CEAS Aeronautical Journal
https://doi.org/10.1007/s13272-022-00608-z

ORIGINAL PAPER

Enhancement of the virtual design platform for modeling a functional system architecture of complex cabin systems

Mara Fuchs[1] · Yassine Ghanjaoui[1] · Jutta Abulawi[2] · Jörn Biedermann[1] · Björn Nagel[1]

Received: 15 February 2022 / Revised: 8 June 2022 / Accepted: 2 August 2022

Abstract

Model-based systems engineering (MBSE) is a fundamental approach for the end-to-end use of digital models in the development of complex systems. The aviation industry in particular, where system complexity is constantly increasing, needs new concepts and methods to overcome ecological and socio-economic challenges. Therefore, domain-specific models are needed for the design and evaluation of systems to support the various system investigations, such as requirements management, installation space optimization, or failure analyses. An end-to-end coupling and linking of these mostly heterogeneous systems offer many advantages (e.g. shorter development times) over working with isolated digital sub-models, natural language documents, and purely physical prototypes. In addition, digitalization allows global and interdisciplinary collaboration of multiple teams of experts on the same virtual product. Since this approach is particularly promising for the configuration of aircraft cabins, a virtual development platform is developed at the German Aerospace Center (DLR) for the conceptual design of the aircraft cabin and its systems. As a result, virtual prototypes of cabin configurations are quickly generated to allow new concepts to be visualized and investigated at an early design stage. Extending the conceptual cabin system design process with a functional system architecture and executable system architecture models promotes information traceability, early failure detection, and requirements verification. The methodology used for this purpose is presented in this paper. The systems modeling language (SysML) is used to build a model for the functional depiction of cabin systems and to link it to existing models of the conceptual cabin design process. The modeling is performed exemplarily for the passenger service functions. Subsequently, the results are automatically transferred to the virtual development platform to experience the generated cabin concept.

Keywords MBSE · SysML · Conceptual system design · Cabin · Cabin systems

✉ Mara Fuchs
mara.fuchs@dlr.de

Yassine Ghanjaoui
yassine.ghanjaoui@dlr.de

Jutta Abulawi
jutta.abulawi@haw-hamburg.de

Jörn Biedermann
joern.biedermann@dlr.de

Björn Nagel
bjoern.nagel@dlr.de

[1] Institute of System Architectures in Aeronautics, German Aerospace Center (DLR), Hein-Saß-Weg 22, 21129 Hamburg, Germany

[2] Department Fahrzeugtechnik und Flugzeugbau, HAW Hamburg, Berliner Tor 5, 20099 Hamburg, Germany

Abbreviations

A/C	Aircraft
ACS	Air conditioning system
act	Activity diagram
API	Application programming interface
ATA	Air transport association
bdd	Block definition diagram
CMS	Cabin management system
CPACS	Common parametric aircraft configuration schema
DLR	German Aerospace Center
FAS	Functional architecture for systems method
fbx	FilmBox
FR	Functional requirement
FSB	Fasten-your-seatbelt sign

Published online: 22 August 2022

ibd	Internal block diagram
ID	Identification number
IFE	In-flight entertainment
INCOSE	International Council of Systems Engineering
MBSE	Model-based systems engineering
NS	Non-smoking sign
OEM	Original equipment manager
OHSC	Overhead storage compartment
OOSEM	Object-oriented systems engineering method
PAX	Passenger
PLC	Product life cycle
PSC	Passenger supply channel
PSU	Passenger service unit
R	Requirement
RTS	Return-to-seat sign
SE	Systems engineering
SoSe	System of systems engineering
SysML	Systems modeling language
TLCR	Top-level cabin requirement
UC	Use case
UML	Unified modeling language
VR	Virtual reality
XML	eXtensible markup language

1 Introduction

Aircraft cabins and their systems are undergoing constant change. As aircraft structures evolve, e.g. due to new propulsion systems or more environmentally friendly aviation, new cabin concepts have to be adapted and developed quickly. This requires rapid development cycles from design to assembly. The integration of new technologies, such as fuel cells or hydrogen propulsion systems, also affects the design of the cabin and the layout of the systems. At the same time, innovative technologies enable new types of system synergies in the cabin. Airlines are demanding individualized, highly efficient cabin configurations. However, the development of new cabin configurations isn't as fast as it needs to be. It is, therefore, crucial to understand the complex interactions between the individual subsystems as early as possible to ensure a holistic and considerable increase in efficiency.

Aeronautics research at the German Aerospace Center (DLR) is therefore pushing forward with the progressive digitalization of the aviation industry. Interlinked models can represent complex system interrelationships and promote plausibility checks, data consistency, data tracing, life cycle monitoring, and cabin reconfigurations. At DLR, a process for designing conceptual aircraft cabin systems has been developed that automatically transfers modeling results to a virtual design platform. This interactive analysis environment enables early, virtual reality-based evaluation of design decisions. This research paper describes how a dynamic and functional system design model is integrated into the conceptual cabin system design process and how the models are linked. It shows how the model-based design approach contributes to the integration and reconfiguration of new technologies, promotes digitization, and thus enables faster development times.

2 Fundamentals

This section presents the methodological foundations for this work. First, essential terms and concepts of model-based systems engineering (MBSE) are explained. Second, principles and concepts of systems engineering (SE) are introduced. These have been used to extend the existing cabin system design and integration process to better meet system requirements and address challenges in system development.

2.1 Model-based systems engineering

The development and practical application of MBSE is being driven by members of the International Council on Systems Engineering (INCOSE). MBSE is the formalized application of modeling in all system development activities and across all phases of the product life cycle (PLC), with the goal of replacing the document-centric approach with the pervasive use of digital models [1]. The use of models is an essential foundation of all engineering disciplines; however, isolated, partial models are still predominantly used to perform specific tasks in the respective disciplines. There is a lack of practical implementations to cover all phases of the PLC and, most importantly, to interact and link the models from the different domains. Only linked, digital models can provide advantages over document-based approaches. These advantages include better traceability in development, uniformity of the information source, earlier detection of errors through the integration of different models, the use of synergies between disciplines, virtual validation in early development phases, and the individual consideration of different needs and requirements of all stakeholders [2].

The interdisciplinary exchange between the models requires a discipline-neutral, standardized modeling language. For this purpose, the systems modeling language (SysML) was developed as an abstract, graphical modeling language for all systems engineering activities. SysML is derived from unified modeling language (UML) and is based on the concept of object orientation. The language provides notations, semantics, and syntax elements for modeling at different levels of abstraction as well as diagrams as views of the content and relationships in the model [3].

Currently, SysML is being revised to better support the industry's goals, such as digitalization and traceability. The planned "SysML v2" is intended to eliminate the formalization deficits of previous language versions and provide

an expressive and precise metamodel. In addition, the new language specification defines a standardized application programming interface (API) to promote integrated system modeling as part of a multidisciplinary development environment. Thus, interoperability between models from different disciplines is improved and a holistic digitized PLC is supported [4].

2.2 Principles and concepts for a successful design process

In order to exploit the possibilities of digital technologies for industrial progress, digital and discipline-specific development models as well as physical systems and products should communicate and interact with each other in the future throughout the entire PLC [5]. In this context, the following concepts and approaches make an important contribution to the realization of this digitalization:

- Systems thinking and system-of-systems visions: offer a holistic perspective for complex systems, which are viewed as interactive and contributing parts of the entire context. Due to the complexity, optimal solutions can only be achieved by optimizing the interfaces and interactions between components. This approach is considered the basis for a successful "System of Systems Engineering" (SoSE) [6].
- Agile systems engineering: the classic systems engineering process follows a predefined and planned sequence of sequential process steps. The increasing complexity of systems and the desire for individual, configurable, and customer-oriented solutions or the rapid integration of technologies require the addition of agile methods to this process [7]. This extension enables fast and effective reactions to unexpected events. It is mainly achieved through modularization principles and interaction organization between modules. Thus, an agile architecture platform emerges as a common base for agile SE [6].
- Knowledge-based engineering: generated and captured knowledge is collected at every stage of the PLC, formalized and made available in digital form. This includes information or data about the product as well as the processes. The ontology used to share this knowledge must be precisely defined and agreed with the various stakeholders. This method enables a high degree of automation of the processes, shorter and more efficient development processes and the securing of important data and knowledge against possible loss [8].

In the following, the conceptual system design process for the aircraft cabin developed at DLR is presented. In this paper, this process is methodically extended based on the three approaches.

2.3 The digital conceptual cabin system design process

The DLR Institute of System Architectures in Aeronautics is pursuing the goal of digitalization in aeronautics and is developing an automated process for the methodical design of aircraft cabin systems and subsequent data visualization in virtual reality (VR) [9]. This process is intended to consider existing, not yet formalized expert knowledge from different disciplines already in early phases of the aircraft preliminary design and to use it for the optimal design of the cabin systems. In addition, data transfer to virtual reality helps to evaluate different design variants in a timely and cost-effective manner and to modify and optimize them in a virtual environment [10].

Figure 1 shows the implemented process. It starts with the import of aircraft structure data (e.g. frame positions) and cabin layout data (e.g. seat positions). The data format used for the information exchange is the common parametric aircraft configuration schema (CPACS), a hierarchically structured XML dataset [11]. This schema was developed at DLR. In the next step, this information is used for the layout of the cabin and its systems. Considering requirements, anthropometric aspects and defined design rules, the cabin systems and modules are arranged and placed by the design algorithm in Matlab (version R2020a).

The design algorithm in Matlab creates a three-dimensional arrangement of the individual cabin objects, which are represented by placeholders made of simple geometric shapes. Subsequently, the object information, the results of evaluation functions (e.g. accessibility of oxygen masks) and the links between the objects are exported to an XML file. The high-resolution 3D modeling of these objects is automated in the next step using the Blender graphics software (version v2.80.75) and the generated XML file. Either modified 3D models from a library or scanned and digitized models of physically existing cabin elements are used [12]. In the process, the 3D objects are named according to their ID. This later allows the 3D objects to be uniquely assigned to the object information in the XML file in the virtual environment. Finally, the data generated from the design algorithm in Matlab and the 3D geometric object models are imported into the virtual scene (Unity version 2018.4.27f1). Here, the virtual cabin model and its data can be interactively explored using VR specific hardware. To visualize or manipulate specific data in the VR, additional functions were implemented [13].

3 Methodical development and integration of a system architecture

The conceptual cabin design process is complemented with a system architecture that is systematically and analytically derived and integrated. In the following, the state of the art

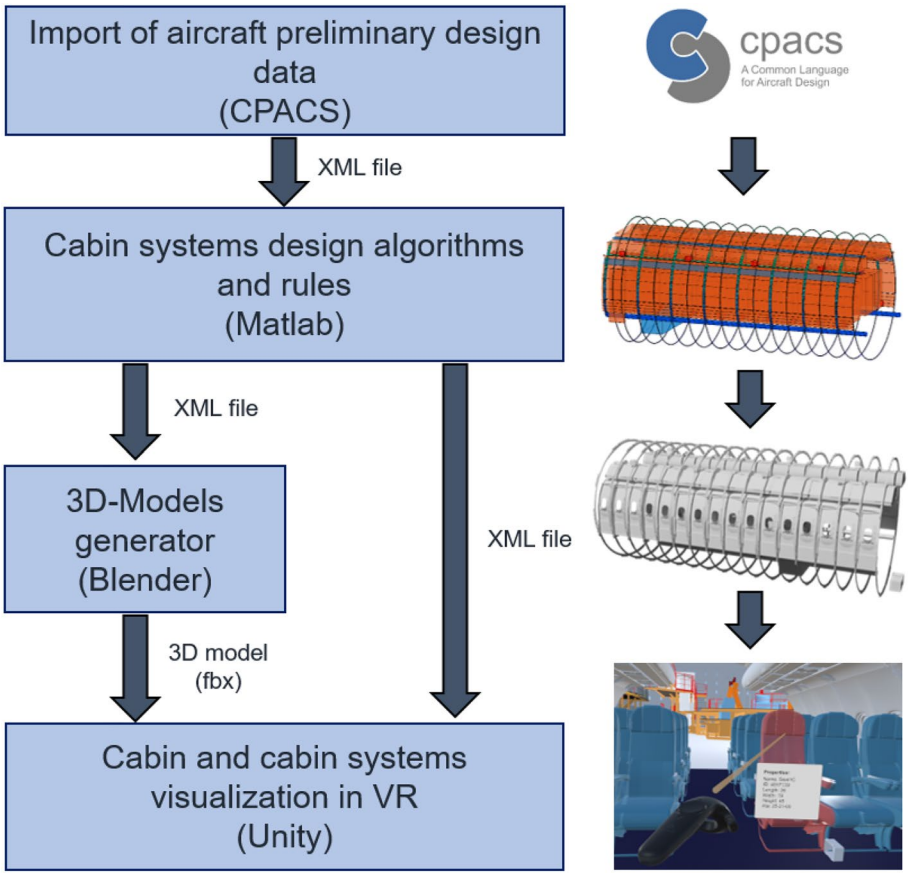

Fig. 1 Conceptual cabin design process and visualization in virtual reality (according to [9])

regarding MBSE methods and their implementation with MBSE tools is introduced first. Then, existing challenges in the process are explained and the model-based methodology developed to extend this process is presented.

3.1 Methods of model-based systems engineering

MBSE consists of three important pillars: the modeling language, tool, and methodology. The main goal of MBSE is to systematically document all information and decisions relevant for system development in digital models. Then these models are used in end-to-end digital processes without having to rely on human interpretation of informal sketches or natural language texts in between. In practice, MBSE approaches vary widely depending on the corporate culture, the development phase, or the level of detail in modeling [14]. Estefan gives a good overview and compares the main methodologies that have been developed and practically used until 2008 [15]. A major challenge remains to select, complement and combine the appropriate methods for each use case. For this work, the following methods are relevant:

- OOSEM: the object-oriented systems engineering method (OOSEM) is a model-based top-down approach, that uses the concept of object orientation and SysML. It includes basic systems engineering activities such as stakeholder and requirements analysis, architecture design, trade-off studies, and verification and validation. It also includes modeling techniques such as causal analysis, black-box and white-box considerations, logical decomposition, partitioning criteria, and variant design [16].
- Pragmatic systems modeling process (SYSMOD): this method assumes that every development builds on an existing technical initial solution. This solution is referred to as the base architecture and is the starting point for the co-evolution and constant refinement of requirements and architecture solutions (zigzag pattern). The solution finding process always tries to keep requirements and technologies in harmony with each other [17].
- Functional architecture for systems method (FAS): this method starts with the modeling of use cases of the intended system based on existing requirements and thus derives a functional system architecture. As a solution-neutral description of the system, it provides a good basis for finding an optimal physical system architecture and for modularizing the system [18].

In the selection of the methods, the three following aspects such as level of detail, automation possibility in the tool (plug-ins) and management of functionality of a system were decisive. Therefore, the FAS method was used in this work and the SysML was applied as an extension in the conceptual cabin system design process for the layout of the functional and logical architecture. Since the system of interest and the requirements are already known and only some minor changes occur in the development, the structure of the SYSMOD is used for the modeling process. With OOSEM, the scope for the method is specified and serves as a guideline.

3.2 Integration of the model-based system architecture into the conceptual cabin design process

In the area of customizing, a flexible architecture is needed in which different seat designs, for example, can be benchmarked against each other. The sole use of SysML for modeling the system architecture as well as the knowledge-based placement of the cabin system components under consideration of geometric requirements is not sufficient. Instead, a coupling of the two approaches is necessary to enable a holistic design of the systems considering all stakeholder needs. Depending on the system analysis, a different abstraction level (top-down) is required, which can be freely selected in each model: SysML-model for requirement management and functional architecture, Matlab-model for geometric placement and evaluation, and VR-model for design.

The cabin design process presented in Sect. 2.3 is extended by establishing and adding a methodology for systematically deriving the functional and logical system architecture. This enables an executable specification as well as a fast reconfigurability of the systems. Moreover, it extends the geometric modeling and placement of the cabin systems in the aircraft to include functional property modeling as well as requirements management. Figure 2 shows how the SysML model is coupled to the conceptual design process of the cabin. In total, the SysML model has three interfaces to the conceptual cabin design process. The first interface deals with the import of data from the preliminary aircraft design, which are used as requirement parameters for the design of the system architecture. These include, for example, the number of passengers to be transported by the cabin or the distribution of aircraft classes (business, economy). To read this data from the XML file, the integrated Java-library in the Cameo Systems Modeler is used. Subsequently, run-time objects of, e.g. type *CabinClass* are generated and the data are stored in the corresponding property parameters of these objects. The second interface is the coupling between the SysML model and the Matlab model. The built-in Matlab data interface of Cameo Systems

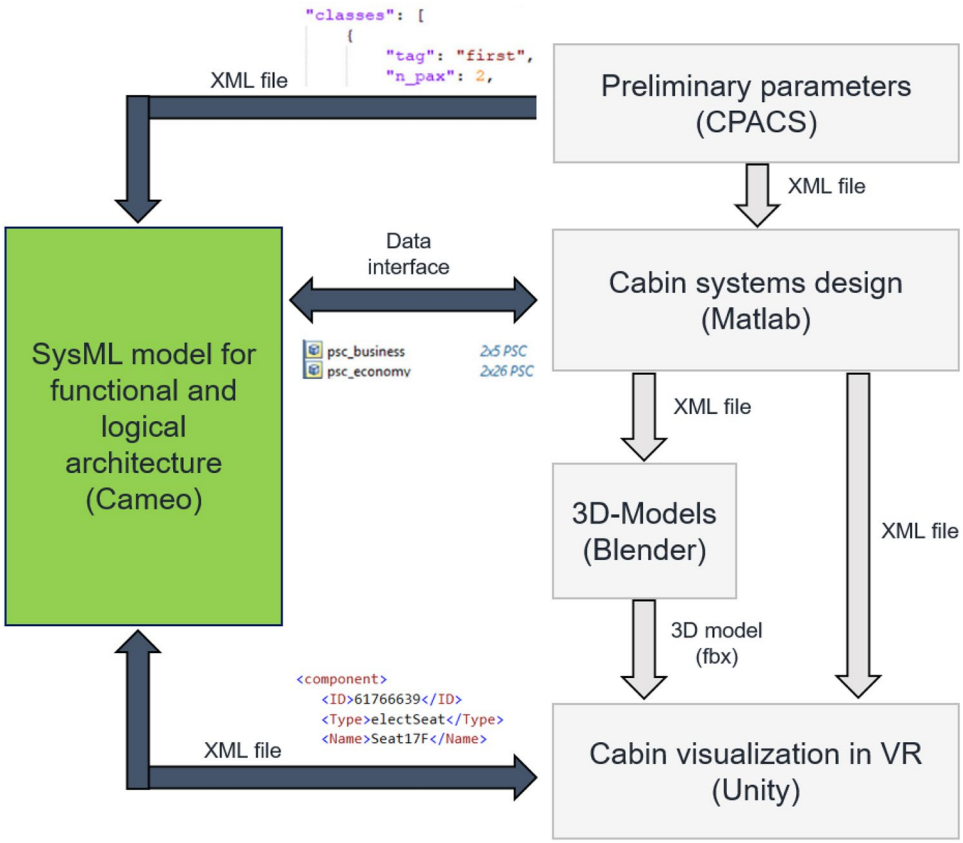

Fig. 2 Integration of the SysML model into the conceptual cabin design process and their interfaces

Modeler is used for this purpose. Once all the cabin components have been instantiated in the executable SysML-models and thus the system architecture has been created, the instantiated objects are automatically transferred to the workspace in Matlab via the data interface. These objects already bring along some properties (name, ID, ATA chapter affiliation, links). However, other properties are still empty (center points, dimensions). Therefore, these objects are now placed within the cabin's design space and the empty properties are filled with the results of the design process. Finally, through the data interface, the results and filled properties can be transferred back to the objects in the SysML model, since an exact assignment is made through the IDs. For the visualization of the cabin concept there is not a direct coupling between the model generator Blender and the SysML model. Instead, the visualization is done via data transfer from Matlab. Thereby all objects with their properties as well as their connections are exported to an XML file. The automated process in Blender reads this data and places 3D objects according to the object type and the calculated position in a three-dimensional model. Next, the 3D model is automatically loaded into the virtual environment. Through the interface between the VR environment and the SysML model, the geometry objects are then supplied with properties and information. Likewise, the data transfer takes place here with an XML file, from which the method in VR extracts all information depending on the currently executed function (for example: displaying links). All interfaces work automatically, so no manual transfer of XML files or object information is necessary.

Figure 3 presents the designed development and integration process of the functional system architecture. The model-based development of the system architecture is performed using SysML. The different process steps take place in different modeling languages or systems (vertical swimlanes) and at different abstraction levels of analysis and development (horizontal swimlanes). Throughout all development phases, aircraft design and configuration information are integrated into the modeling as system requirements. For this purpose, the data is extracted from CPACS and imported into the system analysis. This includes top-level system requirements as well as configuration parameters such as the number of passengers and aircraft class definitions. These requirements are key to the process and are continuously identified, derived, refined, and tracked in parallel with the architecture development process. In this research, requirements management is done with SysML as well. However, it can also be performed with special requirements management programs (e.g. DOORS) and integrated into the SysML models. Before starting the system analysis, the SysML model is organized. In order to promote uniformity and traceability of the model structure, a modeling guideline is created. Based on this, model elements, which are structured in packages and libraries are prepared for better clarity. The system analysis begins with a stakeholder and context analysis. Important stakeholder needs are recognized and interactions with the system environment are identified. System use cases are then derived from both the system analysis and the top-level requirements of the cabin.

The use cases represent the starting point for the FAS method, which is subsequently applied. The result of the method is a block-based functional architecture, in which the various functional blocks are linked together using the corresponding interfaces. Various activities for refining the use cases are allocated in logical functional groups. This results in a pure functional representation of the system that is generally valid and independent of the technical solution. In the last step in the architecture development, a definition of the logical base architecture is performed according to the SYSMOD approach. The base architecture corresponds to the current state of the art and is merely a generic solution that still needs to be refined or configured in detail. Suitable interfaces are modeled for linking the logical components. Respective information and material flows are specified for these interfaces. This enables an extensive consideration of each granularity level in details. However, logical and functional analyses must be addressed separately and linked to each other by allocating the functions to the components. Also, by defining the structure of the system in the logical layer, both functions and logical components regarding different granularity levels are analyzed.

Finally, the system architecture is linked to the external models for the purpose of process extension. In this process, the cabin design parameters are used to configure and instantiate the developed logical architecture. The instances are passed to the design model in Matlab where they are used as further parameterization rules and requirements for the conceptual design. After the design algorithm has geometrically placed the various systems in the 3D cabin assembly space, the geometric results are returned to the architecture description for further analysis. Additionally, functional scenarios for the developed system are modeled in SysML with structural and behavioral elements and diagrams. Lastly, these are transferred to the VR environment where they are visualized for subsequent interactive exploration and modification. The feedback of the results achieved in the virtual environment into the system architecture enables the tracking as well as the verification of the specified requirements.

4 Proof of concept: the passenger service unit

The application of the developed methodology presented in the following is based on the passenger service unit (PSU) as a proof of concept. The PSU provides important service functions to passengers and includes components such as

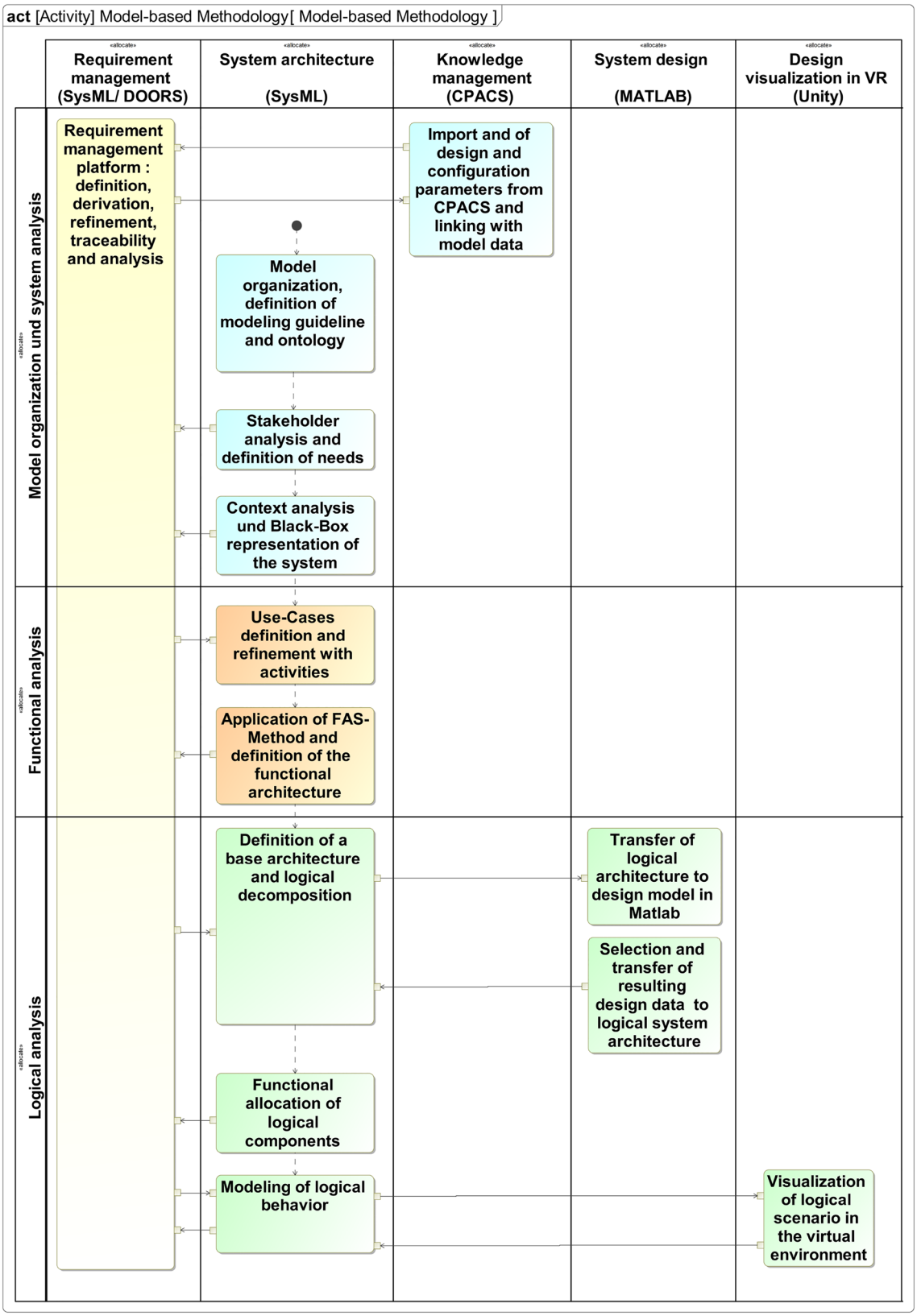

Fig. 3 Development and integration process of the used system architecture

reading lights, flight attendant call buttons, illuminated information signs, and loudspeakers. It is also a part of the passenger supply channel (PSC) along with the individually adjustable air outlets and oxygen masks [19]. These are also grouped as the passenger service functions. The high level of integration, interaction, and dependency on aircraft cabin configuration make the PSU complex. Thus, it is a suitable use case for exemplary application of model-based development and analysis of system architectures.

4.1 Stakeholder and system context analysis

Initially, all stakeholders involved in the passenger service functions are identified and modeled with the SysML language element "actor" (as a stick figure). The ontologically defined stereotype "StakeholderNeed" makes it possible to capture the needs of the stakeholders and to track them in the further development process. Relationships between stakeholders and their needs are mapped in a requirements diagram. Some of the top-level requirements of the stakeholder needs that have an impact on the conceptual design of the PSU are listed in the following:

- R1: The cabin shall provide an ergonomic, intuitive and user friendly interface for the usage of all its functions.
- R2: The cabin must provide oxygen to the passenger in case of an emergency.
- R3: The cabin must be configured for 180 passengers.
- R4: The design and layout of the cabin should consider assembly aspects during the development to facilitate a rapid and cost efficient manufacturing.

The stakeholders, who are considered users of the PSU, are then depicted in a context diagram (Fig. 4) along with the aircraft systems and environmental elements that interact with the PSU (modeled as blocks). In the context diagram, the PSU is located in the center as the system under consideration and is related to the environment elements and actuators. The predefined ontology is used to specify important features of the environment elements such as the ATA chapter. Special stereotypes like "Electrical" or "CabinModule" are used to classify the context.

In an internal block diagram (ibd), the interfaces of the PSU, which is considered as a black box, are described in its environment. In this turnover-oriented analysis, the material and data flows are defined by "interface blocks" and "ports" (Fig. 5). For example, mechanical forces act between the PSU and the baggage handling system (OHSC), and the PSU receives and transmits electrical signals. In addition, disturbance factors such as waste heat or dust from the environment are also modeled.

From this analysis, new requirements regarding the context elements of the passenger service functions as well as its interfaces were derived. Some new requirements are as follows:

- R5: The PSU shall provide interfaces to the oxygen system to flow oxygen in the PSU.
- R6: The PSU shall provide interfaces required for the data communication with external systems.
- R7: The PSU shall provide interfaces to the air conditioning system to allow air to flow in the PSU.
- R8: The PSU electronic components shall be electrically supplied and secured for their continuous functionality.
- R9: The PSU shall provide a user interface for the interaction with the passenger and crew.

These will be functionally refined and supplemented in the following steps.

4.2 Functional analysis and application of the FAS method

Functional analysis and the definition of a functional architecture separate the functions of the system as well as their interaction from the logical and physical solutions in order to meet functional requirements. For this purpose, the top-level requirements for the cabin and cabin systems are defined first. Functional requirements are further refined with use cases, which are specified with SysML relationships and use case diagrams until PSU-specific use cases can be identified. An overview of the results of the functional analysis and the derivation of the use cases can be obtained with the traceability matrix. Here, relationships to different model elements can be shown and traced. Figure 6 shows a section of the use cases for the PSU, where an ID, a name, and the actors involved are assigned to the use cases. Here, the term *General Use Case* refers to the external system use cases from which the PSU's was derived. These external systems are the contextual systems or subsystems that interact not only with the PSU but also those that include the PSU or some of its components. Their use cases are used as an initial point for the PSU to obtain PSU-related use cases.

The PSU use cases represent the starting point for applying the FAS method. They are refined in the first step with activities in activity diagrams to achieve a higher level of detail in the specification. Partitioning using the FAS-specific "I/O Partitions" makes it possible to differentiate activities for exchange with external systems and users from the system-internal activities. Subsequently, the generated activities are linked to functional groups. These group the activities according to common functionalities and are shown in the traceability matrix in Fig. 7. Here, the functional groups are listed on the left and assigned to the corresponding

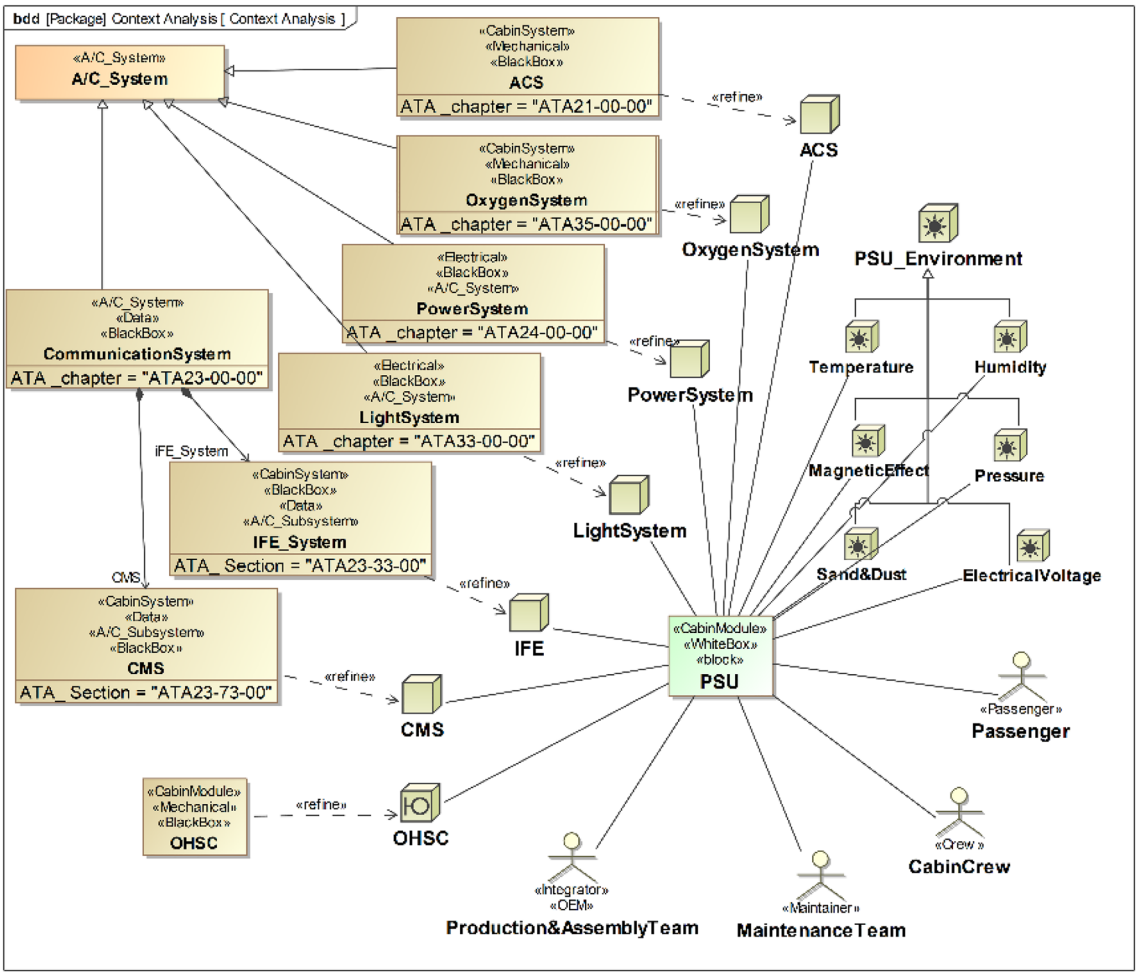

Fig. 4 Context diagram of the passenger service functions

activities via numbers. The arrow illustrates the mapping, with the arrowhead pointing from the tracked model element (functional groups) towards the mapped element (functional activity). The numbers represent the number of relationships with the subordinate activities that have been hidden and are not visible in the matrix.

Finally, blocks with corresponding interfaces are derived from the functional groups and linked with each other. The result of this functional interaction is described in the internal block diagram and represents a functional architecture for the PSU component (Fig. 8). The PSU is represented here by "System1".

This completes the modeling of the functional architecture of the PSU. The next step is to build the logical architecture.

4.3 Derivation of the logical architecture

In this step, logical components of the system are selected and combined with each other to find a technical solution for implementing the functions. Based on the SYSMOD approach, a base architecture for the PSU is used, which is strongly based on existing architectures from the aerospace industry (cf. [19]). The logical structure and hierarchical decomposition are mapped in a block definition diagram. Then, the interactions between the components are modeled in the internal block diagram. Figure 9 shows the defined internal logical structure of the PSU and the data flows to the individual system components. An allocation matrix enables to verify if all functional blocks are realized with logical components and promotes traceability to the higher-level analysis and requirements. Figure 10 shows this coherence.

4.4 Integration to the conceptual cabin system design process

The results of the system analysis and system architecture are linked to the conceptual cabin system design process and further analyzed. For this purpose, relevant information and data must be selected and exchanged with the

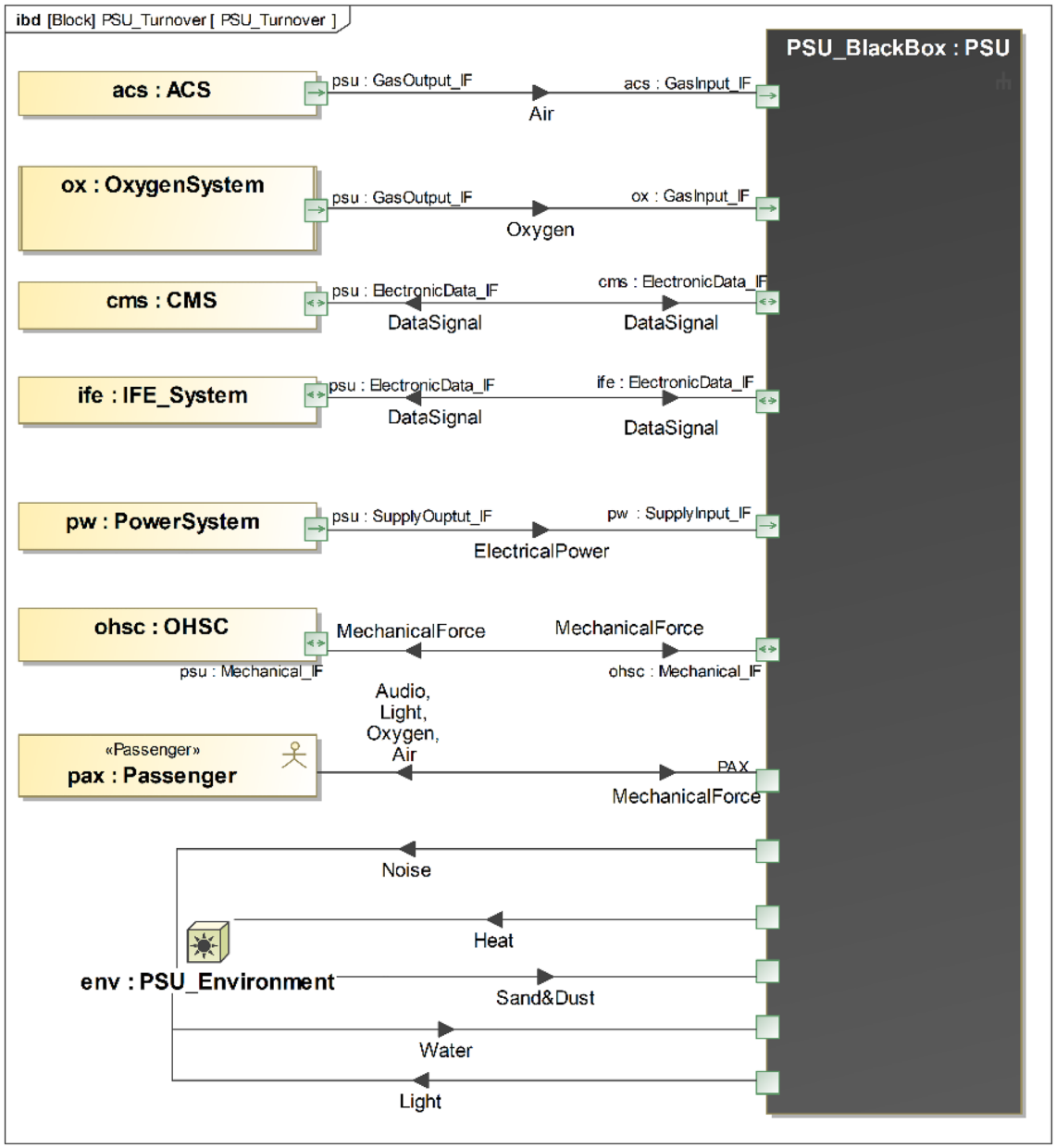

Fig. 5 Turnover diagram of the passenger service functions

respective models. This process starts in the case of the PSU with the import of cabin configuration parameters from CPACS, which determine the number of PSUs, their distribution in the cabin classes as well as their internal structure. These parameters are mainly the number of rows and seats per cabin class and the distribution of seats in the cabin (Fig. 11). Thus, the already defined logical architecture is linked to a specific configuration and instantiated. This interface is automated and synchronizes the SysML model with the CPACS data before each model simulation.

In the next step of the design process, the systems are conceptual designed and geometrically placed in the cabin area with the Matlab model. For this purpose, the SysML model provides data from the instantiated system architecture of the PSU. Figure 12 (1) shows the instantiated objects of the SysML model in the output window of the Cameo Systems Modeler™ (version 19.0). These PSU objects with corresponding internal structures and interfaces are automatically passed as input to the conceptual design model in Matlab. In the following the system architecture given by the SysML model is used. The properties and structures are taken from the SysML model (Fig. 12 (2)). Thus, the objects can be geometrically placed in the cabin using the defined rules and algorithms of Matlab. Moreover, further property

#	Id	Name	Associated Actor	Top-Level Requirement	△ General Use Case
1	PSU-UC-PA-02	Provide Cabin Crew Announcements to Passenger	CabinCrew CMS Passenger SafetyAuthority	TLCR.3 Cabin Management System FR.3 Cabin Announcements	Passenger Address Provide Passenger Service Functionalities
6	PSU-UC-PA-01	Provide Cockpit Crew Announcements to Passenger	CockpitCrew Passenger CMS SafetyAuthority	TLCR.3 Cabin Management System FR.5 Cockpit Announcements	Passenger Address Provide Passenger Service Functionalities
11	PSU-UC-CO-01	Provide Call Flight Attendant functionality	Passenger CabinCrew CMS	TLCR.3 Cabin Management System FR.4 PAX Call	Provide Comfort Provide Passenger Service Functionalities
17	PSU-UC-CO-02	Provide Reading Light for Passenger	CMS Passenger	TLCR.4 Genral Cabin Illumination TLCR.3 Cabin Management System FR.11 Reading Light	Provide Comfort Provide Passenger Service Functionalities
23	PSU-UC-EM-01	Provide Audible Indications for Emergency Signaling	Passenger SafetyAuthority CMS	TLCR.3 Cabin Management System FR.2 Audible Indications	Provide Emergency Evacuation Signaling Provide Passenger Service Functionalities
28	PSU-UC-IL-01	Provide Fasten Seatbelt Signs	Passenger CMS SafetyAuthority	TLCR.4 Genral Cabin Illumination TLCR.3 Cabin Management System FR.6 FSB Signs	Provide Illuminated Passenger Information Signs Provide Passenger Service Functionalities
33	PSU-UC-IL-02	Provide No Smoking Allowed Signs	Passenger CMS SafetyAuthority	TLCR.3 Cabin Management System TLCR.4 Genral Cabin Illumination FR.7 NS Signs	Provide Illuminated Passenger Information Signs Provide Passenger Service Functionalities
38	PSU-UC-IL-03	Provide Return to Seat Signs	Passenger SafetyAuthority CMS	TLCR.3 Cabin Management System TLCR.4 Genral Cabin Illumination FR.14 RTS Signs	Provide Illuminated Passenger Information Signs Provide Passenger Service Functionalities
43	PSU-UC-OX-01	Provide Oxygen Outflow for Passengers	CMS Passenger OxygenSystem	TLCR.2 Oxygen Supply FR.8 Oxygen Outflow	Provide Oxygen Masks for Crew and Passengers Provide Passenger Service Functionalities
51	PSU-UC-AC-01	Provide Passenger Individual Air Condition	Passenger ACS	TLCR.1 Air Outlets FR.9 Individual Air	Provide Passenger Service Functionalities
57	PSU-UC-MA-02	Provide System Condition Communication	CabinCrew CMS GroundCrew MaintenanceTeam ACS	TLCR.3 Cabin Management System TLCR.4 Genral Cabin Illumination FR.15 Systems Condition	Provide Passenger Service Functionalities
62	PSU-UC-RE-01	Provide recorded Announcements	CabinCrew CMS Passenger	TLCR.3 Cabin Management System FR.12 Recorded Announcements	Provide Records Provide Passenger Service Functionalities
67	PSU-UC-RE-02	Provide Recorded Boarding Music	Passenger CMS CabinCrew	TLCR.3 Cabin Management System FR.13 Boarding Music	Provide Records Provide Passenger Service Functionalities

Fig. 6 Traceability matrix for the PSU use cases with corresponding model elements

parameters can be determined and stored as attributes in the object-oriented model. The result of the placement is visualized in Matlab by the three-dimensional arrangement of the corresponding objects in the cabin (Fig. 12 (3)). The results of the placement are then transferred back into the SysML architecture model and used for further analysis. For example, the resulting length of the power supply cables for each PSU can be used to estimate the power requirements for them. Furthermore, the final cabin concept can be visualized and interactively examined for further analysis in the virtual reality environment (Fig. 12 (4)). Here, the PSU is highlighted in red and the user can see its object properties

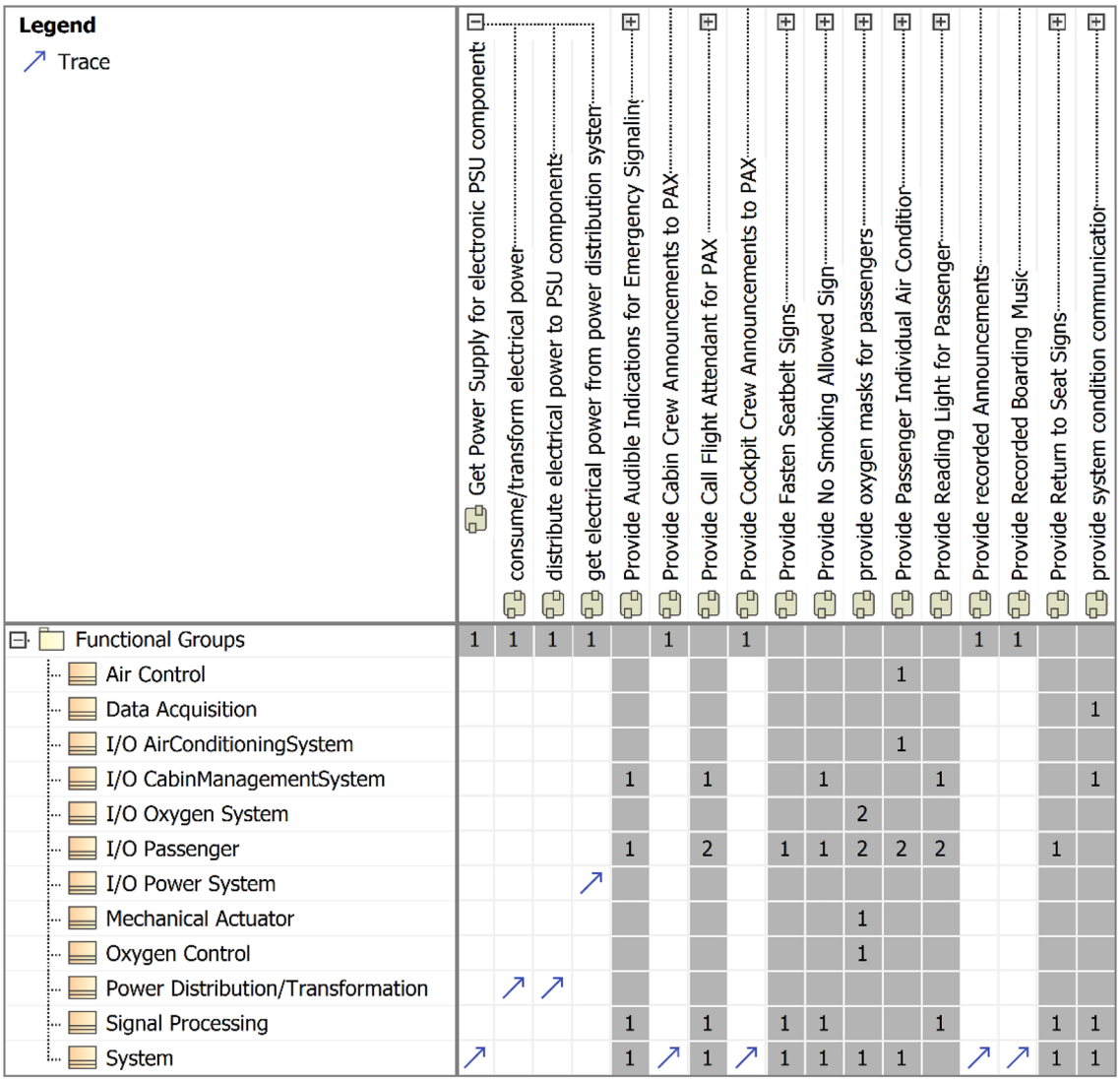

Fig. 7 Traceability matrix for functional blocks with corresponding activities

in a panel. Thus, the construction of a functional and logical system architecture for the passenger service functions is complete. The modeling of the PSUs now enables a decoupled design based on functional criteria, considering the various requirements and interactions. Thereby, the integration of the system architecture into the design process offers the possibility of using the data from the various models for further analyses.

The entire process from importing the top-level requirements to the 3D model in the virtual environment is done in 40 min for an A320 configuration. By applying the model-based approach and coupling the individual models with each other (SysML model, Matlab model and VR model), design and/or position changes can be performed and evaluated directly. For example, different arrangements of the PSU elements in the cabin can be investigated considering the functional as well as appearance requirements. Furthermore, the models can be easily extended and new rules or systems can be added. This significantly shortens the preliminary design process. The use of IDs ensures precise identification and assignment of the objects in each of the models. The IDs are created randomly in the range 10.000×10.000, thus keeping the error of identical IDs low. In addition, data transfer between all models is done through automated interfaces, eliminating the need for manual intervention and ensuring data consistency. The application of a consistent ontology based on the ATA chapters ensures that the approach will be used in the future and thus meets the classification of aircraft systems according to the international standard. Also, the SysML objects are adapted to the defined classes and structures in Matlab. Last, the connection to the CPACS structure enables further processing of the data and

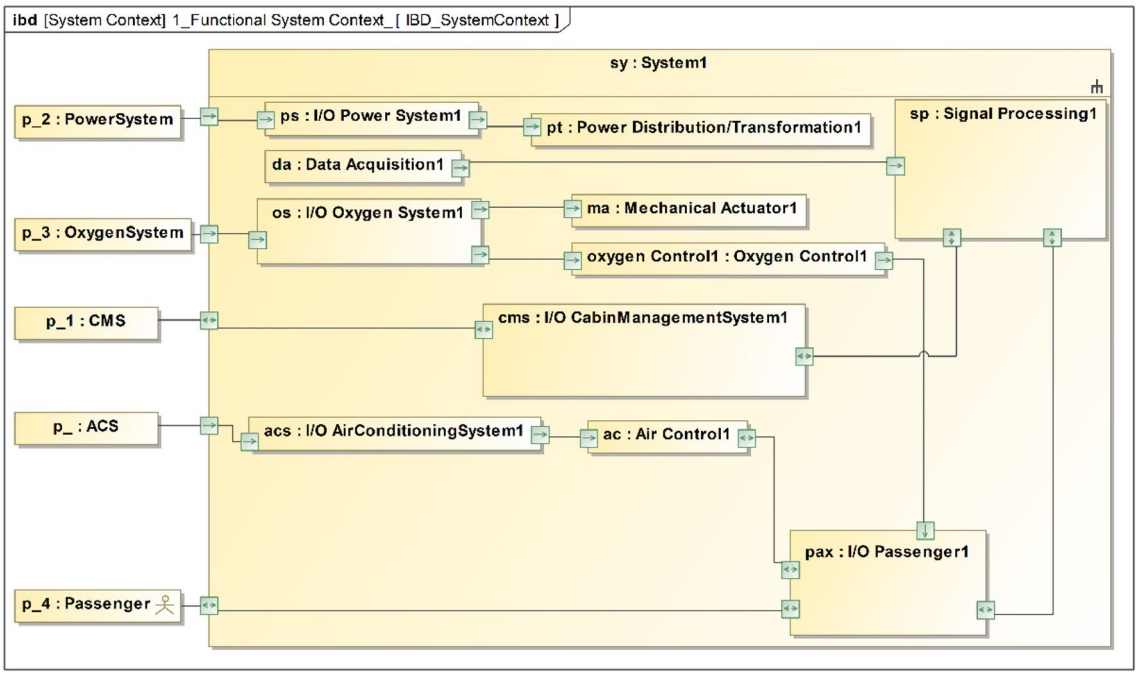

Fig. 8 Functional architecture of the passenger service unit

promotes the exchange of these in the aircraft preliminary design. Nevertheless, the presented method depends on the modeler. Thus, when implementing the method, care had to be taken to ensure that the requirements are quantifiable and that the SysML language is applied correctly. On the tool side, there is a validation for correct application of the SysML language, but this had to be additionally checked and customized if necessary.

5 Conclusion and outlook

In this work, it was shown how the conceptual cabin system design process can be extended to include a model-based functional system architecture. For this purpose, a process was developed that can be used to methodically design passenger service functions. The system architecture is based on the stakeholder and system context analysis and is derived from the resulting functional analysis. Based on the FAS method, a modular separation between the system functions and the corresponding technical solutions was achieved, thus creating a high degree of agility and flexibility in the development process. In addition, a linkage of all system elements and analysis results could be achieved with the help of the SysML relationships and the object-oriented concept. The systematic derivation of the system architecture led to a high degree of traceability and a noticeable reduction in complexity.

Subsequently, the system architecture in SysML was integrated into the cabin system design process. The interoperability between the different models and the data exchange made it possible to link the geometric design of the systems with a systematically derived system model. The use of external model data has allowed specific analyses to be performed at early stages of development.

The next step is to connect the SysML model to the virtual development platform in order to transfer and visualize the simulation protocols of the communication scenarios between PSU and the cabin management system in the virtual cabin model.

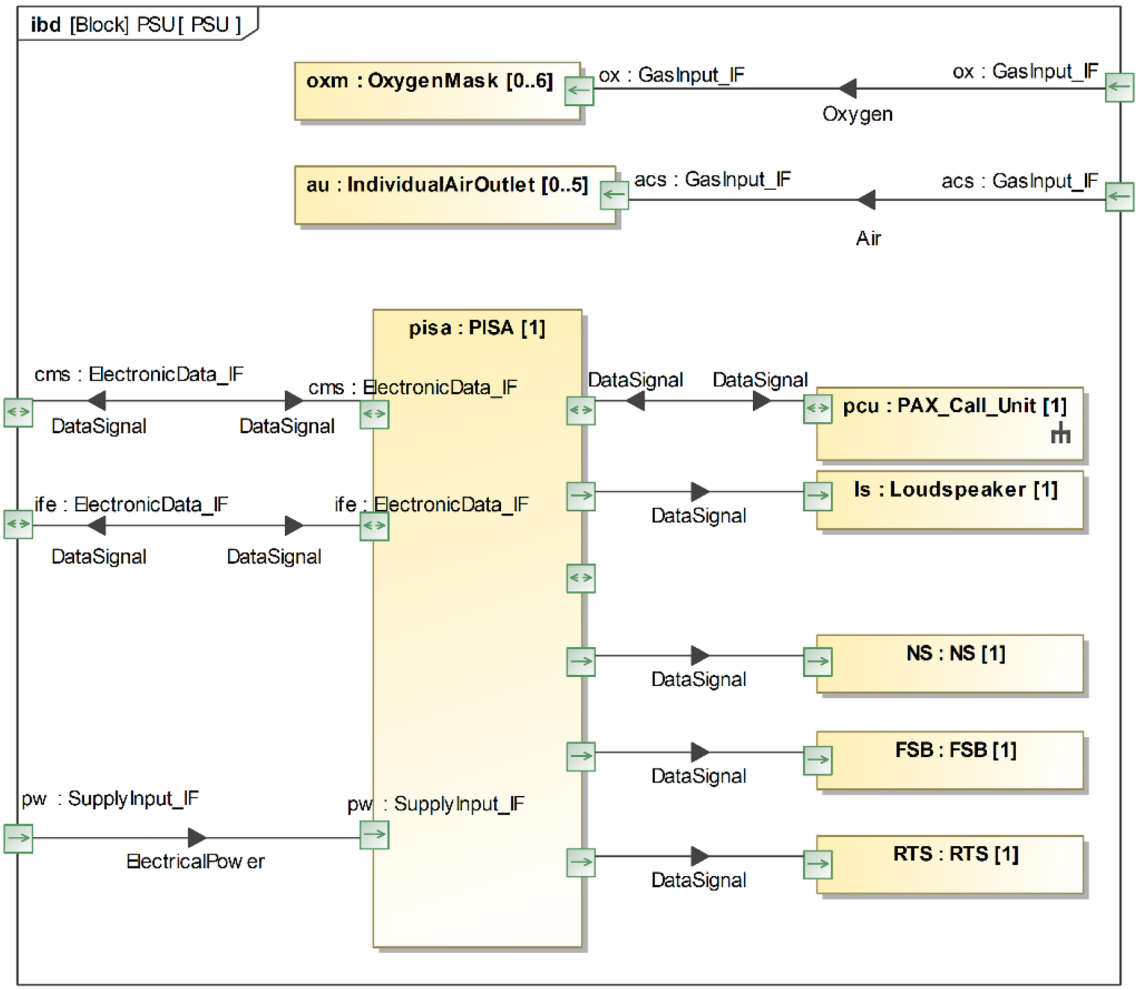

Fig. 9 Internal logical structure of the passenger service unit

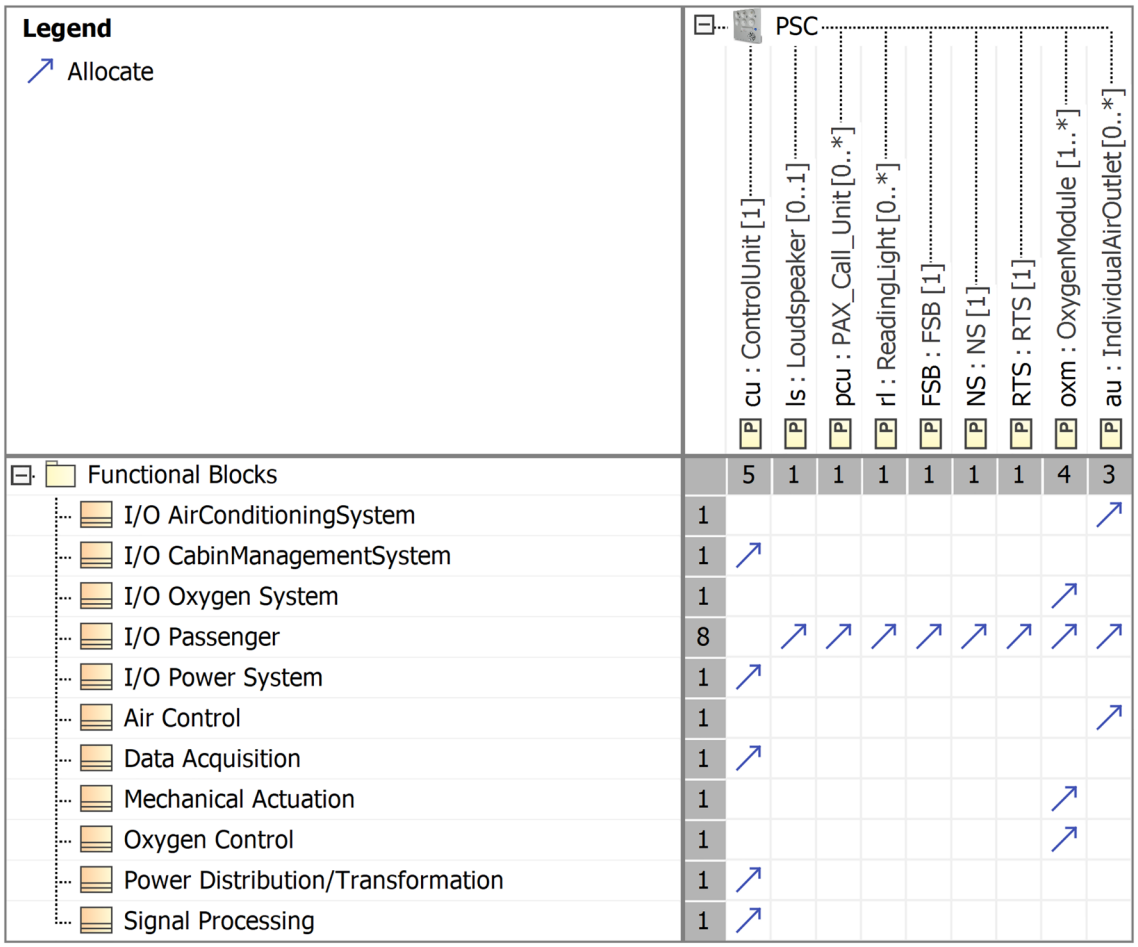

Fig. 10 Allocation matrix for functional blocks to logical components

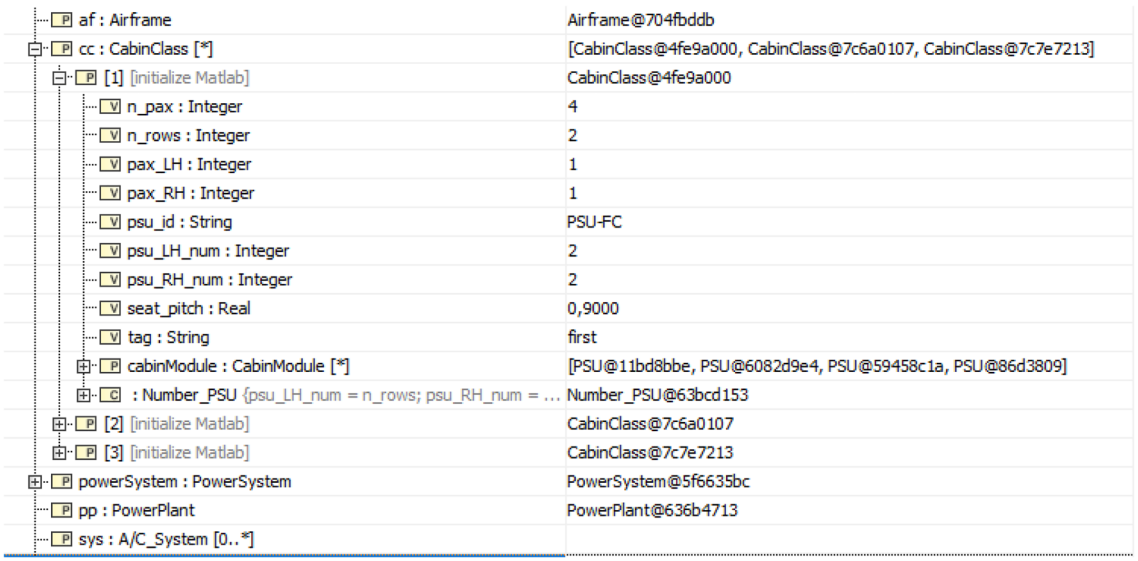

Fig. 11 Data imported from CPACS in the SysML model

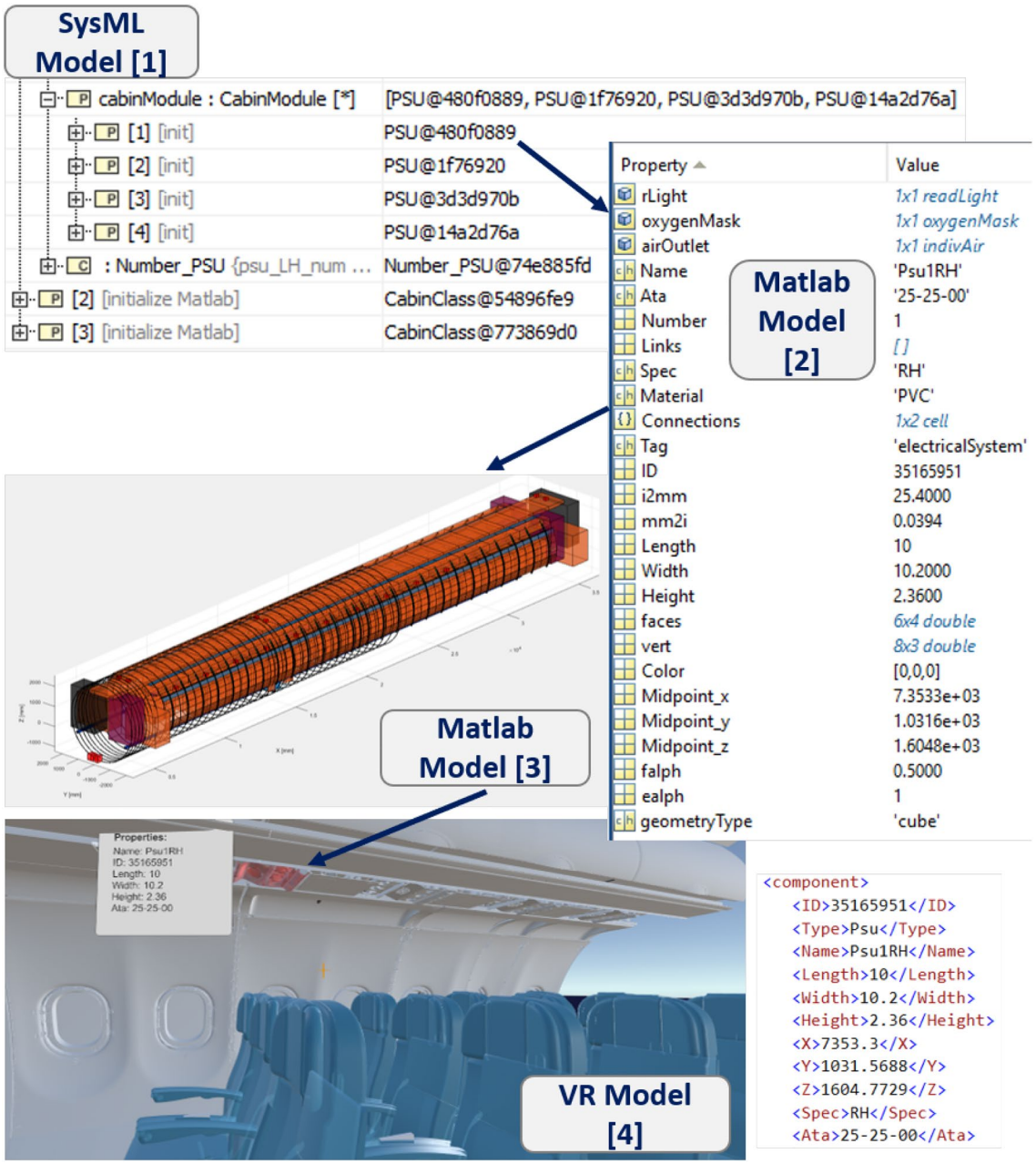

Fig. 12 Object generation in Cameo (1) and subsequent transfer to Matlab (2) as well as visualization of the cabin in Matlab (3) and in the virtual environment (4)

Funding Open Access funding enabled and organized by Projekt DEAL. No funding was received for conducting this study.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. International Council on Systems Engineering (INCOSE).: A Systems Engineering Vision 2020, Seattle. International Council on Systems Engineering, Version 2.03. INCOSE-TP-2004-004-02 (2007)
2. Dickerson, C.E., Mavris, D.: A brief history of models and model based systems engineering and the case for relational orientation. IEEE Syst. J. **7**(4), 581–592 (2013). https://doi.org/10.1109/JSYST.2013.2253034
3. Object Management Group (OMG).: OMG Systems Modeling Language (OMG SysML^{TM}). Version 1.6 (2018). https://www.omg.org/spec/SysML/1.6/PDF. Accessed 19 Jan 2022
4. Friedenthal, S.: Requirements for the next generation systems modeling language (SysML®v2). INSIGHT **21**(1), 21–25 (2018). https://doi.org/10.1002/inst.12186
5. Huang, J., Seck, M.D., Gheorghe, A.: Towards trustworthy smart cyber-physical-social systems in the era of Internet of Things. In: 2016 11th System of Systems Engineering Conference (SoSE), pp. 1–6 (2016). https://doi.org/10.1109/SYSOSE.2016.7542961
6. Walden, D.D., Roedler, G.J., Forsberg, K.: Systems Engineering Handbook. INCOSE-TP-2003-002-04. Wiley, Hoboken (2015)
7. Finkel, S., Märkl, S., Kessler, C.: Zurück in die Zukunft: Systems Engineering! In: Deutsche Gesellschaft Für Luft- und Raumfahrt—Lilienthal-Oberth e.V. (2020). https://doi.org/10.25967/530225
8. Chapman, C., Preston, S., Pinfold, M., Smith, G.: Utilising enterprise knowledge with knowledge-based engineering. Int. J. Comput. Appl. Technol. **28**(2–3), 169–179 (2007). https://doi.org/10.1504/IJCAT.2007.013354
9. Fuchs, M., Beckert, F., Gopani, M., Gindorf, A., Nagel, B.: Virtuelle Realität im digitalen Designprozess von Flugzeugkabinensystemen. In: Deutsche Gesellschaft Für Luft- und Raumfahrt—Lilienthal-Oberth e.V. (2020). https://doi.org/10.25967/530007
10. Fuchs, M.K., Beckert, F., Biedermann, J., Nagel, B.: Experience of conceptual designs and system interactions for the aircraft cabin in virtual reality. In: AIAA Aviation 2021 Forum. https://doi.org/10.2514/6.2021-2773
11. DLR Institute for System Architectures in Aeronautics.: CPACS: Common Language for Aircraft Design. Online (2021). www.cpacs.de. Accessed 23 Aug 2021
12. Rauscher, F., Biedermann, J., Gindorf, A., Meller, F., Nagel, B.: Permanente geometrische Digitalisierung der Flugzeugkabine zur Änderungsnachverfolgung. In: Deutsche Gesellschaft Für Luft- und Raumfahrt—Lilienthal-Oberth e.V. (2020). https://doi.org/10.25967/530008
13. Fuchs, M., Beckert, F., Biedermann, J., Nagel, B.: A collaborative knowledge-based method for the interactive development of cabin systems in virtual reality. Comput. Ind. **136**, 103590 (2022). https://doi.org/10.1016/j.compind.2021.103590
14. Weilkiens, T., Scheithauer, A., Di Maio, M., Klusmann, N.: Evaluating and comparing MBSE methodologies for practitioners. In: 2016 IEEE International Symposium on Systems Engineering (ISSE), pp. 1–8 (2016). https://doi.org/10.1109/SysEng.2016.7753174
15. Estefan, J.A.: Survey of model-based systems engineering (MBSE) methodologies. INCOSE MBSE Initiative, Rev. B (2008). https://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf. Accessed 19 Jan 2022
16. Friedenthal, S., Moore, A., Steiner, R.: A practical guide to SysML: the systems modeling language, 3rd edn. The MK/OMG Press, Morgan Kaufmann, Boston (2015). https://doi.org/10.1016/B978-0-12-800202-5.04001-7
17. Weilkiens, T.: Systems Engineering mit SysML/UML. dpunkt Verlag, Heidelberg (2008) **(ISBN: 978-3898645775)**
18. Lamm, J.G., Weilkiens, T.: Funktionale Architekturen in SysML. In: Maurer, M., Schulze, S.-O. (eds.) Tag des Systems Engineering 2010, pp. 109–118. Carl Hanser Verlag, München (2010)
19. Fuchs, M.: Ein MBSE-Ansatz Für die Auslegung Von Flugzeugkabinen Am Beispiel Eines Passenger Service Channel. Master Thesis. Technische Universität Hamburg (2018)