and creating its system of control". *Computing, Dynamics, and Systems*. 2024; 6 (3): 17–23. DOI: https://doi.org/10.23939/cds2024.03.017.

18. Adams, K. M., Ibrahim, I. & Krahn, S. "Engineering systems with standards and digital models: Development of a 15288–SysML grid". *Systems*. 2024; 12 (8): 276. DOI: https://doi.org/10.3390/systems12080276.

19. Andersson, E. & Broberg, E. “Evaluating the FMI and SSP standards for aerospace co-simulation”. *Master’s thesis, Mälardalen University (DiVA Portal)*. 2025.

20. “Open services for lifecycle collaboration. OSLC core specification 3.0”. *OASIS Standard*. 2021.

21. Zhang, T., Grzelak, D., Zhao, W., Islam, M. A., Fricke, H. & Aßmann, U. “A review on the construction, modeling, and consistency of digital twins for advanced air mobility applications”. *Drones*. 2025; 9 (6): 394. DOI: https://doi.org/10.3390/drones9060394.

22. Allen, S., Hossain, M., Rehmani, H. & Crespi, N. “Performance analysis of OPC UA for industrial interoperability towards Industry 4.0”. *IoT*. 2022; 3 (4): 27. DOI: https://doi.org/10.3390/iot3040027.

23. Łukaszewicz, A., et al. “Review on type of sensors and detection method of anti-collision system of UAV”. *Sensors*. 2023; 23 (15): 6810. DOI: https://doi.org/10.3390/s23156810.

24. Cresswell, A. & Anjos, D. “OSLC and the future of interoperability (Webinar)”. *Jama Software Whitepaper*. 2021.

25. Banerjee, O. “The future of PLM openness: From REST APIs to agentic data (Blog)”. *Beyond PLM*. 2022.

Conflicts of Interest: Author declare that he has no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article

Received 04.08.2025
Received after revision 22.09.2025
Accepted 26.09.2025

DOI: https://doi.org/10.15276/aait.08.2025.21
УДК 004.094

Технологічні платформи, інструменти та стандартизація в єдиному інформаційному просторі автоматизованого проєктування безпілотних літальних апаратів

Мирненко Максим Дмитрович [1)]
ORCID: https://orcid.org/0000-0002-5777-9583; maksym.myrnenko@gmail.com
Шевель Володимир Вікторович [1)]
ORCID: https://orcid.org/0000-0003-0534-0242; v.shevel@khai.edu. Scopus Author ID: 57211430745
[1)] Національний аерокосмічний університет «Харківський авіаційний інститут», вул. Вадима Манька, 17. Харків, 61000, Україна

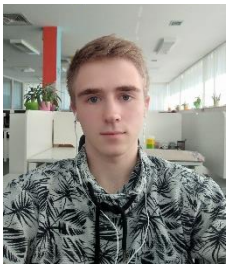
АНОТАЦІЯ

Розроблення безпілотних літальних апаратів є високодисциплінарним процесом, який нерідко ускладнюється фрагментованими цифровими ланцюжками інструментів і ізольованими даними. У цій статті розглядається критична потреба у створенні інтегрованого єдиного інформаційного простору на всіх етапах життєвого циклу безпілотних літальних апаратів. Подано огляд сучасних платформ і стандартів, що забезпечують наскрізну безперервність даних – «цифрову нитку» – від початкових вимог до проєктування, виробництва та експлуатації. Метою є синтез актуальних підходів до досягнення інтероперабельності та формування єдиного авторитетного джерела істини для проєктів безпілотних літальних апаратів. До ключових завдань належать: картографування поширених платформ Model-Based Systems Engineering та Product Lifecycle Management; порівняння стандартів життєвого циклу (напр., systems modeling language, Standard for the Exchange of Product Data, Open Services for Lifecycle Collaboration, Open platform communications unified architecture) між фазами розробки; пропозиція інтеграційної схеми для поєднання різнорідних інструментів; виявлення прогалин у впровадженні. Щодо методів, виконано структурований огляд літератури з використанням баз Institute of Electrical and Electronics Engineers Xplore, Scopus та профільних інженерних ресурсів. Пошукові запити фокусувалися на інтеграції інструментів для безпілотних літальних апаратів, взаємодії Model-Based Systems Engineering та Product Lifecycle Management на прикладах реалізації «цифрової нитки». Критеріями включення були публікації та стандарти, що безпосередньо стосуються міждисциплінарної інтеграції даних; також проаналізовано промислові оглядові матеріали (white papers) і нормативні документи. Результати демонструють насичений ландшафт стандартів для керування вимогами, опису системної архітектури, обміну конструкторськими даними, інтероперабельності симуляцій і зворотного зв’язку на базі IoT під час експлуатації. Наведено порівняльний аналіз (включно з таблицями) відповідності стандартів фазам життєвого циклу,

а також зіставлення інтеграційних підходів (Application Programming Interface, сервісні шини, Open Services for Lifecycle Collaboration, промислові Internet of Things протоколи) за критеріями затримок, масштабованості та безпеки. Окреслено концептуальну схему інтеграції, що спирається на відкриті стандарти для зв'язування моделей systems modeling language, даних Computer-Aided Design/Product Lifecycle Management і телеметрії датчиків у реальному часі в єдиному узгодженому середовищі. В обговоренні висвітлено ключові компроміси: між пропрієтарними Product Lifecycle Management комплексами та відкритістю; між зрілістю й переносимістю моделей Model-Based Systems Engineering (обмеження systems modeling language version 1.0 і перспективи systems modeling language version 2.0); між хмарним і локальним розгортанням з огляду на вимоги безпеки авіакосмічної галузі. Як визначальний чинник успіху виокремлюється організаційна готовність (реінжиніринг бізнес-процесів, залучення стейкхолдерів). У підсумку, уніфікація інформаційного простору здатна суттєво підвищити ефективність, простежуваність і інноваційність розроблення безпілотних літальних апаратів. Водночас реалізація цього бачення потребує не лише технічних рішень, а й послідовного впровадження стандартів і культурних змін у практиках інженерної діяльності. Синтез, запропонований у статті, надає практичні орієнтири та дорожню карту для дослідників і практиків, які прагнуть вибудувати послідовну «цифрову нитку» для складних аерокосмічних систем.

Ключові слова: модельне проєктування; життєвий цикл; цифрова нитка; інтероперабельність інструментів; безпілотник; інтеграція систем

ABOUT THE AUTHORS



Maksym D. Myrnneko – PhD student, Computer Science Department. National Aerospace University "Kharkiv Aviation Institute", 17, Vadym Mank Str. Kharkiv, 61000, Ukraine
ORCID: https://orcid.org/0000-0002-5777-9583; maksym.myrnenko@gmail.com
Research field: UAV, computer vision, machine learning; Gen AI

Мирненко Максим Дмитрович – аспірант кафедри Інформаційних технологій проєктування. Національний аерокосмічний університет «Харківський авіаційний інститут», вул. Вадима Манька, 17. Харків, 61000, Україна



Vladimir V. Shevel – Candidate of Engineering Sciences, Professor, Computer Science Department. National Aerospace University "Kharkiv Aviation Institute", 17, Vadym Mank Str. Kharkiv, 61000, Ukraine
ORCID: https://orcid.org/0000-0003-0534-0242; v.shevel@khai.edu. Scopus Author ID: 57211430745
Research field: automated design, automation of full-scale testing of aerospace systems, automation of educational process preparation

Шевель Володимир Вікторович – кандидат технічних наук, професор кафедри Інформаційних технологій проєктування. Національний аерокосмічний університет «Харківський авіаційний інститут», вул. Вадима Манька, 17. Харків, 61000, Україна