DESIGN CONSIDERATIONS FOR BROKER FEDERATION ARCHITECTURES FOR DATA-DRIVEN BUSINESS PROCESSES IN AVIATION

F. Giertzsch, M. Blecken, R. God
Institute of Aircraft Cabin Systems, Hamburg University of Technology, Hein-Saß-Weg 22, 21129 Hamburg, Germany

Abstract

The competition of airlines and pressure on cost requires them to continuously improve their business processes. For this, a number of innovation projects aim to create data-driven services. To provide the information required to enable a data-driven service in aviation, communication networks interconnect systems installed in the aircraft as well as ground-based systems. In this paper, a novel modularized communication architecture, i.e., interconnecting multiple airborne systems as well as ground-based systems, considering known and relevant aviation standards is proposed. To this end, the concept of message broker federation is studied and results are used to deduce a generic, modularized and federated communication architecture. The design considerations for this communication architecture also comprise the security domain model in accordance with the ARINC 664 P5 standard. Thus, means are proposed to secure the broker-based communication on an architectural level and to ensure airworthiness security. The viability of the proposed generic communication architecture is discussed by creating two specific instances of the generic architecture tailored to realize different example use cases. Both resulting federated broker architectures, one for a physical distributed demonstrator as part of a research project and the other for a virtual extensible testbed, have been implemented and initial validation confirmed its success.

Keywords

messaging architectures, CSMIM, data-driven aviation, cyber-physical system of systems networks, security domains

1. INTRODUCTION

The ever-growing competition of airlines and pressure on cost requires them to invest in improving effectiveness and efficiency of the business processes inside the aircraft cabin for more economical operations and for generating ancillary revenue. In order to enhance such business processes in the age of digital transformation, today's cabin innovation projects follow the approach of designing novel data-driven services. For a data-driven service, information from various data sources is gathered and processed together in order to realize its functions. Examples range from introducing Prognostics and Health Management (PHM) for cabin maintenance to optimized and personalized in-flight catering. The goal of PHM is to increase the availability of cabin equipment, e.g., actuators of business class seat kinematics or functionality of galley inserts like coffee makers, thus, increasing passenger comfort and reducing cabin crew workload. Increased passenger comfort is also one objective of an optimized and personalized catering process, e.g., by providing a service to the passenger that allows meal ordering directly at the seat based on current availability and passenger preferences. The same information can also be used to infer an optimal loading of aircraft galleys [1], i.e., which type of food, beverages and duty-free products are loaded on a specific flight. Thus, communication is not limited to intra-aircraft information exchange, but can range across the entire air transport system forming a system of systems [16].

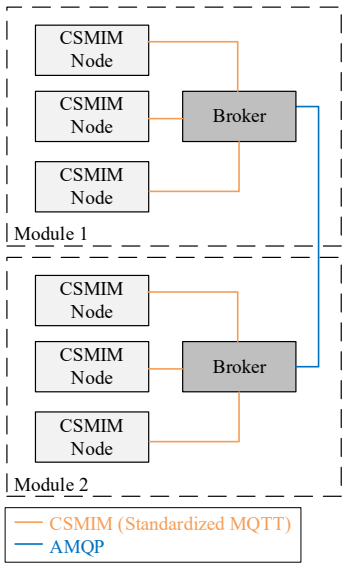

FIG 1. CSMIM-compliant architecture supporting broker managed on module level based on federation concepts.

In scope of the ARINC 853 Cabin Secure Media Independent Messaging (CSMIM) standardization project

doi: 10.25967/630278

[2], research groups [3] and standardization bodies recently worked on common aircraft data models and communication protocols enabling the exchange of information required for the described passenger, crew and airline services inside the aircraft cabin. Although introducing an abstract messaging layer, currently, the CSMIM specification supports only MQTT [4] as a means for transport. That is, networking nodes that are CSMIM-compliant (CSMIM nodes) use MQTT to communicate with a broker. As the standard specifies interfaces, various design variations on both, CSMIM node and broker side, are possible.

In this paper, a class of CSMIM-compliant broker designs that make use of the federation concept is studied. More specifically, an architecture as shown in FIG 1 is considered. This architecture allows to create modules as a grouping of CSMIM nodes into a CSMIM-compliant network. The modules can be connected using the Advanced Message Queuing Protocol (AMQP) [5, 6] while maintaining the CSMIM-compliant MQTT-based communication with CSMIM nodes. This enables shortening product development cycles as integration activities can already start on module level and existing modules can be-reused. Furthermore, aviation regulations require to identify, asses and mitigate security risks, cf. CS 25.1319 in [22]. Broker federation can technically support such mitigation of possibly identified security risks. By separating intra-module communication from inter-module communication, the number of inter-module connections can be reduced such that the attack surface is minimized. In addition, AMQP-features can be configured for the specifics of inter-module connections (blue connector in FIG 1). These can be technical, e.g., as motivated above, a lossy or low-bandwidth air-to-ground connection or contractional, e.g., only certain and agreed information should be exchanged between manufactures, third parties and their systems, respectively.

An overview of current trends and the aviation regulatory framework with respect to messaging technologies, such as CSMIM, AMQP and MQTT, is further explained in Section 2. In Section 3 concrete network design considerations when employing broker federation principles within the aircraft and across the air transport system are proposed. A research project demonstrator has been built based on these design considerations and a virtual testbed has been implemented for further studies. Corresponding results are presented in Section 4. Finally, Section 5 concludes the paper and future research directions are suggested.

2. MESSAGING IN DATA-DRIVEN AVIATION

The information required for data-driven services as introduced in Section 1 is geographically distributed. For example, a catering loading list is generated at a caterer filling the trolleys, whereas, information on the drinks and meals requested by the passengers is known to the cabin crew in the aircraft cabin. In addition, information can be available in different formats or encodings as data. This includes unstructured data such as a loading list written in natural language [7]. Within the aircraft cabin, there also exists no standardized transformation of information into data, but interface control documents are often mutually agreed between engineers. As an approach to solve this, research groups and standardization bodies have worked towards a concept for standardization both,

- a data model including an encoding specifying a transformation from information into data, as well as,
- the communication protocol that allows to exchange information described by the data model [2, 3].

The resulting CSMIM specification introduces a central data service (CDS) that is composed of an MQTT version 5 (MQTTv5) message broker and additional central services, managing, for example, discovery of information or authentication and authorization. A CSMIM node is communicating with the CDS using CSMIM operations in order to, for example, provide and receive information according to the CSMIM data model. More specifically, the CSMIM data model is based on the concept of CSMIM objects. CSMIM objects are itself composed of CSMIM resources that can be read, written or executed, cf. [2] for details of the structure of CSMIM objects and resources as well as the semantics of the corresponding CSMIM operations. A CSMIM node that exposes one or multiple CSMIM objects with its resources to the CSMIM network acts as a CSMIM sever. A CSMIM client uses the aforementioned CSMIM operations to access the CSMIM object. The CSMIM network is specified to be information-centric, that is, CSMIM clients do not specifically address the CSMIM server that exposes the CSMIM object, but the CSMIM object itself.

Although introducing a client-server model, different application layer protocols may implement the CSMIM operations. The current version of the specification, however, only defines a mapping onto MQTTv5. That is, for each CSMIM operation, rules are defined as to which MQTT control packets shall be used and how each packet field shall be filled. This includes, for example, a definition of the MQTT topic to be used.

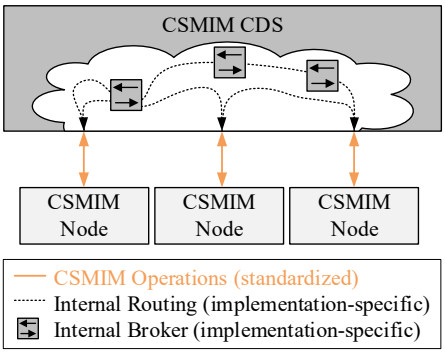

FIG 2. CSMIM communication architecture.

As depicted in FIG 2, the internal design of the CDS, i.e., how it is implemented, is, however, not enforced by the specification, but is limited to the interaction between the CDS and CSMIM nodes via CSMIM operations. The idea of this paper is to apply the concept of broker federation to the MQTT broker as part of the CDS. By this, no central message broker, but a set of multiple internal cooperating message brokers route messages between CSMIM nodes. As motivated above, this design strategy can support the creation of modules, each comprising a standalone

CSMIM-compliant network. This enables

- segregated communication between a set of CSMIM nodes due to regulation-driven security requirements or business-driven information protection requirements as further outlined in Section 2.1 as well as
- natively extending the messaging architecture to include multiple aircraft and ground infrastructure and services as described in Section 2.2.

The approaches proposed in this paper specifically address the realization of business processes. Business processes are processes implemented by airlines, in addition to operational processes, for a more economical operation, increased passenger service or comfort as well as for generating ancillary revenue. Operational processes, in contrast, refer to processes that are required by regulations on air operations [19], e.g., a passenger safety briefing as per EASA CAT.OP.MPA.170.

Furthermore, for this paper it is assumed that the execution or non-execution of aforementioned business processes, shall not have any effects on the safety and airworthiness of the aircraft [18, 19]. Thus, examples for business processes include processes related to catering, e.g., meal preparation and distribution as well as corresponding reporting activities.

2.1. Local System Federation Architectures within the Aircraft

The messaging standard CSMIM is specifically designed for information exchange within the aircraft cabin. Within this geographic scope, i.e., locally within the aircraft, contract-based information exchange [3] as well as ARINC 664 Part 5 [8] standard compliance could benefit from broker federation. In this context, contract-based information exchange refers to an agreed information exchange with defined communication partners. As detailed below, ARINC 664 Part 5 introduces the so-called aircraft domain model for safety and security critical functions. As this paper focuses on business processes, failure conditions (safety) and threat conditions (security) of corresponding technical functions must not have any effect on the safety and airworthiness of the aircraft. However, failure conditions and threat conditions may still have effects on the reputation and revenue of airlines, thus, impacting the airlines business. In addition, information from functions with higher criticality may be required to implement certain business processes [17], e.g., aircraft-related context-information such as the phase of flight or departure and destination airport. Thus, the proposed concept needs to include a security architecture considering aviation security guidelines and principles e.g., through the ARINC 664 Part 5 standard, so that it can actually be used as a technical baseline for implementing novel business processes within the aircraft cabin.

2.1.1. Contract-Based Information Exchange

A CDS, as per CSMIM specification, employs authentication and authorization services that provide functions than could be used to implement contract-based information exchanges. A CSMIM node can control through tagging who gets access to the CSMIM objects exposed to the network. For this, access is defined through a combination of a role-based access control model and certificates. The CDS is responsible to enforce the resulting access control rules. In case of a system design composed of equipment distributed through the aircraft cabin, CSMIM could be used for intra-system communication fostering company-wide harmonization of interfaces with similar characteristics. Using specific tagging, intra-system messages can be privately exchanged through a shared CDS. In order to minimize potential intellectual-property discussions, a system broker can be added to the system. System broker and CDS broker are intertwined into a broker federation in which the system broker handles intra-system communication and only previously contractually agreed inter-system messages are forwarded to the CDS.

2.1.2. ARINC 664 Part 5 Standard Compliance

For developing the security architecture of networked aircraft systems, ARINC 664 Part 5 proposes an aircraft domain model [8]. This security-by-design approach assigns functions of systems to a domain, allows controlling interaction as well as expectations between domains and eases the risk management required by ED 203A [9]. For this, ARINC 664 Part 5 defines four security domains, namely:

- Aircraft Control Domain (ACD),
- Airline Information Services Domain (AISD),
- Passenger Information & Entertainment Services Domain (PIESD),
- and the Passenger Owned Device Domain (PODD).

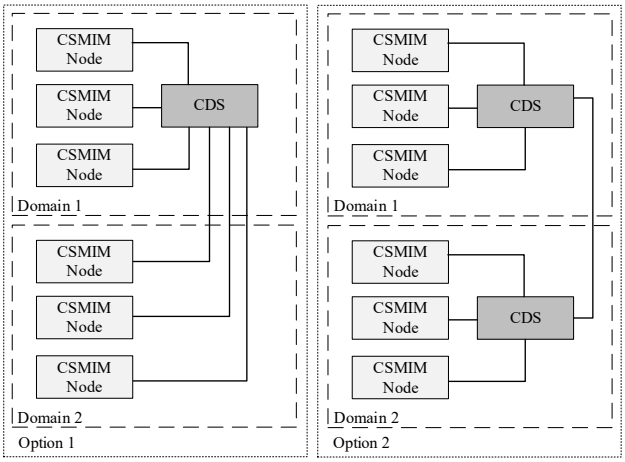

FIG 3. Multi-domain CSMIM network. Option 1, left: Multi-domain CDS. Option 2, right: Federated CDS.

The ACD requires strict protections as functions of this domain handle flight-relevant control information as well as functions required for the safe execution of cabin operational processes. The AISD contains service functions for both flight crew and cabin crew. In the context of this paper, these can, for example, be a set of functions allowing the cabin crew to identify passengers that ordered special meals. The PIESD together with the PODD form the counterpart for the passenger services. These domains contain functions that are accessed by passenger, e.g., in-

flight entertainment, ordering food or drinks as well as remote controls for the seat.

In [17] an approach for the of design of data-driven aircraft cabin networks, such as a CSMIM network, considering the security guidelines and principles described by ARINC 664 Part 5 as well as ED-203A has been developed. This includes that during architecture development it is suggested to group functions such that the number of interfaces are minimized as this might reduce the overall attack surface.

In a CSMIM network, CSMIM nodes, more specifically their assigned functions, could be assigned to different domains. If a single CDS is deployed in this network, as, for example, shown in FIG 3 (left), it must be designed to separate CSMIM nodes assigned to multiple different domains. That is, each connection from a CSMIM node in *Domain 2* opens yet another inter-domain communication link that must be secured. By introducing one broker for each security domain, as, for example, shown in FIG 3 (right), the number of inter-domain connections are reduced simplifying risk analysis and possibly implementation. In this architecture, only broker have to implement a domain gateway, whereas CSMIM nodes interfaces are limited to intra-domain connections.

2.2. Global System of Systems Federation Architectures within Air Transportation

Messaging in aviation is not limited to networked aircraft systems, that are designed and maintained according to aviation safety and security regulations. Data-driven services interact with servers on ground, e.g., functions provided by the airline operation center or the airport [10]. The scope cannot only be extended geographically, but also along the product lifecycle, e.g., for order management in the aviation supply chain [11].

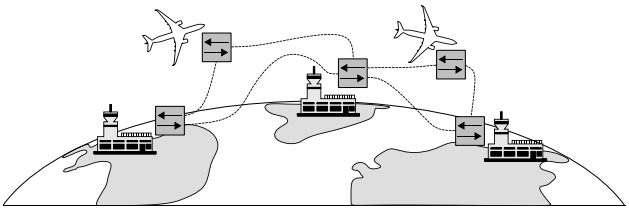

FIG 4. Example global federation architecture based on [10].

As shown in FIG 4, aircraft cabin networks that comprise a message broker, such as a CDS in a CSMIM network, can be connected with other aircraft and expanded into the ground segment by broker federation forming a System of Systems [10]. As detailed below in Section 2.3, message broker implementations based on the Advanced Message Queuing (AMQ) model [5, 10], introduce the notion of message queues. This concept can be used to buffer messages in the aircraft and on ground until successfully transmitted to ground or to aircraft, respectively. By this, AMQ-based federation can handle lossy networks or networks that are temporarily disconnected by design. Thus, there is no need to specifically handle these links by custom built software components.

2.3. AMQP as a Backbone Messaging Protocol

The AMQ model as part of the Advanced Message Queueing Protocol (AQMP), version 0-9-1, specification [5] defines the messages broker semantics. Due to its wide range of supported messaging pattern, stated in Section 1.2.6 in [5], messages transported through other protocols, e.g., MQTTv5 [4], can be tunneled through AMQP-based broker [12]. This opens the possibility to realize a CSMIM CDS based on an AMQP message broker. One well-known AMQP broker with MQTTv5 support is RabbitMQ that is written in Erlang [10, 12-15].

The main entities of the AMQ model, as depicted in FIG 5, are queues and exchanges. Messages are generated by producers and then sent towards the broker. A message can either be routed directly into a queue or further distributed by different types of exchanges. Messages that arrive at an exchange are routed to queues based on the specific semantics of that exchange type. The routing can, for example, depend on bindings that refer to certain message properties [5]. If the message matches a binding, it is forwarded to bound queue. In the context of CSMIM that, currently, employs an MQTT-based application layer protocol, AMQ topic exchanges are specifically important. A queue is bound to an AMQ topic exchange by defining a topic pattern. A producer sending a message to a topic exchange appends a topic to the message. If a topic matches a topic pattern defined in a binding, it is forwarded to the corresponding queue. Messages are forwarded to multiple queues in case the topic matches multiple topic pattern. By using topic exchanges, the behavior of an MQTT broker can be re-constructed. as AMQ topics fulfil at least those properties and features of MQTT topics. Finally, messages that are buffered in queues can then be forwarded to consumers. In case of topic exchanges, as indicated above, producers and consumers may either sent and receive messages through AMQP or MQTT.

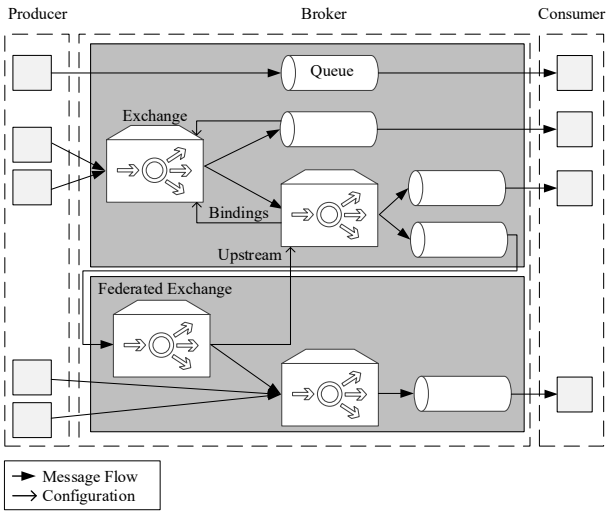

FIG 5. Entities of the AMQ model including federation principles.

Broker federation is not explicitly defined in the AMQ model or protocol, but provided as a broker-specific feature. Specifically, RabbitMQ, an open source message broker software, uses AMQP to implement the communication between federated broker and allows to federate exchanges as well as queues.

For this paper, the RabbitMQ feature of exchange federation, as also depicted in FIG 5, is used. On broker level, RabbitMQ introduces the notion of a downstream (broker) and an upstream (broker). The wording aligns to the dependency of these broker instances similar to the software engineering discipline, see, e.g., [23]. A downstream (broker) specifies and connects to an upstream (broker) to create a directed data flow from the upstream to the downstream. Then, an exchange can be configured on the downstream to be federated onto the upstream. As a result, any message published on the upstream exchange that has a matching binding on the downstream exchange, is forwarded from the upstream exchange to the downstream exchange and then delivered to any queue that has a matching binding. In order to manage temporary connection losses or lost messages between upstream and downstream, a queue on the upstream-side buffers messages until the connection to the downstream is re-established. [6]

3. DESIGN CONSIDERATIONS FOR BROKER FEDERATION ARCHITECTURES

Airline data-driven business processes can evolve over time. Depending on the current market demands, processes might be altered, suspended or new processes are introduced. As a result, information needs of the corresponding technical functions also change and need to be implemented by a modified or even newly introduced messaging service. Therefore, in this paper, design considerations for flexible and configurable messaging architectures are presented. As deduced in the previous Section 2, the resulting architecture should support a harmonized information exchange within the aircraft cabin and information available in ground-based systems should be accessible by systems installed in the aircraft cabin and vice-versa.

In order to fulfill this need, the generic RabbitMQ-based architecture shown in FIG 6 is proposed as a baseline. The depicted architecture describes how a single module is constructed and any two modules are interconnected through RabbitMQ federation. A module may be instantiated in the aircraft or as part of the ground-based infrastructure.

The aircraft cabin is composed of multiple modules, each being a CSMIM-compliant network on its own, i.e., CSMIM nodes can provide and consume information through a CDS. A module refers to any grouping of aircraft equipment communicating through a shared CSMIM network. Depending on how the corresponding systems are planned to be certified, a module can represent a system with its corresponding equipment, a sub-system or even equipment belonging to different systems. The binding between the *Topic Exchange* and the *To-Module-i* allows to specify which information is shared with *Module i*. Thus, as motivated above, this concept allows the CDS of a module to be configured such that information not needed outside the module actually stays within the module and is never shared with other modules.

To allow CSMIM nodes to connect to a RabbitMQ-based CDS, a plugin is available that provides an MQTTv5 interface to clients [15]. Messages are then received as MQTTv5 messages and made available through an AQMP topic exchange. If a message passes through this exchange that contain information subscribed by a CSMIM node, it is emitted as an MQTTv5 message, accordingly.

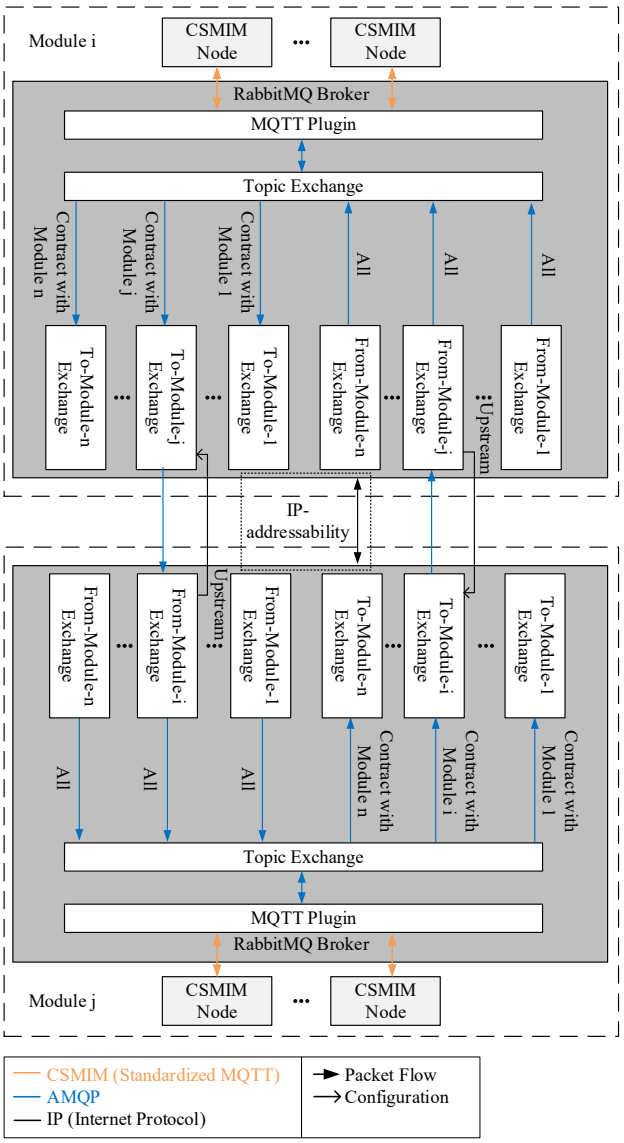

FIG 6. Generic RabbitMQ-based federated CSMIM CDS architecture.

The concept of federated exchanges allows to create a uni-directional message flow from an upstream exchange to a downstream exchange. A bi-directional message flow between federated exchanges on two different brokers, as depicted in FIG 6, can be established by mutually configuring the other broker as an upstream.

In order to define another broker as an upstream, an IP address needs to be specified. Thus, in case a bi-directional link is required, the corresponding brokers need to be mutually *IP-addressable*. Here, the property *IP-addressable* refers to a pair of two endpoints. An endpoint is *IP-addressable* if its IP address is routable from the other endpoint, i.e., the endpoint is directly reachable through its IP address from the other endpoint. Especially in case of a link between an aircraft system and a ground system, this may not be inherently given as the corresponding aircraft equipment can be connected through, for example, satellite or cellular link performing Network Address Translation

(NAT) [20]. This can be solved by, for example, introducing a Virtual Private Network (VPN) [12] requiring only one IP-addressable endpoint to be initially contacted for the VPN setup, either in one of the networks to be connected or a negotiator.

The creation of bi-directional communication links carries the risk of routing loops. In a setup as shown in FIG 6, for example, a CSMIM node in *Module i* and *Module j* may subscribe to the same topic *t*. If a CSMIM node in *Module i* publishes a message that matches this topic *t*, it will be routed to subscribers in *Module i.* Through the *To-Module-j Exchange* and *From-Module-i Exchange* it will also be forwarded to the *Topic Exchange* in *Module j*. If messages are not tracked, the *Topic Exchange* in *Module j* may itself route the message not only to subscribers in *Module j*, but also back again to *Module i* as CSMIM nodes of this module subscribed to a matching topic. RabbitMQ, however, implements different means to prevent such routing of messages [21].

4. EXAMPLE SYSTEM AND SYSTEM OF SYSTEMS FEDERATION ARCHITECTURE

The proposed generic architecture for implementing airline business processes with federated message brokers is exemplarily applied to realize

- a concrete use case as part of a federated research project demonstrator as well as
- an extensible virtual testbed that can be used to do a functional validation of the proposed federation architecture and, for future studies, to investigate specific features or configurations of the generic federation architecture.

4.1. Physically Distributed and Federated Project Demonstrator

A research project demonstrator required a specific configuration of the proposed generic federation architecture described in the previous Section 3. In scope of the research project, the producer and consumer of information do not conform to the CSMIM specification. By this, the *MQTT Plugins* as well as a corresponding *Topic Exchanges*, c.f. FIG 6, can be omitted, resulting in the architecture shown in FIG 7. Messages shall be exchanged between aircraft modules as well as between each aircraft module and a ground station. The modules of the generic architecture can, therefore, be mapped to this architecture as follows:

Module 1	↦	*Aircraft Module 1*
…		
Module n	↦	*Aircraft Module n*
Module n+1	↦	*Aircraft Concentrator*
Module n+2	↦	*Ground Station*

The exchanges are mapped as follows[1]:

Aircraft Module i:		
From-Module-n+1 Exchange	↦	*To-Aircraft-Exchange*
To-Module-n+1 Exchange	↦	*To-Ground-Exchange*
Aircraft Concentrator:		
To-Module-1 Exchange … *To-Module-n Exchange*	↦	*To-Aircraft-Exchange*
From-Module-1 Exchange … *From-Module-n Exchange*	↦	*To-Ground-Exchange*
Ground Station:		
To-Module-n+1 Exchange	↦	*To-Aircraft-Exchange*
From-Module-n+1 Exchange	↦	*To-Ground-Exchange*

In case of the aircraft concentrator, *To-Module-i* and *From-Module-i* exchanges are pooled as no specific handling of messages to and from aircraft modules is required, e.g., dropping certain topics due to contracts limiting information exchange. In addition, in order to ease configuration management of aircraft modules, one central aircraft concentrator module is introduced, instead of relying on direct module-to-module communication. That is, each aircraft module needs to configure just one upstream broker, namely the aircraft concentrator. The aircraft concentrator itself needs to configure one upstream for each aircraft module as well as one upstream for receiving messages from the ground station destined for aircraft modules. Correspondingly, for each aircraft, the ground station needs to also configure one upstream for receiving messages from the aircraft.

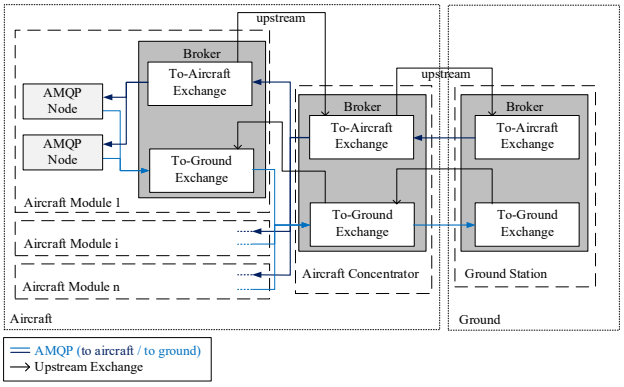

FIG 7. Implemented physically-distributed broker federation architecture with ground services[2].

The architecture shown in FIG 7 has successfully been physically realized as part of the research project demonstration. Two aircraft modules are implemented on a Linux-based embedded system specifically designed as an aviation-certifiable electronic hardware. The modules are connected via Ethernet to a certified aircraft cabin server running the aircraft concentrator message broker. Through the cellular interface of the cabin server, a VPN tunnel establishes the connection with a commercial-grade, internet-facing Linux-based server representing the ground station. In addition to the message broker, the ground station server executes applications that interact with applications on the aircraft modules through the federated message broker.

[1] Exchanges that are not mapped are omitted in the table.

[2] For readability, AMQP nodes as part of the ground station are omitted. As for Aircraft Module 1, such AMQP nodes communicate with the exchanges of the ground broker.

This setup also shows that the range of applicability of the underlying proposed design considerations is not limited to CSMIM CDS federation, but can be re-used for a broader scope of airline business processes.

4.2. Extensible Virtual Testbed

The research project required a specific configuration of the generic architecture depicted in FIG 6, omitting CSMIM-compliance and dedicated handling of inter-module message exchange. In order to comprehensively assess the suitability of the proposed approach, an extensible virtual testbed as shown in FIG 8 has been implemented. Technically, this testbed is based on Linux container and virtual networking technology. Compared to the use case oriented physical demonstrator, this allows to add and remove individually IP-addressable modules without the need to add hardware, e.g., additional embedded systems or servers.

Similar to the project demonstrator, multiple aircraft modules are interconnected through a central aircraft concentrator. In order to configure (blue ellipse *CFG* in FIG 8) inter-module messages, *MQTT Exchange* and *From-Module Exchange* are separated. As part of *CFG*, a topic filter binding can be specified that defines which messages are routed to the federated *From-Module Exchange*. Messages are then forwarded through the *Cabin Exchange* of the *Aircraft Concentrator* to any other aircraft module in case a matching subscription of a CSMIM node exists.

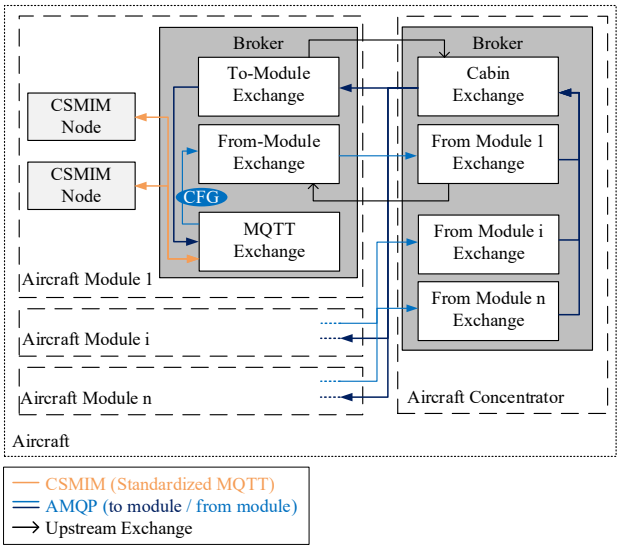

FIG 8. Implemented virtual testbed for studying federated CSMIM CDS.

For initial functional validation, the architecture shown in FIG 8 with two aircraft modules, i.e., $n = 2$, *Aircraft Module 1* and *Aircraft Module 2*, and an aircraft concentrator has been instantiated in the virtual testbed. In each aircraft module one CSMIM node, each with a CSMIM server and a CSMIM client, is connected to its module CDS. Both CSMIM servers expose a CSMIM object with one integer-valued readable resource that is incremented once a second. The CSMIM client of both CSMIM nodes subscribes to a topic filter covering the topics of both readable resources.

Upon startup of both CSMIM nodes, as expected, the corresponding CSMIM clients receive an update of each resource once a second. For each resource, each counter value is received once, showing that RabbitMQ has successfully been configured

- to receive inter-module messages as well as intra-module messages and,
- to prevent routing loops that would result in messages being received multiple times.

Based on the aforementioned example subscriptions, messages are forwarded through the exchanges in the originating module (*MQTT Exchange* and *From-Module Exchange*), the aircraft concentrator (*From-Module-1 Exchange* or *From-Module-2 Exchange*, respectively, depending on the message origin and *Cabin Exchange)* and the receiving module (*To-Module Exchange* and *MQTT Exchange)*. However, as the CSMIM client in the originating module also subscribed to the corresponding topic, if not prevented by RabbitMQ, the message could have been received twice

- directly through the *MQTT Exchange* of its corresponding module CDS and,
- through the loop generated by the *Cabin Exchange* in the aircraft concentrator.

The validation is limited to basic functional tests to show that the proposed federation architecture can be used to realize the local and global airline business processes motivated in Section 2. The virtual testbed is, however, designed such that it can support further in-depth validation activities.

5. CONCLUSION AND OUTLOOK

Within this paper a generic architecture for realizing airline business processes in a system of systems context alongside with design considerations for use case specific instantiations of that architecture are proposed. System of systems in this context refers to not limiting the scope to, for example, interacting systems installed in the aircraft. Depending on the information need of a specific airline business process, the generic architecture supports integrating various systems installed in the aircraft as well as ground systems into a shared communication network. The design considerations as well as the generic architecture have been deduced taking into account aviation security guidelines and principles. The architecture is based on modules, each built around a specifically configured RabbitMQ message broker to enable CSMIM-based intra-module message exchange. Through broker federation modules are connected to, additionally, support a contract-based inter-module communication.

An initial validation of the generic broker federation architecture has been performed by applying it to two use cases. A physically distributed and federated architecture following the proposed design considerations has been successfully implemented. It is composed of multiple aircraft modules and an aircraft concentrator module linked through a local network as well as a ground server

connected via a cellular-based internet connection. In addition, a virtual testbed has been set up that has been used to show successful integration of CSMIM-compliant nodes into the federated messaging architecture. Through virtualization the testbed can serve as an extensible platform for future in-depth studies.

Furthermore, in future work it is planned to integrate the proposed approach for federated aviation messaging architectures and the related design considerations into model-based methods for the development of networked aircraft cabin systems. This may include to provide guidance and validation suites to ensure compliance with the corresponding design principles, security standards, and the security domain model.

ACKNOWLEDGEMENTS

This work was supported by the LuFo VI-1 project “Lernendes Galley-Catering-System” (engl.: Learning Galley Catering System, LGCS, project number 20D1928C) funded by the Federal Ministry for Economic Affairs and Climate Action based on the decision by the German Bundestag.

Contact address:

fabian.maximilian.giertzsch@tuhh.de

REFERENCES

[1] LGCS: Learning Galley Catering System, https://www.tuhh.de/fks/en/research/lgcs (visited 2024/01/05)

[2] ARINC 853: Cabin Secure Media-Independent Messaging (CSMIM) Protocol, ARINC Cabin Systems Subcommittee, https://aviation-ia.sae-itc.com/standards/arinc853-853-cabin-secure-media-independent-messaging-csmim-protocol (2024)

[3] F. Giertzsch, C. Hertwig, U. Salomon and R. God, “Requirements and Technical Trade-Offs for a Communication Standard in a Data-Driven and Interconnected Aircraft Cabin,” SAE Int. J. Adv. & Curr. Prac. in Mobility 3(3):1197-1205, doi: 10.4271/2021-01-0011 (2021).

[4] MQTT Version 5.0, Edited by A. Banks, E. Briggs, K. Borgendale and R. Gupta, OASIS Standard (2019)

[5] AMQP Version 0.9.1, Advanced Message Queuing Protocol, Protocol Specification (2008).

[6] RabbitMQ Broker Federation, https://www.rabbitmq.com/federated-exchanges.html (visited 2024/01/05)

[7] F. Giertzsch, H. Hintze, B. Heinke, and R. God, “Network Design Criteria to Introduce Data Analytics within the Aircraft Cabin”, 7th International Workshop on Aircraft System Technologies (2019).

[8] ARINC 664, P5, “Aircraft Data Network, Part 5, Network Domain Characteristics and Interconnection” (2005)

[9] EUROCAE ED-203A, “Airworthiness Security Methods and Considerations“ (2018)

[10] S. Melzer, J. Speichert, O.C. Eichmann and R. God, “Simulating Cyber-Physical Systems Using a Broker-Based SysML Toolbox,” 7th International Workshop on Aircraft System Technologies (2019).

[11] H. Wang and S. Melzer, “Simulation of Ordering Processes across different Supply Chain Tiers in the Aviation Industry,” 2022 IEEE International Systems Conference (SysCon), Montreal, QC, Canada, pp. 1-6, doi: 10.1109/SysCon53536.2022.9773906 (2022)

[12] A. Prajapati, “AMQP and beyond,” 2021 International Conference on Smart Applications, Communications and Networking (SmartNets), Glasgow, United Kingdom, pp. 1-6, doi: 10.1109/SmartNets50376.2021.9555419 (2021)

[13] RabbitMQ Message Broker, https://www.rabbitmq.com/ (visited 2024/01/05)

[14] Erlang Programming Language, https://www.erlang.org/ (visited 2024/01/05).

[15] MQTT 5.0 support is coming in RabbitMQ 3.13, https://blog.rabbitmq.com/posts/2023/07/mqtt5/ (visited 2024/01709)

[16] O.C. Eichmann, J.G. Lamm, S. Melzer, T. Weilkiens and R. God, „Development of functional architectures for cyber-physical systems using interconnectable models,” Systems Engineering (2024), 1-19. https://doi.org/10.1002/sys.21761

[17] H. Hintze, F. Giertzsch and R. God, “Design Approach for Secure Networks to Introduce Data Analytics within the Aircraft Cabin,” SAE Int. J. Adv. & Curr. Prac. in Mobility 2(2) (2020), 737-746, https://doi.org/10.4271/2019-01-1853

[18] SAE ARP4761A, “Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment” (2023)

[19] R. God, S. Melzer, “Connected Cabin – Information Centric Operation of Future Connected Cabin, Teilvorhaben: Spezifikation und Integration cyber-physischer Betriebs- und Geschäftsprozesse, Schlussbericht“ (2020)

[20] J. F. Kurose, K. W. Ross, Computer networking: A top-down approach (5. ed., international ed.). Boston, Mass., Munich [u.a.]: Pearson Addison-Wesley (2010).

[21] RabbitMQ Federation Reference, https://www.rabbitmq.com/federation-reference (visited 2024/09/11)

[22] European Union Aviation Safety Agency (EASA), CS-25, “Certification Specifications for Large Aeroplanes” (2023)

[23] Y. Wu, Z. Yu, M. Wen, Q. Li, D. Zou and H. Jin, “Understanding the Threats of Upstream Vulnerabilities to Downstream Projects in the Maven Ecosystem,” 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), Melbourne, Australia (2023)