

AN ARINC 664 P5- AND ARINC 853-COMPLIANT PLATFORM ENABLING SUPPLIERS TO DEVELOP DIGITAL AIRCRAFT CABIN SOLUTIONS

S. Garcia Aguilar , F. Giertzsch , R. God 

Hamburg University of Technology, Institute of Aircraft Cabin Systems, Hein-Saß-Weg 22,
21129 Hamburg, Germany

Abstract

This paper presents a modular, ARINC 664 P5- and ARINC 853-compliant platform implemented via unified cabin hubs to facilitate the development of interoperable digital aircraft cabin solutions. To overcome the challenges of proprietary interfaces and system heterogeneity, the platform leverages the Cabin Secure Media-Independent Messaging (CSMIM) protocol, employing a data-driven publisher-subscriber architecture to enable secure and scalable communication in non-safety-critical aircraft cabin applications. Through the use of standardized data models and serialization, the platform integrates diverse supplier components, such as lighting, sensors, and galley equipment. Adopting a "Platform for X" approach, these hubs serve as versatile CSMIM network nodes capable of assuming various predefined functional roles via software reconfiguration. To support this agility, the platform leverages the Yocto Project build system to optimize lifecycle management and certification efforts while maintaining strict ARINC 853 compliance. The platform supports deployment within the Airline Information and Services Domain (AISD) and Passenger Information and Entertainment Services Domain (PIESD), utilizing a unidirectional interface from the Aircraft Control Domain (ACD) to enable extended In-Flight Entertainment (IFE) functions as motivated by ARINC 664 P5 Appendix H. Experimental validation involving physical hardware verification and GNS3 network emulation confirmed that Node identity is a configurable software property rather than a hardware constraint. The results demonstrate that secure, controlled cross-domain communication can be achieved via RabbitMQ federation without compromising the domain segregation required by ARINC 664 P5.

1. INTRODUCTION

The aviation industry is continuously evolving towards greater digitalization and connectivity to enhance the passenger experience, improve operational efficiency and integrate systems within aircraft cabins. As shown in FIG 1, modern aircraft cabins comprise a diverse array of systems and equipment, including lighting, sensors, seats, lavatories, trolleys, and galley devices, many of which are sourced from different suppliers with proprietary interfaces and communication protocols. This diversity presents significant challenges when it comes to achieving seamless interoperability, secure data exchange, and the uniform management of cabin systems. These systems are often integrated with each other based on proprietary bidirectionally agreed document-based system interface definitions (SIDs) [1]. In addition, each system may span internal communication networks for intra-system communication.

To address these challenges, aviation standards offer structured frameworks for system architecture and data transfer. Specifically, ARINC 664 P5 [2] defines Network Domain Characteristics and Interconnection for aircraft domain segregation, while ARINC 853 [3] governs Cabin Secure Media-Independent Messaging (CSMIM) for application-level communication. The CSMIM protocol addresses the issue of proprietary data models by introducing a standardized, data-driven, publisher-subscriber architecture. Inspired by the concepts of Integrated Modular Avionics (IMA) [4] and Avionics Full Duplex Switched Ethernet (AFDX) [5], as shown in FIG 2, CSMIM provides a central network for message exchange. While CSMIM focuses on inter-system exchange, it may also be used for intra-system communication. As outlined in this paper, specific network configurations allow facilitating technical communication in compliance with the ARINC 664 P5 aircraft domain model.

To enable standardized interoperability across diverse cabin systems and suppliers, ARINC 853 provides a unified framework for data modeling and messaging formats. This framework was used to establish a centrally organized public CSMIM knowledge base [6] maintained by a working group as part of the ARINC Cabin Systems Subcommittee (CSS). According to [6], it "contains concrete interface definitions for various cabin equipment. These definitions are considered "standard" and should be used by all cabin equipment suppliers as far as possible." For that, the working group orchestrates the addition of new object types and ensures adherence to a common design ruleset.

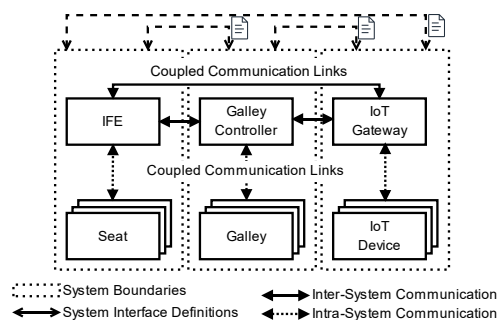


FIG 1. Array of aircraft cabin systems and components.

Each network integrator can instantiate a private repository based on the public knowledge base and include specific devices (instances) to it which provide or require CSMIM objects. This also enables the extraction of standardized SIDs from the repository.

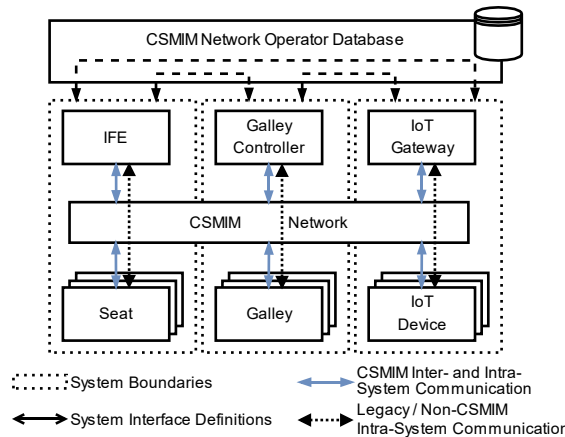


FIG 2. CSMIM-based architecture for standardized cabin communications.

However, a significant implementation gap remains. CSMIM-based networks involve deploying multiple types of devices that perform specialized functions. These include Central Data Service (CDS), Client Nodes, Server Nodes, Gateway Nodes, Proxy Nodes, and Leaf Nodes [3]. Each of these devices has its own unique network and communication requirements [3]. This specialization can result in inflexible, single-purpose configurations and increased maintenance complexity when cabin configurations evolve or new functionalities emerge. To bridge this gap, this paper introduces a modular cabin hub which combines hardware and software capabilities into a unified solution capable of assuming multiple CSMIM functional roles. Using the open-source Yocto Project [7] framework for creating customized, optimized Linux distributions for embedded systems, the platform enables suppliers and integrators to deploy and adapt cabin solutions flexibly through a configurable software build system. This approach ensures full compliance with the ARINC 853 and ARINC 664 P5 standards, providing the agility required for modern aircraft cabin configurations. The following sections detail the underlying standards and protocol architectures, the design and capabilities of the modular cabin hub, and its integration within aircraft domain models.

2. BACKGROUND AND STANDARDS

The integration of diverse cabin services requires strict domain segregation as well as application-layer interoperability. This interoperability enables an efficient deployment of new applications, eliminating the need for custom interfaces for each service. Both requirements must be fulfilled, as modern cabin functionalities inherently rely on the seamless exchange of data among functions that operate at different levels of criticality.

This section describes the aircraft domain segregation model defined by ARINC 664 P5 and the standardized messaging framework defined by ARINC 853. Finally, it identifies the implementation gaps in these standards that motivated the development of the proposed platform.

2.1. ARINC 664 P5 Aircraft Domain Model

Aircraft architectures follow rigorous principles of security domain segregation, where such domains are defined based on the criticality of the functions performed. In [8], the importance of maintaining clear separation between security domains in next generation communication platforms is highlighted. The ARINC 664 P5 standard defines a framework for domain segregation to ensure that safety-critical functions are protected from non-safety-critical ones, while providing guidelines for their secure interconnection. As shown in FIG 3, this model groups functions into four different domains:

- **ACD, Aircraft Control Domain:** The highest criticality domain, responsible for flight-critical functions and essential aircraft systems.
- **AISD, Airline Information and Services Domain:** A domain used for airline-specific applications like flight tracking and operational data.
- **PIESD, Passenger Information and Entertainment Services Domain:** The lowest criticality domain, which covers In-Flight Entertainment (IFE), passenger connectivity, and cabin services.
- **PODD, Passenger-Owned Devices Domain:** Comprises external passenger devices that connect to the aircraft network via the PIESD.

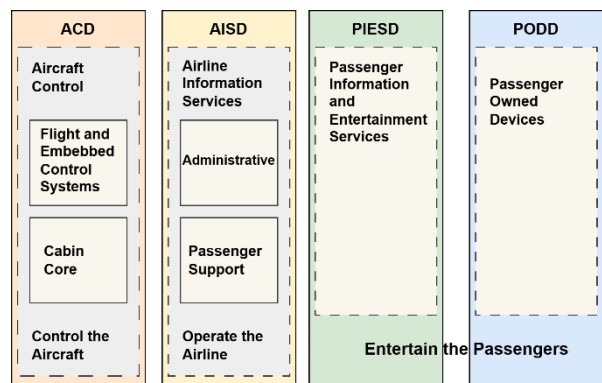


FIG 3. ARINC 664 P5 aircraft domain segregation model, based on [2].

While segregation is essential for safety, modern passenger experiences require a controlled flow of data between these domains. Motivated by ARINC 664 P5 Appendix H and research on the security implications of the aeronautical digital transition [9], there is an increasing demand for cross-domain functionality. This capability enables the crew interfaces in the AISD to interact with the passenger seat functions in the PIESD, while facilitating interactions from the PIESD to the AISD, such as when a passenger presses the IFE PAX call button to notify the cabin crew. To maintain security, all such cross-domain interactions must be rigorously controlled.

In addition, unidirectional interfaces from the ACD are deployed, ensuring that data can flow from high- to low-criticality domains without compromising the integrity of the aircraft control systems. For example, allowing the AISD to receive flight information, such as the remaining flight time, to assist crew members in managing cabin processes.

2.2. ARINC 853 CSMIM Protocol Fundamentals

The ARINC 853 specification defines CSMIM, a standardized, data-driven publisher-subscriber architecture designed to ensure interoperability across diverse aircraft cabin systems.

The CSMIM working group, part of the ARINC Cabin Systems Subcommittee (CSS), maintains the CSMIM Knowledge Base via GitHub [6]. This repository is a collaborative ecosystem where industry members can define and share concrete interface descriptions for various cabin components. The working group provides standardized object types to ensure that all suppliers can implement a unified set of data models, reducing integration complexity. CSMIM manages access to these resources through modular data models based on objects and resources, leveraging MQTTv5 [10] as the messaging transport. This enables lightweight communication, which is ideal for high-density aircraft cabin environments. To optimize performance and bandwidth, the protocol employs CBOR [11] as an efficient binary serialization format. Standardizing message structures and device discovery, CSMIM allows a unified ecosystem in which various devices, including sensors, lighting and galley equipment, can integrate and communicate seamlessly. The CSMIM specification defines specific functional roles (CDS, Client Nodes, Server Nodes, Gateway Nodes and Proxy Nodes) to manage these communications across the aircraft cabin network. FIG 4 shows how diverse cabin components are integrated within a CSMIM-based network architecture.

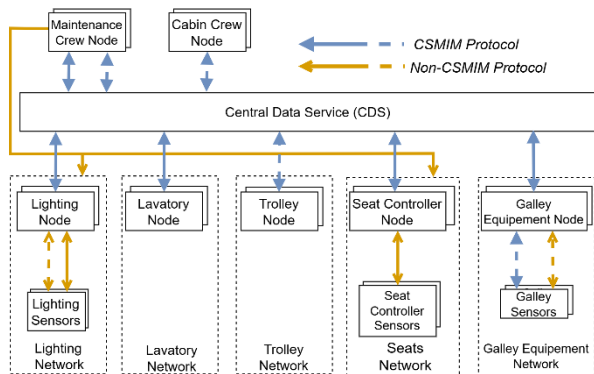


FIG 4. CSMIM network topology with integrated cabin components.

The CDS acts as the main orchestrator. Its primary function is to host the MQTT broker, which facilitates all publish/subscribe traffic. It also requires essential services, such as Node authentication, resource registration and access control configuration. Cabin components can operate as Client Nodes or Server Nodes, while Gateway Nodes and Proxy Nodes can be configured as specialized clients for protocol translation or as federated brokers that connect different network segments. Section 3 provides a detailed structural breakdown of these interactions. To maintain security, the CDS implements Role-Based Access Control (RBAC), restricting publishing and subscribing permissions to authorized Nodes.

Transport Layer Security (TLS) must be implemented to encrypt all data exchanges and ensure the confidentiality and integrity of communications.

2.3. Challenges in Cabin Network Integration

Despite the progress made by ARINC 664 P5 and ARINC 853, there is still a significant gap in how these standards are implemented in aircraft cabins. Current deployment strategies often rely on specialized, single-purpose Nodes that are hard-coded to specific network functions. This specialization creates a cascade of technical and operational challenges.

First, relying on discrete devices as dedicated Gateway Nodes, Proxy Nodes or Client/Server Nodes results in a fixed configuration that lacks flexibility to adapt to new use cases. Furthermore, as cabin configurations evolve with the introduction of new technologies, managing a large fleet of heterogeneous single-function devices increases maintenance costs and complicates long-term lifecycle management. Finally, the proliferation of these unique device types significantly increases the administrative workload associated with certification, as each specific hardware and software configuration requires rigorous, independent testing and validation, complicating the certification process for both airlines and original equipment manufacturers (OEMs).

Therefore, there is an urgent need for a flexible, multi-role, unified cabin hub, enabling a platform capable of transitioning between CSMIM roles. This would enable the platform to adapt to changing and evolving aircraft cabin environments while reducing overall architectural complexity and the associated certification effort.

Additionally, while ARINC 853 provides a framework for interoperability and TLS ensures transport-layer security, neither standard addresses the granular privacy of the data itself. Once a message is decrypted by an authorized subscriber, the raw values are fully exposed. In a multi-supplier environment, this creates a vulnerability where sensitive passenger behavior patterns or proprietary equipment data could be extracted by unauthorized parties who possess valid network credentials [12].

3. SYSTEM REQUIREMENTS AND ARCHITECTURAL CONVERGENCE

Following the identified need for a flexible, multi-role platform capable of hosting diverse cabin functions, it is essential to define the technical specifications that a unified cabin hub must satisfy. The proposed solution adopts a "Platform for X" approach, conceptualizing the hub as a versatile entity where "X" symbolizes its adaptability to various CSMIM network roles.

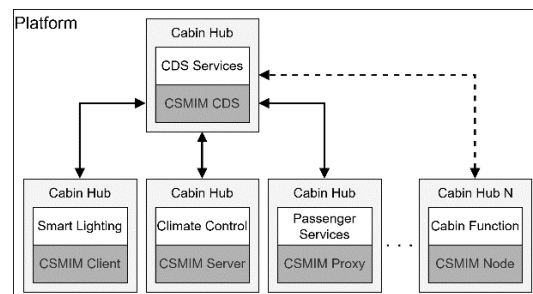


FIG 5. Decoupling and reconfiguration approach for cabin hub design.

As shown in FIG 5, the proposed platform is composed of configurable CSMIM Nodes hosted on cabin hubs, each

executing distinct cabin functions. More specifically, the top layer of each Node hosts the application logic, such as CDS services, smart lighting automation, or climate control, while the underlying layer defines the CSMIM functional role. This design ensures the communication infrastructure remains independent of the specific application functions. It enables the platform to transition between CSMIM network roles through software reconfiguration without requiring modifications to the deployed application logic.

This design decoupling allows the deployment of multiple cabin functions while ensuring that all data exchanges align with standardized data models and security protocols.

TAB 1. Functional roles in a centralized CSMIM architecture

Type
Roles
Description of Functions
CSMIM Central Management
Central Data Service (CDS)
The main network coordinator is responsible for managing devices and the information they share. It enforces role-based access control (RBAC) and records network activity. It acts as an MQTT broker and incorporates various services that run as MQTT clients or broker plug-ins. The CDS also uses broker federation to connect to Gateway and Proxy Nodes, establishing central broker federation rules and configurations that control which communications are allowed on the network.
CSMIM Node
Server Nodes
These devices natively host and manage their own data models and CSMIM resources. Unlike Leaf Nodes, which require an intermediary, Server Nodes can act as standalone, CSMIM-compatible endpoints. They can participate directly in the network by acting as MQTT clients.
Client Nodes
These devices initiate requests to Server Nodes in order to read or write data, execute actions, or discover resources. They consume data and services, but do not own any resources themselves. They also act as MQTT clients.
Gateway Nodes
Intermediaries that bridge Leaf Nodes to the CSMIM network via broker federation. They perform active protocol translation and media conversion (e.g., Bluetooth or Serial to CSMIM) to integrate non-CSMIM-compliant devices into the CDS ecosystem.
Proxy Nodes
Intermediaries that enable connection between different CSMIM networks (segments) via broker federation. They represent connected segments to the CDS as a single Server Node without altering underlying data formats (e.g., CBOR).
Leaf Node
Leaf Nodes
Devices like sensors, with limited power and processing capabilities, do not implement the full CSMIM communication protocols natively and therefore rely on a Gateway Node to translate messages to the CDS.

To achieve this architectural convergence, this section is organized into three parts. Section 3.1 defines the functional network architecture and the hierarchy of responsibilities across different Node types. Section 3.2 details the operationalization of the “Platform for X” deployment concept to eliminate hardware heterogeneity. Section 3.3 addresses the primary challenge of achieving this flexibility while maintaining the strict security boundaries described by ARINC 664 P5 security domains and supporting the ARINC 853 communication model.

3.1. Functional Network Architecture

To ensure scalability and interoperability, the network architecture is organized into three levels of responsibility, CSMIM Central Management, CSMIM Nodes and Leaf Nodes. The communication flow within these distinct layers is illustrated in FIG 6.

As detailed in TAB 1, this architecture assigns specific technical capabilities to each layer. The Central Data Service (CDS), at the highest level, provides global coordination and security enforcement via Role-Based Access Control (RBAC). The intermediate level comprises CSMIM Nodes (Server Nodes, Client Nodes, Gateway Nodes, and Proxy Nodes), which handle active processing and routing of standardized data. At the lowest level are Leaf Nodes, which represent the most constrained devices. These devices lack native CSMIM capabilities and rely on Gateway Nodes for network integration.

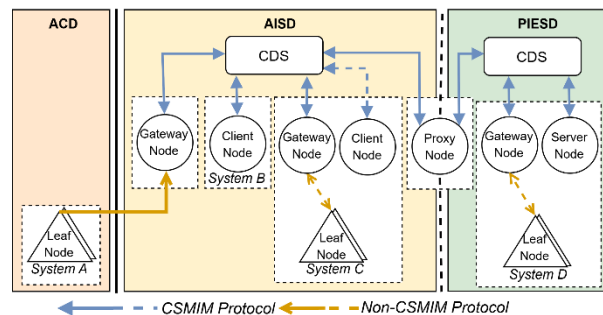


FIG 6. CSMIM protocol network diagram.

3.2. “Platform for X” Deployment Concept

The platform must move beyond the constraints of fixed-function Nodes by implementing the “Platform for X” deployment concept. The expression “for X” was chosen to symbolize the adaptability to various CSMIM roles. The requirement stipulates that a single hardware platform must be able to transition between any of the defined CSMIM functional roles through software reconfiguration, rather than requiring hardware replacement. Implementing the “Platform for X” concept offers advantages to both aircraft integrators and component suppliers. Using a unified, single-source codebase reduces the complexity of software maintenance. By replacing device-specific software versions for each Node type, engineers can use a unified source codebase which can be configured flexibly via role-specific parameters.

Thus, the same underlying software architecture can be used for all CSMIM defined roles by simply altering its configuration profile during the build or deployment phase. Consequently, this approach minimizes hardware heterogeneity across the aircraft cabin and enables a standardized fleet of unified hubs to adapt to evolving

mission requirements [13]. It addresses the challenges identified in Section 2.3 by reducing the logistical challenge of managing specialized devices and simplifying the long-term management of the cabin ecosystem lifecycle.

Furthermore, the platform uses a standardized CSMIM communication library [1] that simplifies the complexities of the ARINC 853 protocol from the application logic. The architecture provides a communication stack within the platform code, allowing suppliers to implement their specific services by assuming a CSMIM Node role simply by interfacing with this established library.

Separating the application logic from the underlying communication protocol ensures consistent implementation of the fundamental data exchange mechanisms across all CSMIM Nodes, reducing the certification effort for new cabin components by leveraging the platform verified communication capabilities.

3.3. Domain Interfaces and Criticality Considerations

The architecture must strictly adhere to the segregation of the highly critical ACD domain. In order to maintain the integrity of the flight-critical systems, any interface connecting to the ACD via the Avionics Data Communication Network (ADCN) must be unidirectional, ensuring that data can be ingested from the ACD while preventing any outbound communication from lower-criticality domains back to the ACD. The platform is also designed for flexible deployment within both the AISD and the PIESD. Allowing the platform to operate in one or both of these domains enables the architecture to provide secure cross-domain functionality. This flexibility is essential for enabling advanced services, such as providing cabin crew with an interface to monitor cabin status (Function A in FIG 7) or controlling business class seat actuators (Function C in FIG 7). Furthermore, "extended IFE" functions, as motivated in ARINC 664 P5 Appendix H, are realized through the cooperation of Nodes spanning different domains. In these instances, as shown for Function B in FIG 7, where communication between the Client/Server (C/S) Nodes is via a Proxy Node that bridges domains.

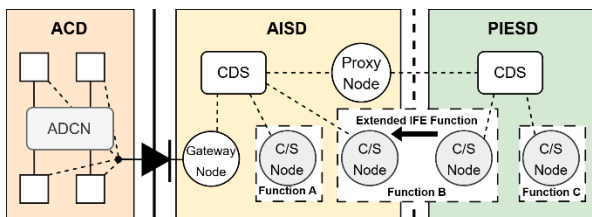


FIG 7. Realization of extended IFE functions according to ARINC 664 P5 Appendix H using CSMIM.

In order to implement cross-domain communication while maintaining the domain segregation mandated by ARINC 664 P5, the platform uses the RabbitMQ [14] broker federation feature, described in [15]. This feature enables the creation of modular, federated, CSMIM-compliant networks, each of which corresponds to a distinct security domain. These networks can then interconnect their message brokers via Advanced Message Queuing Protocol (AMQP) [16] exchanges. Each aircraft domain module runs a dedicated RabbitMQ broker that handles intra-domain MQTTv5 communications with connected

CSMIM Nodes, while the broker federation feature provides controlled, secure and scalable uni- or bi-directional message flows between these brokers.

To prevent unauthorized communication from compromising the domain segregation, a topic-filter whitelist is implemented at the Proxy level. This ensures that only specifically authorized messages can transit from the PIESD to the AISD, while all other traffic is blocked.

4. MODULAR CABIN HUB

The proposed architecture shifts from static software integrated into dedicated hardware to a more flexible, software-defined approach. As detailed in Section 3.2, the "Platform for X" deployment concept decouples configurable CSMIM network roles from their intended functions. This allows a single unit to adapt to various operational requirements via software.

4.1. Platform Software Architecture

As shown in FIG 8, the software architecture employs a modular approach to ensure task segregation, secure communication, and flexible CSMIM deployment. An internal virtual switch and routing engine manage all intra-platform traffic and network configurations at the lowest level of the software stack, making possible the "Platform for X" concept by enabling the platform to reconfigure network interfaces based on the functional CSMIM role without manual hardware intervention.

Building on this foundation, the architecture uses two parallel virtual machines (VMs) that both leverage K3s [17] for application orchestration and container management. This setup follows a similar edge-cloud approach as described in [18]. The first environment, the Communication VM, hosts the communication module, which serves as the backbone of the CSMIM ecosystem. This module includes the RabbitMQ broker with enabled MQTTv5 and broker federation features to facilitate large-scale messaging. Within this VM, specialized communication services are deployed based on the CSMIM Node role. When the platform assumes the CDS role, essential services such as RBAC are deployed.

For instance, when acting as a Gateway Node, the system facilitates integration by intercepting messages from Leaf Nodes that communicate using various non-CSMIM protocols, such as MQTTv3 [25], Bluetooth, and serial interfaces.

The Gateway Node acts as a CSMIM Server for these Nodes, translating native messages into the appropriate CSMIM object topics, structures, and semantic values with the help of an internal client that listens to incoming messages. For example, the Gateway Node can convert data from a Leaf Node communicating via Bluetooth into the standardized CSMIM format before forwarding it to the CDS through broker federation. The CDS then registers the Gateway Node as the authoritative source for those Leaf Nodes. As identified in Section 2.3, although CSMIM provides transport-layer security, it does not protect the content of messages from unauthorized subscribers. To address this issue and prevent an increase in Leaf Node complexity, the Gateway Node introduces predefined noise to Leaf Node raw values before forwarding the CSMIM-formatted message to the CDS. With this approach, the Gateway Node serves as the trusted aggregator for global

differential privacy (GDP), following the design proposed in [12]. Consequently, authorized subscribers only receive an approximation, while maintaining individual privacy.

In parallel, the Functional Services VM hosts the Functional Services module. This environment leverages K3s to orchestrate and execute the intended functions of various aircraft cabin components, such as climate control, lighting automation, and web servers. By decoupling these high-level services from the underlying communication infrastructure, the platform achieves the flexibility required for evolving digital aircraft cabin ecosystems.

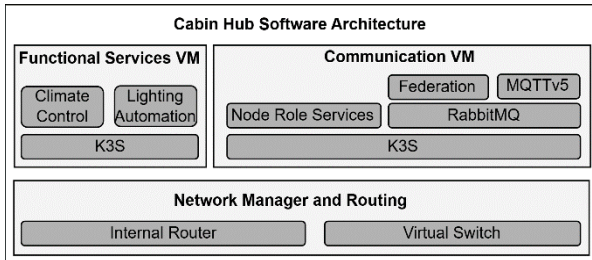


FIG 8. Software architecture for the modular cabin hub.

4.2. Platform Hardware Architecture

The hardware implementation of the platform is based on the LattePanda Sigma Single-Board Computer (SBC) [19], representing modern cabin servers and IoT aggregators by adopting the architectural approach of current industry platforms, such as the Collins Aerospace AWCS-1000 [20], which serve as multi-functional computing platforms. Unlike single-purpose devices, the chosen hardware can host all the previously described CSMIM roles.

To bridge the gap between modern wireless sensors and legacy wired infrastructure, the platform incorporates a versatile communication suite. It leverages dual Ethernet and dual-mode Wi-Fi (client/access point) as the primary physical endpoints, allowing the virtualized routing engine and network management layer to regulate traffic and ensure domain segregation.

By incorporating Bluetooth and high-speed USB ports, the hardware can accommodate specialized peripherals like CAN-to-USB adapters for legacy networks. All of these interfaces together create the physical entry points needed for the Gateway Node's protocol translation logic.

Furthermore, the hardware is optimized to support edge computing capabilities within the aircraft cabin. By extending the software orchestration layer to external environments and supporting connections to satellite communication systems, the platform facilitates reliable data offloading during aircraft turnaround periods. The inclusion of an integrated LTE antenna further facilitates air-to-ground message exchange via RabbitMQ federation, as documented in the *i+sCabin2* report project [21], ensuring that the software-defined CSMIM network remains synchronized with ground-based infrastructures for operational and passenger services.

4.3. Automated Continuous Integration and Continuous Deployment

To manage the deployment of multiple software-defined roles, an automated build and deployment pipeline is implemented. As shown in FIG 9 and FIG 10, the architecture employs a two-part workflow consisting of a Continuous Integration (CI) stage hosted on a dedicated server and a Continuous Deployment (CD) stage managed via a KVM switch interface [22] connected to the LattePanda Sigma. This automation enables remote integration sessions, allowing developers to update or reconfigure hardware targets without requiring direct physical access.

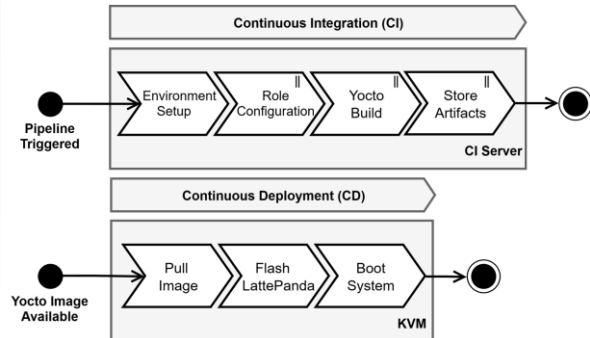


FIG 9. Automated CI/CD pipeline for software-defined role deployment.

The Continuous Integration (CI) workflow begins with an initial environment setup, where a containerized build environment is initialized to ensure that all builds are executed within a controlled and consistent context. Subsequently, during the role configuration stage, an automated Yocto layer manages the unified codebase by injecting role-specific parameters and network configurations into the Yocto Project. This stage serves as the technical realization of the “Platform for X” concept, automating the transition between the functional roles defined in TAB 1 by customizing each software build according to specific CSMIM role requirements.

In parallel (||, FIG 9), the Yocto Build engine generates optimized software images specifically tailored to each assigned CSMIM role, and these resulting bootable images are simultaneously versioned and archived in the artifact storage stage.

The Continuous Deployment (CD) process begins with the specific Yocto artifact being retrieved from the CI server. This image is then automatically flashed onto the target LattePanda Sigma hardware through the KVM interface, after which the platform is initialized to instantiate the new software-defined CSMIM role.

By automating these stages, the platform achieves two primary architectural objectives. First, role-specific determinism is ensured by removing unnecessary components and services at build time, resulting in a lean, function-specific image. Second, automated network configuration is achieved by embedding role-specific interfaces and routes directly into the file system, thereby eliminating the need for manual, error-prone setups during deployment.

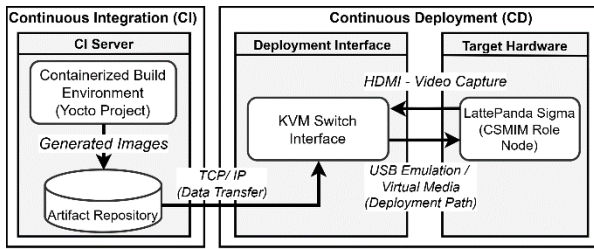


FIG 10. Components and interfaces of the deployment pipeline.

5. EVALUATION AND SIMULATION

The proposed modular cabin hub was validated using a two-stage approach: initial physical verification on LattePanda Sigma hardware, followed by a network emulation using QEMU [23] virtual machines and the GNS3 [24] network emulator. The primary objective of the evaluation was to verify the "Platform for X" concept by ensuring the seamless integration of various CSMIM roles and demonstrating that secure cross-domain communication can be achieved without compromising the domain segregation defined by ARINC 664 P5.

5.1. Functional Agility Assessment (The "Platform for X" Concept)

First, a physical validation was conducted on the LattePanda Sigma hardware to ensure that software images generated via the Yocto Project were compatible with the target hardware, and that all the hardware-specific drivers and interfaces were operating as intended. Based on this physical baseline, four distinct roles (CDS, Server, Gateway and Proxy) were deployed as independent CSMIM Nodes within a GNS3-simulated topology, as shown in FIG 11. The cabin hub VMs were instantiated using QEMU from role-specific images. However, to isolate network validation, these VMs implemented only role communication services and the role network configuration, omitting all active cabin functions.

To demonstrate that role identity is a software configuration property rather than a hardware constraint, multiple instances of the same CSMIM role were deployed using identical shared images: specifically, two Gateway Nodes (Gateway-1, Gateway-2) and two CDS Nodes (CDS-1, CDS-2) were instantiated. The only difference between these instances was their network configurations, confirming the viability of the "Platform for X" approach.

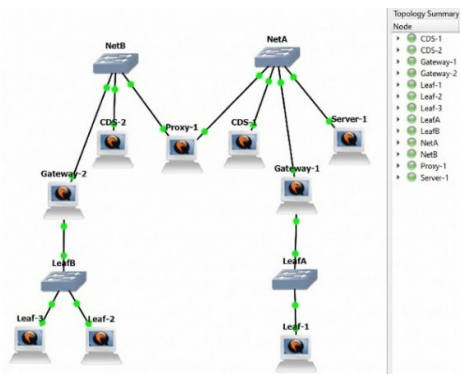


FIG 11. Validation of multi-role CSMIM interoperability via GNS3/QEMU emulation.

5.2. Scalability and Network Load Analysis

In order to evaluate the platform's ability to integrate heterogeneous aircraft cabin equipment, a mediated communication path was validated in the simulation:

Leaf Node → Gateway Node → CDS Node

Validation results demonstrated successful automated onboarding, with the Gateway Node acting as a DHCP server providing dynamic IP addresses to the connected Leaf Nodes. Furthermore, as described in Section 4.1, the Gateway Node successfully intercepted raw MQTTv3 [25] messages from non-standardized Leaf Nodes and transformed them into CSMIM-compliant structures, in line with the "transparent usage" outlined in [20]. The translation was achieved through bi-directional mapping where metadata was extracted from incoming MQTTv3 payloads and embedded into MQTTv5 user properties. Conversely, the required header information was injected back into outgoing message payloads to maintain compatibility with legacy hardware. The messages published by the Leaf Nodes were ingested by the Gateway Node's local RabbitMQ broker and forwarded to the CDS via broker federation. Testing confirmed robust connectivity and 100% packet delivery across the local subnet.

5.3. Cross-Domain Workflow Validation (ARINC 664 P5/853 Integration)

A relevant requirement was to enable interaction between separate aircraft domains (e.g. AISD and PIESD). This was tested by implementing two independent network segments (NetA and NetB) connected via a Proxy Node. Attempts to send either ICMP (ping) or MQTT messages between CDS-2 (NetB) and CDS-1 (NetA) were unsuccessful.

However, messages were successfully exchanged between CDS-2 and CDS-1 via the Proxy Node using RabbitMQ federation, demonstrating that CSMIM messages were shared selectively and securely via a controlled application-layer bridge without exposing the network to unauthorized traffic. The evaluation revealed that, although common loop-prevention methods in message brokers can prevent multi-hop transit, adjusting the 'max-hops' setting in the broker federation configuration enables stable, end-to-end communication over multiple hops.

6. CONCLUSION AND OUTLOOK

This paper has presented a modular, software-defined platform designed to bridge the implementation gap between ARINC 664 P5 domain segregation and ARINC 853 CSMIM interoperability. By introducing the "Platform for X" concept, the proposed architecture shifts away from rigid, single-purpose hardware toward a flexible hub capable of assuming multiple functional roles through software reconfiguration via the Yocto Project.

Experimental validation conducted through both physical hardware implementation and GNS3/QEMU simulations confirms that Node identity is a configurable software property rather than a hardware constraint. Furthermore, RabbitMQ federation demonstrates that secure, controlled cross-domain communication can be achieved following the segregation requirements of the ARINC 664 P5 aircraft domain model. This approach reduces architectural complexity and provides a scalable framework for suppliers

to integrate heterogeneous cabin components into a unified ecosystem. Future work will focus on evaluating platform performance under high-traffic scenarios to quantify the latency impacts of broker federation and integrating Model-Based Systems Engineering (MBSE) into the deployment pipeline. Building upon previous research [1], the goal is to automate the generation of the software build process and CSMIM communication configurations directly from system models. This would enable a seamless transition from model design to an ARINC-compliant hardware deployment, further reducing manual configuration errors and accelerating the certification process for digital cabin solutions.

ACKNOWLEDGEMENTS

This work was supported by the LuFo VI-2 project “Innovative offene Anwendungs-Edge Cloud-Plattform zur Integration von Kabine und Flughafen und Sensor-Technologien” (engl.: Integrated Airport and Cabin, Digital & Sensory Technologies, INTACT, project number 20D1928C) funded by the Federal Ministry for Economic Affairs and Climate Action based on the decision of the German Bundestag.

Contact address:

sergio.garcia.aguilar@tuhh.de

REFERENCES

- [1] F. Giertzsich, M. Blecken, R. God. Towards a Model-Based Systems Engineering Framework for the Design and Configuration of Communication Networks in a Data-Driven and Interconnected Aircraft Cabin. 27th IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS), 2024, pp. 216–226. <https://doi.org/10.1145/3652620.3687425>
- [2] SAE International. ARINC 664 P5 Standard (Aircraft Domain Model). <https://www.sae.org/standards/content/arinc664p5/>
- [3] SAE International. ARINC 853 Standard (CSMIM Protocol). <https://saemobilus.sae.org/standards/arinc853-853-cabin-secure-media-independent-messaging-csmim-protocol>
- [4] EASA. Integrated Modular Avionics (IMA) Concepts/Standards. <https://www.easa.europa.eu/en/downloads/48264/en> (accessed 2026/08/05).
- [5] SAE International. Avionics Full Duplex Switched Ethernet (AFDX) Standards. <https://www.sae.org/standards/content/arinc664p7-1/>
- [6] SAE ITC. CSMIM Knowledge Base. GitHub Repository. <https://github.com/ARINC-IA/CSMIM> (accessed 2026/08/05).
- [7] The Yocto Project. The Yocto Project (Open-source build system). <https://www.yoctoproject.org/> (accessed 2026/08/05).
- [8] S. Poulain, W. Fischer, F. Giertzsich. Next Aircraft Hyperconnected System Platform. 2025 AIAA DATC/IEEE 44th Digital Avionics Systems Conference (DASC), Montreal, QC, Canada, 2025. <https://doi.org/10.1109/DASC66011.2025.11257427>
- [9] M. S. Ben Mahmoud, A. Pirovano, N. Larrieu. Aeronautical communication transition from analog to digital data: A network security survey. *Computers & Security Review*, vols. 11-12, pp. 1–29, 2014. <https://doi.org/10.1016/j.cosrev.2014.02.001>
- [10] OASIS. MQTTv5 Protocol Specification. <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html> (accessed 2026/08/05).
- [11] IETF. CBOR (Concise Binary Object Representation) Standard. <https://www.rfc-editor.org/rfc/rfc8949.html> (accessed 2026/08/05).
- [12] N. Vyas, B. Zhao, A. Baltaci, G. de C. Bertoli, H. Asghar, M. Klügel, G. Schramm, M. Kubisch, D. Kaafar. Privacy-Preserving IoT in Connected Aircraft. Nov. 19, 2025. <https://arxiv.org/pdf/2511.15278>
- [13] V. Starke, F. Thielecke. Network Simulation for future distributed Aircraft Cabin Platforms. 2025 AIAA DATC/IEEE 44th Digital Avionics Systems DASC <https://doi.org/10.1109/DASC66011.2025.11257282>
- [14] RabbitMQ. RabbitMQ Messaging Broker Documentation/Software. <https://www.rabbitmq.com/> (accessed 2026/08/05).
- [15] F. Giertzsich, M. Blecken, R. God. Design Considerations for Broker Federation Architectures for Data-Driven Business Processes in Aviation. DLRK, 2024. <https://doi.org/10.25967/630278>
- [16] OASIS. AMQP (Advanced Message Queuing Protocol) Specification. <https://docs.oasis-open.org/amqp/core/v1.0/cs01/amqp-core-complete-v1.0-cs01.pdf> (accessed 2026/08/05).
- [17] K3s. K3s (Lightweight Kubernetes Distribution). <https://docs.k3s.io/> (accessed 2026/08/05).
- [18] S. Garcia Aguilar, F. Zabihzadeh, F. Giertzsich, R. God. Applications and Architecture for UWB Localization in Aircraft Cabins and Aviation. DLRK, 2025. <https://doi.org/10.15480/882.16038>
- [19] LattePanda Sigma <https://www.lattepanda.com/latte-panda-sigma> (accessed 2026/08/19).
- [20] Collins Aerospace, "AWCS-1000 System Interface Unit Product Brochure," [Online]. Available: <https://www.rtx.com/collinsaerospace/-/media/CA/product-assets/files/public/products/product-brochures/cabin/ife/awcs-1000.pdf?rev=39075c03806d4c729f30023912fbc4ca> (accessed 2026/08/19).
- [21] R. God, M. Blecken, F. Giertzsich. Model-based Systems Engineering for a cyber-secure, certifiable, intelligent, and smart cabin (i_sCabin2). Hannover: Technische Informationsbibliothek, 2026. <https://doi.org/10.34657/38819>
- [22] KVM. KVM Switch (“keyboard, video and mouse”) <https://www.lenovo.com/us/en/glossary/kvm-switch/> (accessed 2026/08/05).
- [23] QEMU. QEMU Virtual Machine Emulator. <https://www.qemu.org/> (accessed 2026/08/05).
- [24] GNS3. GNS3 Network Emulator. <https://docs.gns3.com/> (accessed 2026/08/05).
- [25] OASIS. MQTTv3 Protocol Specification. <https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html> (accessed 2026/08/05).