CEAS Aeronautical Journal
https://doi.org/10.1007/s13272-022-00610-5

ORIGINAL PAPER

Expansion of the cabin description within the CPACS air vehicle data schema to support detailed analyses

Jan-N. Walther[1] · Christian Hesse[1] · Marko Alder[1] · Jörn Y.-C. Biedermann[1] · Björn Nagel[1]

Received: 28 February 2022 / Revised: 3 August 2022 / Accepted: 8 August 2022

Abstract
The air vehicle data schema CPACS (Common Parametric Aircraft Configuration Schema) provides a large variety of options for describing passenger aircraft on different levels of fidelity and exchanging product information in collaborative processes. The section describing the aircraft cabin has been applied primarily in the context of preliminary design processes in the past. However, in the wake of recent developments to integrate more detailed analyses in fields such as vibro-acoustics or passenger comfort, the limitations of the present cabin definition have become increasingly obvious. In this paper, a revised version of the cabin definition is, therefore, presented, which has been adopted as of CPACS version 3.4. A key objective of the new definition is the integration of high-fidelity component geometry models, provided as either polygonal 3D meshes or CAD geometry. For efficient data storage, individual components are collected in a library node in CPACS and subsequently placed inside the fuselage via references. The range of available cabin component types is extended by including paneling elements and luggage compartments. Furthermore, nodes for referencing structural elements are provided to enable modeling of structural connections, e.g., in finite-element models. Aside from the sheer parametric description of the cabin in CPACS, its correct geometric interpretation is also a key aspect, which will be highlighted throughout this work. It is demonstrated using a reference implementation, which allows for the generation of 2D and 3D models for validation.

Keywords CPACS · Aircraft · Cabin · XSD · Virtual mock-up · FEM

1 Introduction

In times of significant socioeconomic changes due to the advancing climate change, the aviation community, too, needs to provide radically new, sustainable technologies more than ever. Digitalization is an indispensable instrument to realize such developments. Two aspects must be highlighted especially: the first aspect is the provision of analysis capability using numerical simulation, which significantly reduces the need for physical tests. To gain a holistic view of the aircraft product, it is insufficient to only take into account the classical disciplines such as aerodynamics and structures. Instead, disciplines such as cabin design provide key boundary conditions as well, which can have a fundamental impact on the overall design, thus potentially impairing the success of an aircraft development program.

The second important aspect of digitalization is the integration of the disciplinary analysis competences along a digital thread, to enable seamless collaboration among disciplinary departments or institutions and master the complex multi-disciplinary nature of aircraft design in the virtual domain. The establishment of suitable interfaces for data exchange is an essential step towards this goal. The air vehicle data schema CPACS (Common Parametric Aircraft Configuration Schema) [1, 2] constitutes one such interface, which serves to collect and distribute knowledge provided by different disciplines. Aside from classical disciplines such as preliminary design and structural design, cabin design is also among the disciplines supported by CPACS.

The original cabin definition in CPACS has been proposed by Fuchte et al. [3] in order to take into account cabin design aspects in preliminary design. However, new requirements, resulting from the incorporation of detailed analyses, have rendered this definition insufficient. Therefore, a revised version of the cabin definition is introduced

✉ Jan-N. Walther
jan-niclas.walther@dlr.de

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

Published online: 23 August 2022

in this paper, which has replaced the original definition as of CPACS version 3.4.

In the following, the objectives for the development of the new cabin definition are stated in Sect. 1.1. Then, a short introduction to the capabilities for modeling air vehicles in CPACS is provided in Sect. 2, as well as some technical background on the underlying schema definition. In Sect. 3, the new cabin definition is discussed in detail. Finally, the modeling capabilities of the new cabin definition are demonstrated in two application examples in Sect. 4.

1.1 Objectives

By developing a new cabin definition for CPACS several goals are pursued. First, the derivation of detailed geometry models for human factors analysis applications using virtual reality [4–6] from CPACS data sets is to be enabled. On the one hand, this requires the capacity to reference external cabin component models. On the other hand the scope of components supported in CPACS must be expanded to include components such as sidewall panels and luggage compartments. These components have not been supported by CPACS so far but play an essential part when building an immersive virtual representation of the cabin.

Another key goal is to improve the interconnection between the cabin and fuselage structure definitions. This is beneficial when including cabin elements in structural analyses using the finite element method (FEM), e.g., for crash and ditching [7, 8] or vibro-acoustics [9] problems. Furthermore, redundant definitions in CPACS can be avoided by favoring links and references over explicit definitions, which come with an inherent risk of inconsistent data sets.

In general, an efficient approach to modeling is pursued, which avoids data duplication as much as possible. At the same time, the accessibility of the schema for the user should be preserved. To this end, previously established best practices for modeling in CPACS, e.g., as found in the structural definition, have been considered and previously available standard types in CPACS have been used where possible. Furthermore, the documentation within the schema has been expanded.

2 Background

In this section, a short overview of the modeling facilities of the CPACS air vehicle data schema is given. Furthermore, some background is provided on data modeling using schemas in order to illustrate the technical basis for implementing the extensions of the cabin description.

2.1 CPACS

As explained in Sect. 1, the Common Parametric Aircraft Configuration Schema (CPACS) is being developed at the German Aerospace Center (DLR) in order to enable distributed collaborative aircraft design processes. It has already been applied successfully in numerous projects for both internal data exchange among different institutes of the DLR [10–12] and communication with external partners [13, 14].

Essentially, CPACS data sets are human-readable text files in extended markup language (XML) [15] describing air vehicles using a hierarchical data structure, which can contain parameter values, as well as other information, such as texts for documentation or settings for associated tools. An excerpt of the structure is shown in Fig. 1. The level of fidelity of the product description can be increased successively by filling in the available nodes, starting with a simple description of the outer geometry up to detailed disciplinary information, e.g., on structural concepts [16]

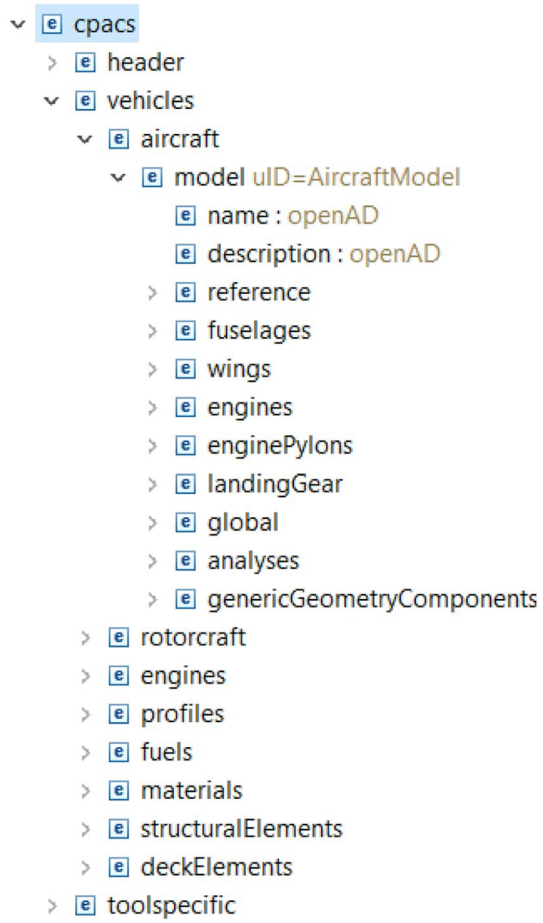


Fig. 1 CPACS XML tree [20]

or high-lift systems [17]. In addition, operational aspects can also be considered, e.g., by providing mission profiles [18, 19].

Translating the parametric data from CPACS to geometry models requires the application of external algorithms, which are usually combined into software libraries. To this aim, DLR provides the TiGL Geometry Library [21, 22]. A graphical user interface—the TiGL Viewer, shown in Fig. 2—is shipped as part of the library, in order to provide new users with a low-threshold entry point to CPACS. Aside from TiGL, more specialized geometry libraries are being developed, e.g., for aerodynamics [23, 24] or fuselage and cabin [25].

A comprehensive online documentation is available for CPACS, which provides detailed information on the interpretation of the different nodes in the schema. In addition some conventions have emerged from practical application of CPACS, which affect the cabin definition presented in the following. First, all values in CPACS should be provided in SI units. Second, the origin of the fuselage is usually placed at the tip of the fuselage at the height of the center of the cylindrical section (for conventional tube-and-wing configurations). As illustrated by Fig. 2, the x axis denotes the longitudinal axis of the aircraft, whereas the z axis points upwards. Consequently, the y direction points to the right in flight direction.

2.2 XML schema definition

As mentioned previously, CPACS files are XML text files. The CPACS schema is, however, always provided in the form of an XML schema definition (XSD) file [26]. Schema definitions can be used to precisely determine the structure of an XML data set, i.e., the hierarchy or the allowed number of occurrences of a given node. XML files can be validated against an XSD schema to assert their syntactic correctness. This is an important prerequisite to allow for reliable automatic processing of the otherwise freely editable XML text files.

XSD schemas can be visualized intuitively as block diagrams, which reflect the hierarchical structure. An example is given in Fig. 3. Aside from the hierarchy, several other properties of the schema are shown: A simple solid border of a node block implies that it should occur exactly once. If multiple occurrences are allowed, two stacked blocks are shown, along with an annotation providing the range for the number of occurrences. Optional nodes are bounded by a dashed line. Block diagrams are used in the following to depict the node types defined as part of the new cabin description.

Node attributes are not shown in the diagram, but an equally important part of XML. In contrast to child nodes, attributes are not intended to provide information that is to be processed, e.g., parameter values. Instead, they are meant to describe properties of the node itself, providing hints on how the contained data should be processed.

3 Cabin description using CPACS

In the following, the new cabin definition is presented in detail. The basic approach is illustrated by Fig. 4. In order to avoid data duplication, as stated in the objectives, the cabin components are collectively predefined in a hierarchy level outside of the actual cabin description, the so-called *component library*. The idea of a library node is adopted from the structural element definition node. It can be understood as a store of predefined semi-finished parts or bought-in components, which can be utilized to assemble the product. As such, the component library provides the geometric description of the component. In CPACS, the entries can be referenced from the actual cabin node inside the fuselage an arbitrary number of times, thus creating the *cabin instance*. Here, only the positions of the individual

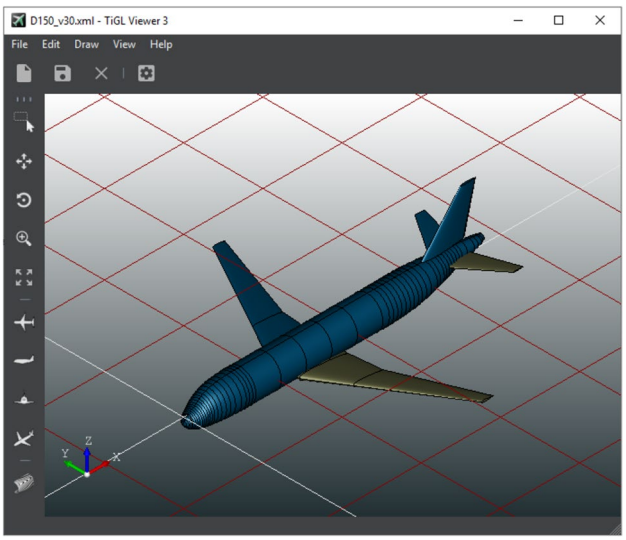


Fig. 2 CPACS visualization in TiGL Viewer 3 [20]

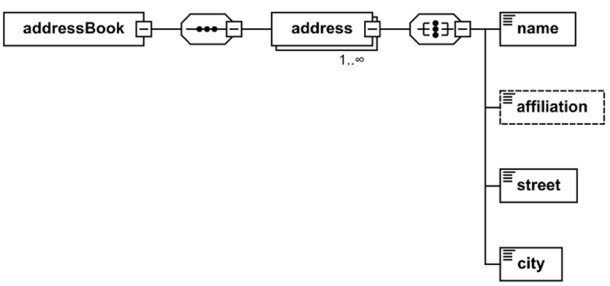


Fig. 3 XSD diagram representation address book example

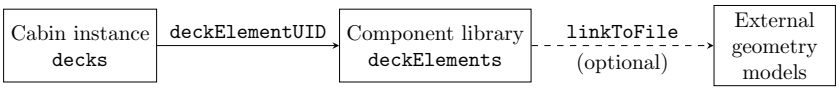


Fig. 4 Fundamental structure of the new cabin definition

component instances are provided, but no new geometry is defined. The component library definition in the `deckElements` node is described in Sect. 3.1 followed by a description of the cabin instance definition inside the fuselage using the `deck` node in Sect. 3.2.3.

3.1 The cabin component library `deckElements`

The `deckElements` node provides the component library and is, therefore, an important part of the new CPACS cabin definition. Unlike the cabin component definition in the previous CPACS cabin definition, it is not subordinate to any fuselage or deck instance, but placed directly under `/cpacs/vehicles`, similarly to, e.g., structural profile definitions. Entries, which are defined at this level, can in theory be referenced by any aircraft component found in CPACS. In the following, the supported cabin component types are first listed and explained. Then, the way components are described is examined in detail, taking the seat module as an example. Finally, the possibilities to store various kinds of metadata specific to certain component types are addressed shortly.

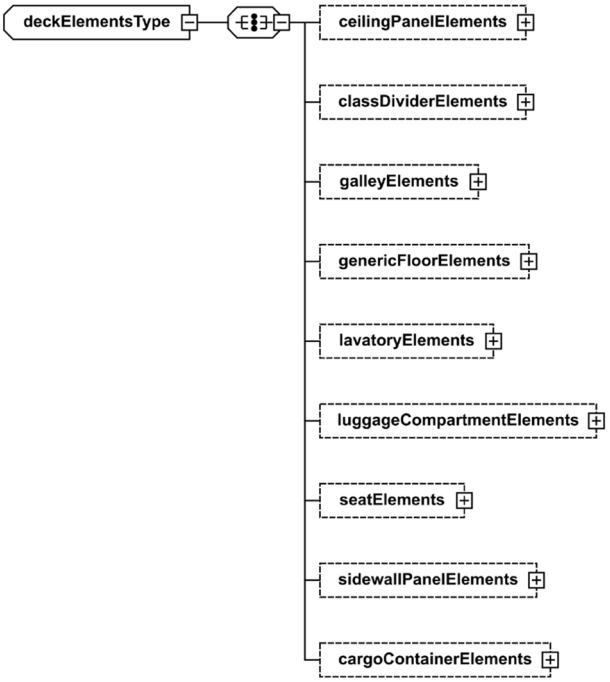


Fig. 5 XSD diagram of the subsidiary nodes of `deckElements`

3.1.1 Supported component types

The diagram in Fig. 5 shows all immediate children to the `deckElements` node. A total of nine different cabin element types are available. Some types, such as seat modules (`seatElements`) and generic floor-based elements (`genericFloorElements`) are based on types from the old cabin definition.

In addition, the new cabin definition also makes it possible to explicitly describe specific cabin elements, such as galleys or lavatories, using dedicated nodes. In the old definition, this differentiation was instead implemented using a node named `type` in the generic floor element definition. The drawback of this approach is, that important metadata, which is specific to a certain element type, such as the number of trolleys in a galley, cannot be stated explicitly. Instead the multi-purpose node `number` is provided, which must be interpreted in different ways, depending on the given type. Constructs such as this are avoided in the new definition for the sake of improved clarity and accessibility, as stated in the goals. Therefore, the dedicated cabin element definitions in the new schema now provide specific metadata nodes, e.g., the `numberOfTrolleys` node in the galley element definition.

In addition to the floor-based elements, several entirely new component types are made available. This includes secondary structure elements, such as sidewall panels (`sidewallPanelElements`) or ceiling panels (`ceilingPanelElements`) and luggage compartments (`luggageCompartmentElements`), but also separator walls (`classDividerElements`). The capacity for including these types of elements is essential for applications concerned with passenger experience, since they occupy large sections of the passengers field of vision.

Furthermore, in order to support the description of cargo decks in CPACS, a cargo container definition type (`cargoContainerElements`) is also included. The handling of containers is analogous to regular cabin elements.

The unspecific `genericFloorElements` node is retained, albeit with a reduced component definition without any metadata. The intention is for it to be used as a fallback solution for those cabin components, which cannot be modeled using the other available types. Conceivable examples include stair modules or special comfort modules, such as bars. That said, the current range of cabin components supported by CPACS is primarily based on the legacy definition combined with requirements for virtual cabin mock-up generation as described by Walther et al. [27]. Thus, additional

component types such as crew rest compartments or cabin attendant seats could be added in a future CPACS release, should the need arise.

3.1.2 Example: component description of a seat module

The cabin component definition nodes listed above are typical list nodes, found often throughout CPACS, which can contain an arbitrary number of child entries of their respective entry type. To illustrate, Fig. 6 provides the diagram for the `seatElementType` node type, which describes the children of the `seatElements` node.

The physical description of the components is essentially the same for all nodes, and consists of a geometry and a mass definition (`geometry` and `mass` node, respectively). The geometry of the components can be described in one of two ways: If an external geometry model is available, it can be referenced in the `genericGeometryComponent` node by providing the file path (absolute or relative to the location of the CPACS file) using the `linkToFile` node. CPACS imposes no restriction on the data format of the referenced file, allowing support for both mesh-based formats like STL or glTF and CAD exchange formats such as STEP and IGES. In principle, even the embedding of analysis models, such as FEM meshes, is imaginable. The limiting factor is the availability of suitable interfaces, which must be provided by the geometry library used.

In addition, a `transformation` node is provided. It is not meant to define the actual position of the component in the fuselage, but to prepare the external models for further processing in the context of CPACS. Coordinate transformations in CPACS can be expressed using a standard node, where they are broken down into scaling, rotation and translation. In this way, models, which have been provided, e.g., in millimeters can be re-scaled to SI units, which are customary in CPACS. Furthermore, the model can be realigned to make the subsequent placement in the fuselage more comprehensible. While this step is not required, it has proved to be a good choice, e.g., for seat modules to place the origin $\mathbf{0}$ in the middle of the forward lower edge of the bounding box:

$$\mathbf{0} \overset{!}{=} \left[x_{\min}, \frac{y_{\max} + y_{\min}}{2}, z_{\min} \right]. \tag{1}$$

If no detailed model is available, a cabin component can instead be represented by its bounding box. In this case, the `boundingBox` node is applied instead of the `genericGeometryComponent` node. The dimensions of the component are given by its diagonal vector $\mathbf{v_{box}} = [\Delta x, \Delta y, \Delta z]$. The origin of the vector and hence the placement of the box in space can be set using the `origin` node.

Aside from the geometry, each cabin component can be assigned a mass using the optional `mass` node. A standard node for mass definitions is available in CPACS, which not only supports the specification of a scalar mass, but also its position and inertia tensor. Using this data, components can be taken into account in the form of point masses, e.g., in dynamic structural analyses.

Furthermore, the diagram in Fig. 6 shows a node named `numberOfSeats`. It specifies the number of individual seats within the seat module and thus the number of passengers, which can be seated. The node is an example for type specific metadata, which differs depending on the component type. A similar example is the `numberOfTrolleys` node found in the galley definition. Due to the separate definitions, the component type definitions are open

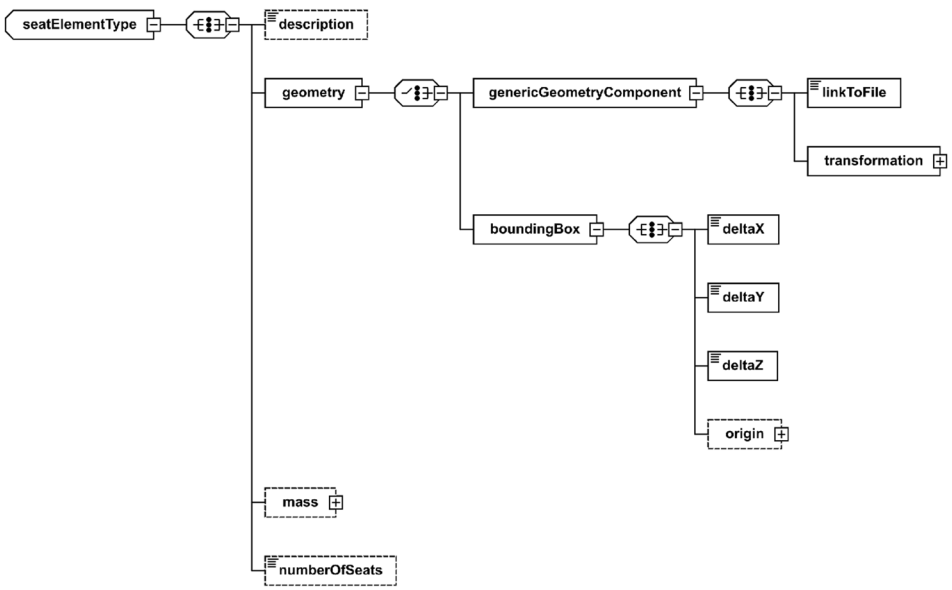


Fig. 6 XSD diagram of the component description for seats

to independent future extensions if further metadata nodes should be required.

A very important property, which is not shown in the diagram is the `uID` (unique identifier) attribute. Any `uID` can only be assigned once in CPACS and serves as a key, which can be used to reference the element in question. The schema requires the attribute `uID` to be assigned to each entry in the cabin component library. A possible `uID` for an entry in the `seatElements` node could be `seatElement0001`, even though arbitrary combinations of characters are allowed.

3.2 The cabin instance `deck`

As in the old cabin definition, the `decks` node, which is a child of the `fuselage` node, is used to create a cabin instance inside the fuselage. This node is also a CPACS list node and can, therefore, contain multiple children of type `deck`. In this way, multiple decks within a single fuselage can be represented, e.g., a passenger and a cargo deck. In the subsequent sections, the subsidiary nodes of the `deck` definition node are first grouped into different classes. Then the modeling details are portrayed for each class.

3.2.1 Classification of nodes in `deck`

The entries of which the `deck` node is composed are given in Fig. 7 and can roughly be divided into four categories:

First, top-level cabin data are given, which concerns the cabin as a whole. For instance, this includes the position of the cabin in the fuselage or a reference to a floor structure definition. Consequently, the nodes `transformation`, `parentUID`, `floorStructureUID` and `cabinGeometry` are all included in the first category, as well as nodes containing more general metadata, such as `name`, `description` and `deckType`.

The second category contains nodes, which are used to place cabin components. Since the entries from the `deckElements` node are referenced here, these nodes, which include `seatModules`, `sidewallPanels`, `cargoContainers` etc., mirror the structure of the cabin component library.

Furthermore, the nodes `aisles` and `spaces` serve to delineate areas of the fuselage, which are to remain unobstructed, such as passageways to the exits. Information on the exits themselves is stored in the `deckDoors` node, which constitutes the fourth category.

3.2.2 Top-level cabin properties

The top-level cabin properties class includes the previously mentioned `name` and `description` nodes. These nodes are mainly intended to improve the readability of

Fig. 7 XSD diagram of the `deck` node

CPACS. Whereas the mandatory node `name` should in practice contain a clear and concise label, the optional description node provides users with the possibility to share more extensive information in human-readable form. In addition, the deck must be assigned a `deckType` using the respective node. Admissible values are "`passenger`", "`VIP`", "`cargo`" or "`livestock`". Combinations of different deck types can be implemented by defining separate deck instances to represent different sections.

More specific geometric information is given in the `transformation` node. Once again, it contains a CPACS standard transformation, which this time describes the positioning of the cabin in space. In agreement with CPACS convention, the transformation is to be performed w.r.t. the coordinate system of the parent node, i.e., the fuselage, by default. However, other aircraft component can be selected as reference instead by using the `parentUID` node. As illustrated by Fig. 8, the origin of the cabin coordinate system should usually be placed at the cockpit rear wall in x direction and the floor height in z direction.

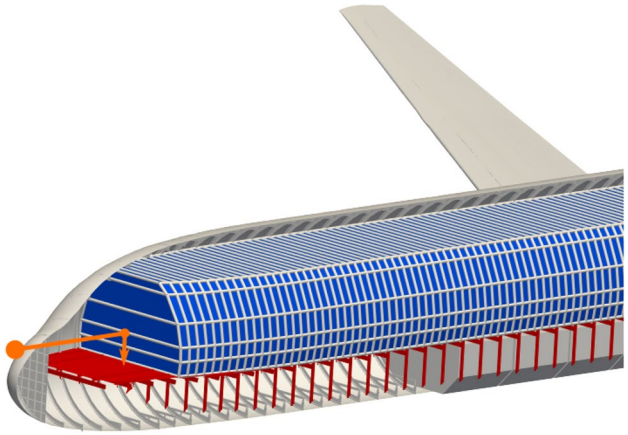
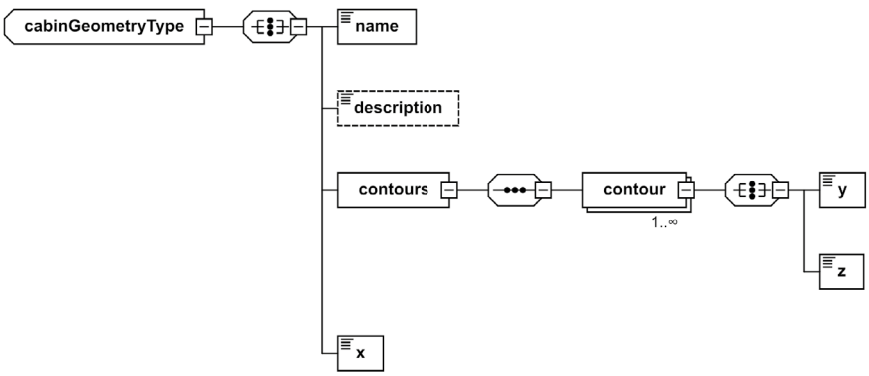


Fig. 8 Cabin geometry (blue), referenced floor structure (red) and transformation (orange)

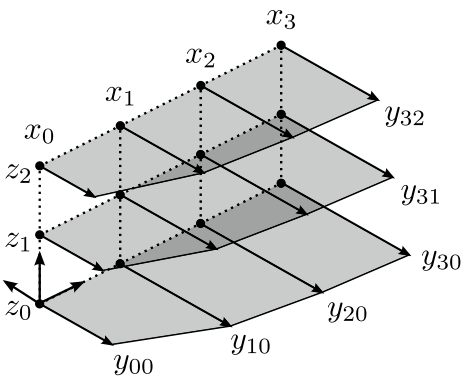
In order to improve consistency between the floor height and the structural definition, the `floorStructureUID` node was introduced. It can be used to link a floor structure definition to the cabin definition, which allows design algorithms to take this information into account.

Like all entries to follow, the outer cabin geometry, which is stored in the `cabinGeometry` node and is shown in blue in Fig. 8, is subject to the transformation. A similar node named `cabGeometry` can be found in the old cabin definition, which has, however, undergone substantial revision. The current definition is given in Fig. 9a. The node provides the boundary of the cabin, which is described using the cabin width, which is given for discrete points on a rectilinear grid in the xz plane, as illustrated by Fig. 9b. In CPACS, the x positions of the rectilinear grid are stored as a vector in the `x` node. Then each row in z direction is understood as a contour line, which is accordingly stored in a `contour` node. Since the height coordinate of the j-th contour line z_j is a constant, it is stored as a floating point number of type

Fig. 9 Description of the contour lines for the cabin geometry



(a) XSD diagram



(b) Sketch

double in the z node. The corresponding widths of the cabin y_{ij} at the positions x_i are stored as a vector for each contour line and stored in the y node.

Visibly, the outer cabin geometry definition in CPACS is decoupled from the surrounding structure. Consequently, an update will be necessary any time the fuselage geometry or structure are changed. Even though a method for determining the cabin geometry based on the structure has been presented by Walther et al. [27], this represents an obvious entry point for inconsistencies in the new cabin definition. Nonetheless, the advantages of keeping the cabinGeometry node in the schema outweigh the drawbacks, especially since it provides a simple geometric approach to the cabin and thus provides a low-barrier interface for cabin-centric design and analysis processes.

3.2.3 Component positioning

Many of the nodes found in the deck type serve the purpose of positioning the cabin components presented in Sect. 3.1.

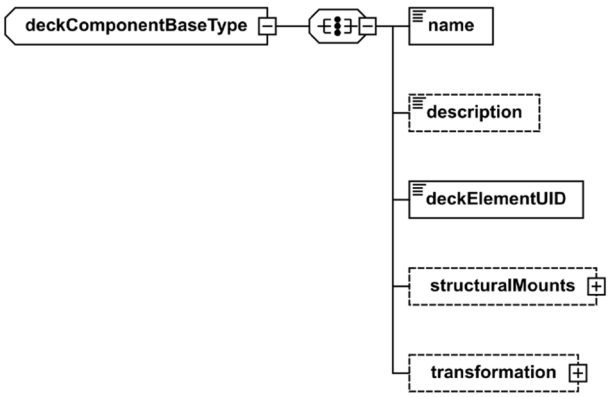


Fig. 10 Basic definition of a cabin component positioning node (XSD diagram)

The placement nodes for all component types adhere to the schema given in Fig. 10. Aside from the boilerplate entries name and description, a deckElementUID must be provided. It corresponds to the uID attribute of the component definition, e.g., seatElement0001 from Sect. 3.1.2. As per CPACS transformation rules, the component models are placed at the deck origin by default. If the component geometry was described using the genericGeometryComponent node, the transformations described therein should already have been applied at this point.

In order to arrange the components into a layout, another transformation must be applied, which is given in the transformation node of the component positioning node. At this point, it is necessary to differentiate between secondary structure and floor-based components. The secondary structure includes the nodes sidewallPanels, luggageCompartments and ceilingPanels. These elements are placed in space using the CPACS standard node for three-dimensional transformation. All other elements are placed on the floor, which means their position in height direction is fixed. This is reflected by the schema, by assigning the type transformation2D to the transformation node. This is another standard node in CPACS, which limits translation to longitudinal and transverse direction and rotation to the height axis, as illustrated by Fig. 11. Notably, the component positioning defined here and the model alignment transformation described in Sect. 3.1.2 are mutually dependent, which it why it is recommended to follow a consistent model alignment strategy, such as the one formulated in (1).

Furthermore, the schema permits the connection of structural components using the structuralMounts node. It contains a list of uIDs of structural components to which the cabin component is connected. No requirement is given w.r.t. the type of structural component, which means seat rails (in the case of seats, see Fig. 11) can be referenced as

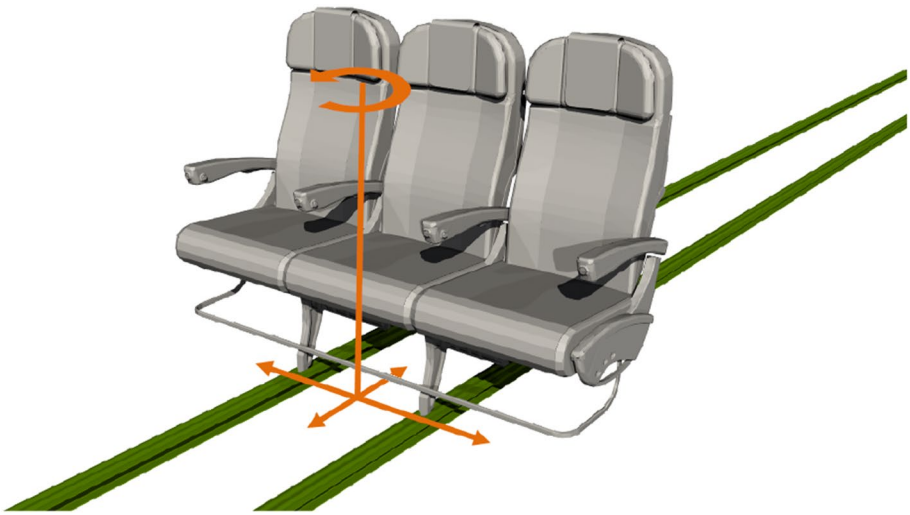


Fig. 11 2D transformation (orange) and referenced structural components (green) for an example seat

well as frames (in the case of sidewall panels). Combinations are also possible. The structural connections must be stored in the cabin instance, as opposed to the component library, since each component in the cabin instance is linked to a specific set of structural component instances, i.e., two different seats might be connected to different seat rails.

In the current version of the schema, the linking provided by the `structuralMounts` node is purely semantic and does not contain any geometric information. As such, no explicit connection points are specified. Still, the available information can be utilized, e.g., for coupling cabin component mass points to their corresponding structural elements using a distance criterion.

3.2.4 Aisles and spaces

Another class of nodes contains descriptions of unobstructed surfaces, e.g., the areas surrounding the exits. They contain important information to enable, e.g., egress simulations. Even though the free spaces can, strictly speaking, be deduced from the layout of the cabin components, it still makes sense to store them as a requirement, e.g., due to regulations or top-level design decisions, and to visualize them as part of the layout.

The free areas are defined using the nodes `spaces` and `aisles`, which have been carried over from the old cabin definition without modification. Both are based on a 2D poly line in the xy plane and thus provide the two vector nodes `x` and `y`. However it must be noted, the points for the spaces describe a polygon as shown in Fig. 12, whereas they describe the open center line for the aisle as illustrated by Fig. 13. For the free spaces the height is also given as a scalar in the `height` node and the polygon is extruded in z direction accordingly to build a 3D shape. In contrast, for the aisle, the aisle widths in y direction at each sample point are given and stored as a vector in the `width` node. Consequently, unlike the `spaces` node, the `aisles` node only describes a 2D surface.

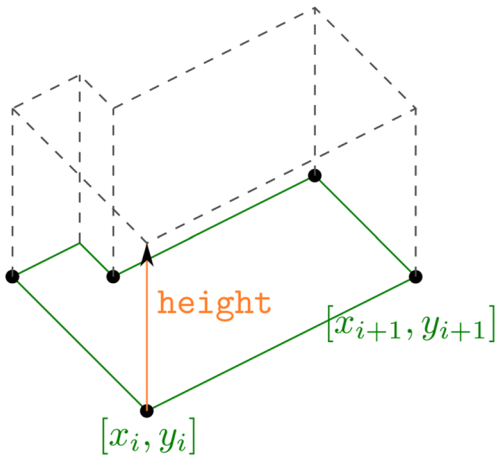


Fig. 12 Geometric definition of free spaces

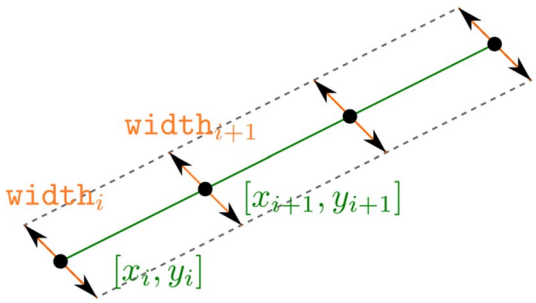


Fig. 13 Geometric definition of aisles

3.2.5 Exits

The exit layout of an aircraft plays a major role in the context of boarding, evacuation and certification and is, therefore, another important aspect of the cabin [28]. As a result, the cabin definition includes a door definition, which has been revised, as well. The corresponding diagram is shown in Fig. 14. The door definition contains important information, most importantly the number of passengers for which a given exit has been designed or certified. In addition, a type can be assigned to the door using the `doorType` node, whose value may be one of "`boarding`", "`cargo`", "`emergency`" or "`service`".

An important change with respect to the old cabin definition was made in the description of the door opening geometry. Currently, doors in CPACS can be defined in several places, e.g., depending on structural components, or using openings in the fuselage, the so-called cutouts. The old cabin definition provides even more modeling options. This redundancy creates an opening for inconsistencies. For the new cabin definition it was, therefore, decided that the cutouts

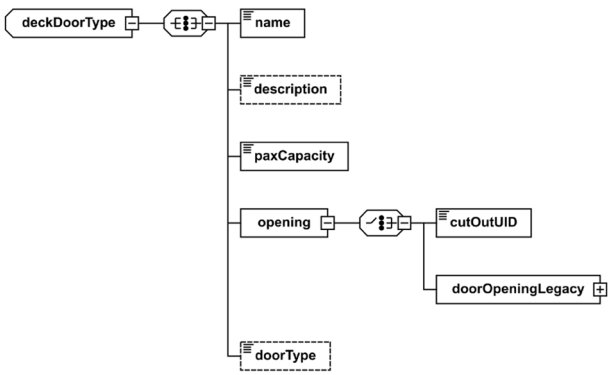


Fig. 14 XSD diagram of the door definition

should provide the basis for the description of any structural openings including the door geometry. Consequently, it is strongly recommended to use the newly introduced `cutOutUID` node to describe the door opening in the `deckDoor` node. The basic approach to the description of openings using cutouts is visualized in Fig. 15. A cutout is defined by a rounded rectangle profile with the dimensions `deltaX` and `deltaY`, which is extruded along an `orientationVector`. The local x axis for the extrusion is given by the `alignmentVector`. The center location of the profile on the fuselage surface is given by the intersection of the surface with a positioning vector, which is defined in the same way as the frame and stringer positions [16]. The cutout shape is then given by a Boolean operation between the fuselage surface and the extruded volume. While the old door definition has been retained in the `doorOpeningLegacy` node as an alternative to referencing a cutout for backwards compatibility, the employment of this node for modeling is discouraged.

4 Application examples

The versatility of the modeling options described in the preceding sections is showcased in this section using two examples. First, the design of a conventional high-density single-aisle cabin originating from Walther et al. [25] is presented.

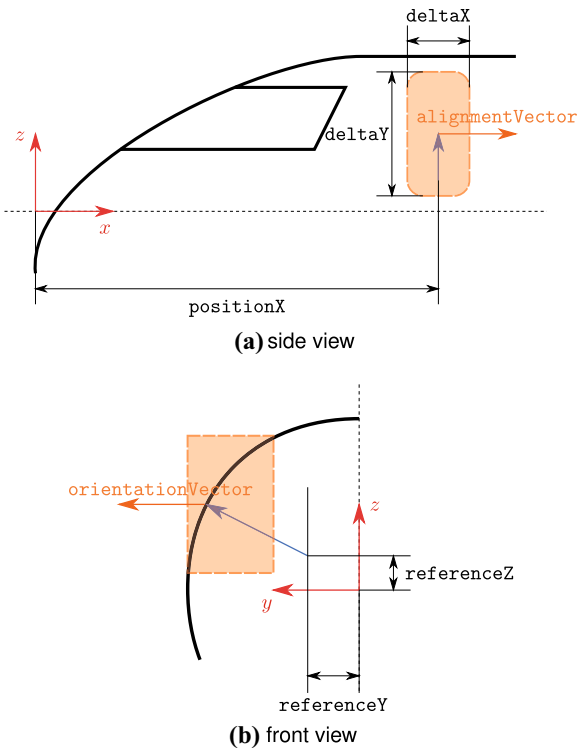


Fig. 15 Sketch of the cutout definition approach in CPACS [27]

It is then compared to a two-deck long-range design, which also contains novel cabin components.

4.1 Short-range cabin

Figure 16 shows the layout of passenger accommodation (LOPA) of the single-aisle configuration. The starting point is the so-called D150 configuration of the DLR, a replica of the Airbus A320 with a six-abreast seating arrangement. The passenger capacity is increased from 150 to 250 by stretching the cylindrical section of the fuselage. In order to reduce the fuselage length even with a high number of passengers, a very low level of comfort with a seat pitch of 28” has been selected. Furthermore, the number of galley and lavatory modules are consciously kept low at two and three, respectively. Aside from a standard triple seat module, used within

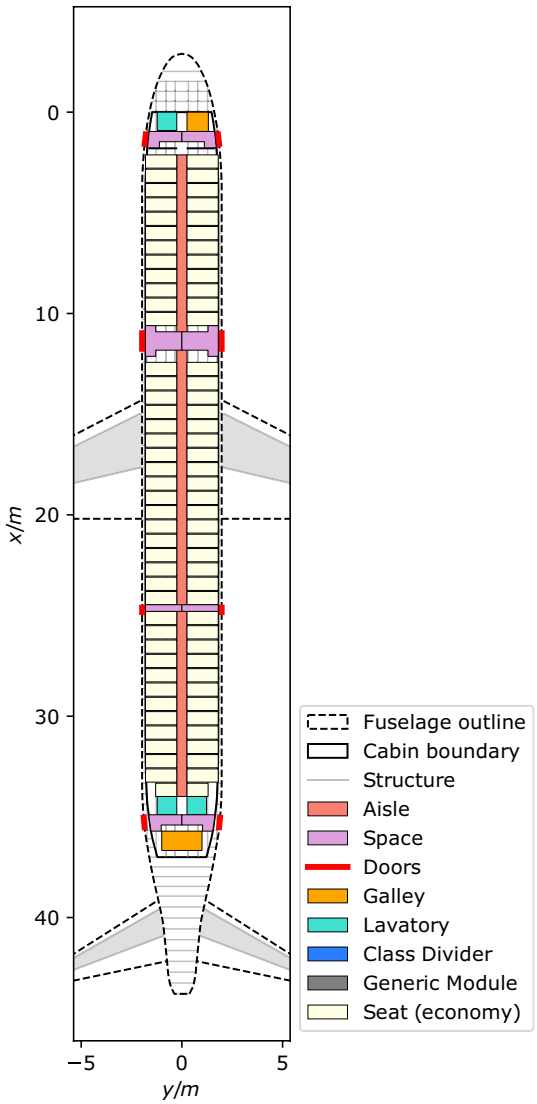


Fig. 16 LOPA of a short-range cabin (250 PAX)

the cylindrical section, the single class layout also contains a twin seat module, which is installed in the rear part of the cabin due to the contraction of the fuselage.

Four exits are provided in the cabin, corresponding to the exit types C, A, III and C from the CS-25 certification specification [28] counting from the front. They allow for the evacuation of up to 255 passengers and are marked in red on the fuselage boundary in Fig. 16. The corresponding passageways and assist spaces have been modeled using the `spaces` node in CPACS. Visibly, they are aligned with the door openings and do not interfere with any of the cabin components. This also applies to the aisle, for which a width of 20" has been assumed.

Figure 17 shows the three-dimensional model of the cabin plotted in Paraview [29], which has been generated using the fuselage and cabin geometry library by Walther et al. [25]. Whereas the galley and lavatory modules are represented by boxes, detailed geometry models are deployed for the seats. The secondary structure elements, i.e., sidewall panels, luggage compartments and ceiling panels, are represented in this way, as well. The doors are modeled as cutouts in the structural model. Furthermore, the cutaway visualization provides a view of the cargo deck including containers.

Walther et al. [25] show that models, such as the one shown in Fig. 17, can be ported to video game engines such as Unity [30], and provide the basis for interactive human factors analyses using virtual reality as discussed, e.g., by Fuchs et al. [6]. This shows, that detailed analysis models for human factors can be derived based on the proposed definition, in keeping with the first objective stated in Sect. 1.1.

4.2 Long-range cabin

The cabin arrangement for the two decks of the long-range configuration is given in the form of LOPA plots in Fig. 18. The upper deck and the main deck are shown, which can be represented together in a single CPACS data set. The design combines elements of the existing Airbus A380 cabin layout with some novel features.

A three class layout for 554 passengers has been adopted, which places the majority of the first and business class seats on the upper deck, while most economy seats are found on the main deck. Even from the simplified view, it can be perceived, that several unconventional cabin components have been installed in the economy class. Compartment modules are placed in the forward part of the economy section, which are represented by large blocks and provide a spatially separated area for up to four passengers. Behind them, a series of staggered single seats installed at an angle of 45° is found in the center block of the cabin, promising increased legroom and improved privacy. This arrangement is a showcase of the capacity for rotating seats using the transformation node in CPACS (s. Sect. 3.2.3).

The exits are once more accessible via passageways. For configurations with more than one aisle, it is furthermore required to provide a passageway between the aisles. Depending on the exit type, an overlap or neighborhood criterion w.r.t. the outer passageways must be fulfilled [28].

A further particularity is the modeling of stair elements, which are not provisioned by the CPACS schema and are, therefore, represented by `genericFloorElement` entries.

Figure 19 shows a cutaway view of the 3D cabin model. The same modeling and visualization pipeline as in Fig. 17 is used. A total of seven different seat models can be discerned. In addition to the conventional seats also used in the conventional single-aisle configuration in the previous section, the economy class contains the aforementioned staggered seats and two types of compartment modules (outer blocks and center block). In the business class, different seat models are used for the upper and lower deck. Finally, a first class module is also used.

In contrast to the short-range cabin, no paneling elements are shown. This is due to the designs being in different stages of development. The flexibility of provisioning descriptions for components in order to provide a means for the storage as new knowledge becomes available without requiring it upfront in the sense of a multi-fidelity paradigm is one of the major strengths of CPACS [31].

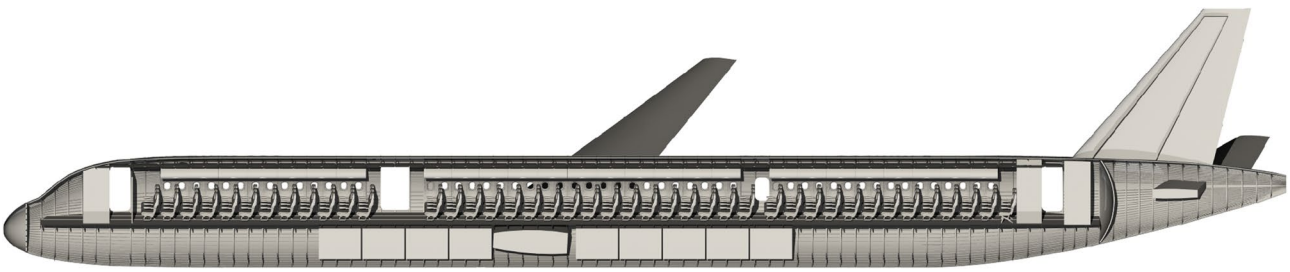


Fig. 17 Cut view of the short-range cabin modeled in 3D

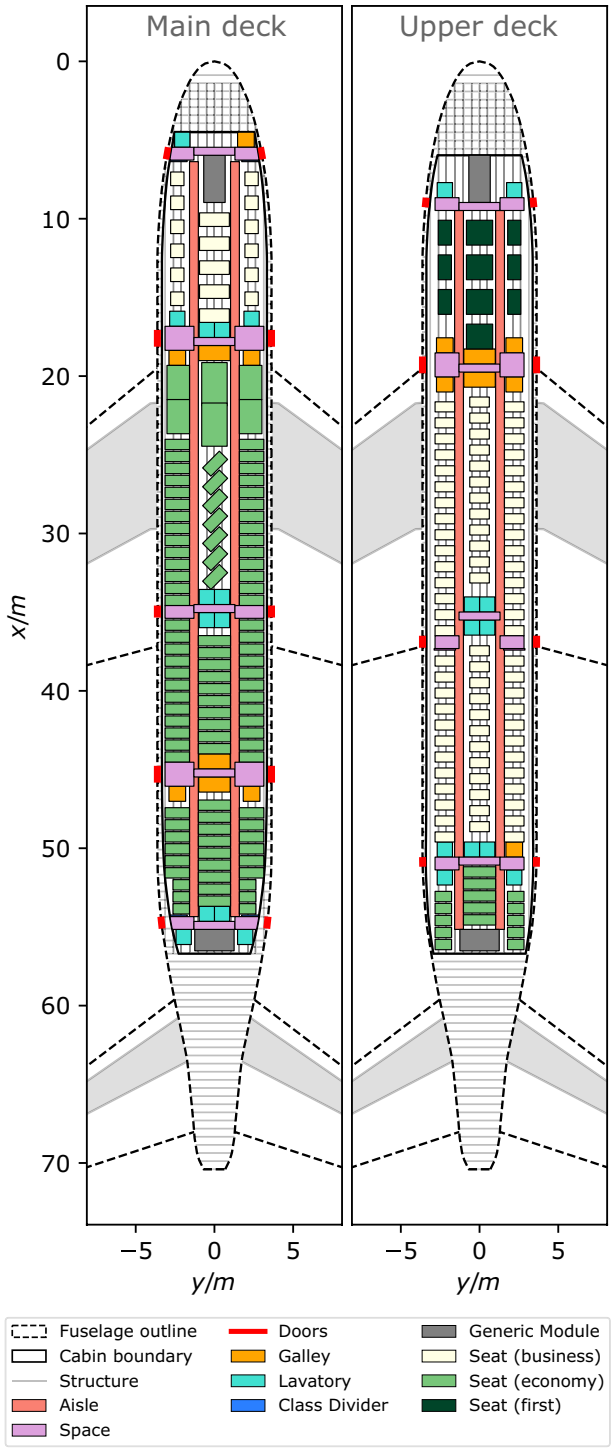


Fig. 18 LOPAs of the main and upper deck of the long-range configuration (554 PAX)

5 Conclusion

In this paper, a revised definition for cabins in the CPACS aircraft data schema is introduced. The objective of the development is to enable the generation of relevant analysis models for human factors and structural analysis taking into account the cabin directly from CPACS.

To this end, the `deckElements` node has been separated from the preexisting `decks` node, to store the definitions of the cabin components. In addition to the previously established cabin component types, a number of new component types including sidewall panels and luggage compartments are now supported. Furthermore, the component definitions allow for the incorporation of external geometry models, which can significantly improve the level of detail, e.g., for the derivation of virtual reality models. Other properties, such as mass and center of gravity can be provided as well. Each component is defined only once in the `deckElements` node and subsequently referenced, resulting in significantly leaner data sets.

The `decks` node is retained to hold the description of the instance of a cabin within a fuselage. To this end, the predefined cabin components can be referenced multiple times and individually positioned. Semantic linking of individual cabin components to structural components such as frames or seat rails and of the cabin as a whole to a floor structure is possible using the `structuralMounts` and `floorStructureUID` entries, respectively. That said, an application, where the connections are exploited, e.g., in a finite-element analysis, is yet to be shown.

The accessibility of the cabin definition for new users has been improved by applying previously established modeling practices for CPACS as much as possible. For instance, `transformation` nodes are used for positioning, which are found in many definitions in CPACS.

Moreover, the versatility of the revised cabin definition is showcased by two examples. A short-range layout demonstrates, that conventional layouts can be represented to a very high level of fidelity. Aside from the seating layout, secondary structure elements such as sidewall panels and luggage compartments are also present. Furthermore, adherence to regulatory requirements can be verified by modeling free spaces. It is shown, that a suitable for human factors analysis using virtual reality can be derived. The long-range double-deck configuration, on the other hand, illustrates the substantial modeling freedom offered by the schema. In addition to multi-deck support, seven different seat models in three classed are referenced, including some unconventional designs.

The new cabin definition has been formalized in an XML Schema Definition and is included in CPACS as of version 3.4. However, this is not the end of development. For example, a precise description of structural connection points of cabin components would be desirable. This would on the one hand enable a more accurate representation of the connections in FEM models, on the other hand the additional knowledge could also be used to refine component placement in cabin layout generation processes. Finally, it is the

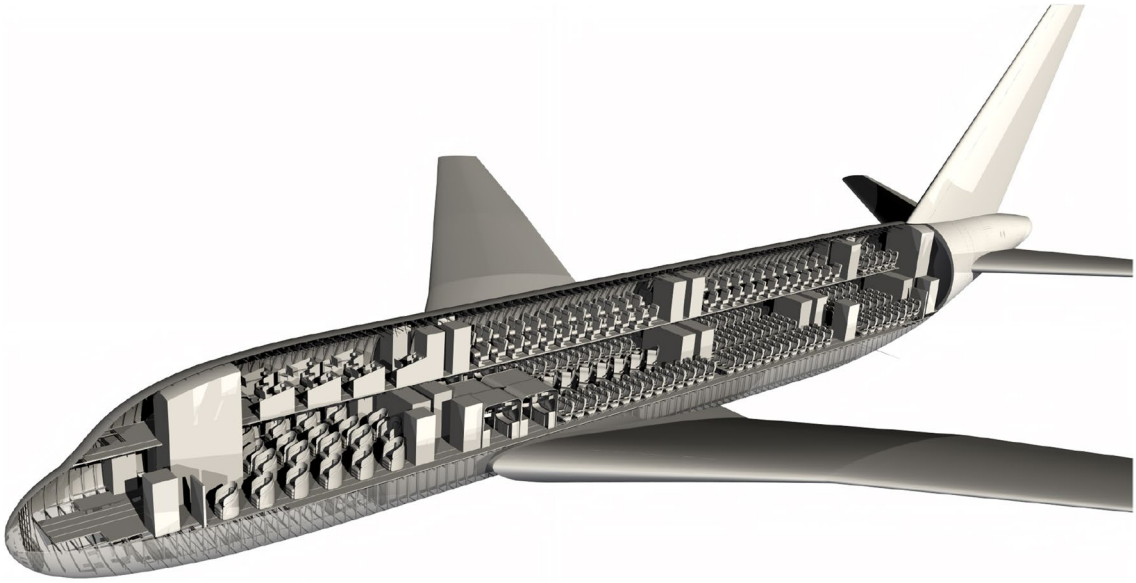


Fig. 19 3D model of the double-deck long-range cabin

hope of the authors, that the new definition, along with a growing spectrum of connected analysis capabilities, will help to establish the topic of cabin modeling and analysis as an inherent part of the virtual aircraft design process.

Acknowledgements The authors would like to thank all participants of the CPACS Stakeholder Workshop on 31 May 2021 for their helpful feedback to the cabin definition presented. Further thanks go to Thomas-Mathias Bock and Alex Gindorf for sharing the component models shown, as well as Daniel Silberhorn, who provided the CPACS data set of the long-range configuration.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Alder, M., Moerland, E., Jepsen, J., Nagel, B.: Recent Advances in Establishing a Common Language For Aircraft Design With CPACS. In: Aerospace Europe Conference - AEC2020, Bordeaux, France (2020)
2. DLR Institute for System Architectures in Aeronautics. CPACS Website. (2022). http://cpacs.de
3. Fuchte, J., Gollnick, V., Nagel, B.: Integrated Tool for Cabin and Fuselage Modeling in Future Aircraft Research. In: Workshop on Aircraft System Technology (AST). Apr. (2013). https://elib.dlr.de/82767/
4. Crescenzio, F. D., Bagassi, S., Asfaux, S., Lawson, N.: Human centred design and evaluation of cabin interiors for business jet aircraft in virtual reality. In: International Journal on Interactive Design and Manufacturing (IJIDeM) 13.2 (Apr. 2019), pp. 761–772. https://doi.org/10.1007/s12008-019-00565-8.
5. Engelmann, M., Drust, D., Hornung, M.: Automated 3D cabin generation with PAXelerate and Blender using the CPACS file format. en. In: Deutscher Luft- und Raumfahrtkongress 2020. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., (2020). https://doi.org/10.25967/530014.
6. Fuchs, M., Beckert, F., Biedermann, J., Nagel, B.: A collaborative knowledge-based method for the interactive development of cabin systems in virtual reality. In: Computers in Industry 136 (Apr. 2022), p. 103590. https://doi.org/10.1016/j.compind.2021.103590.
7. Schwinn, D. B.: Parametrised fuselage modelling to evaluate aircraft crash behaviour in early design stages. In: International Journal of Crashworthiness 20.5 (Mar. 2015), pp. 413-430. https://doi.org/10.1080/13588265.2015.1022435.
8. Siemann, M. H., Schwinn, D. B., Scherer, J., Kohlgrüber, D.: Advances in numerical ditching simulation of flexible aircraft models. In: International Journal of Crashworthiness 23.2 (Aug. 2017), pp. 236-251. https://doi.org/10.1080/13588265.2017.1359462.
9. Hesse, C., Allebrodt, P., Walther, J.-N.: Integration of multi-physics analysis into the cabin design process using virtual reality. In: AIAA AVIATION 2021 FORUM. American Institute of Aeronautics and Astronautics, July (2021). https://doi.org/10.2514/6.2021-2776.
10. Kroll, N., Abu-Zurayk, M., Dimitrov, D., Franz, T., Führer, T., Gerhold, T., Görtz, S., Heinrich, R., Ilic, C., Jepsen, J., Jägersküpper, J., Kruse, M., Krumbein, A., Langer, S., Liu, D., Liepelt, R., Reimer, L., Ritter, M., Schwöppe, A., Scherer, J., Spiering, F., Thormann, R., Togiti, V., Vollmer, D., Wendisch, J.-H.:

DLR project Digital-X: towards virtual aircraft design and flight testing based on high-fidelity methods. In: CEAS Aeronautical Journal 7.1 (Dec. 2015), pp. 3-27. https://doi.org/10.1007/s13272-015-0179-7.
11. Pfeiffer, T., Moerland, E., Freund, S., Hasan, Y. J., Bertsch, L., Flink, J.: Ergebnisse des Flugzeugvorentwurfprojekts FrEACs (Future Enhanced Aircraft Configurations). In: Deutscher Luft- und Raumfahrtkongress. (2017)
12. Görtz, S., Krumbein, A., Ritter, M., Hofmann, J.: DLR-Projekt VicToria - Virtual Aircraft Technology Integration Platform. In: Deutscher Luft - und Raumfahrtkongress 2018. (2018). https://elib.dlr.de/121695/
13. Moerland, E., Deinert, S., Daoud, F., Dornwald, J., Nagel, B.: Collaborative aircraft design using an integrated and distributed multidisciplinary product development process. In: ICAS 2016. Sept. (2016). https://elib.dlr.de/111005/
14. Ciampa, P. D., Nagel, B.: AGILE Paradigm: The next generation collaborative MDO for the development of aeronautical systems. In: Progress in Aerospace Sciences 119 (Nov. 2020), p. 100643. https://doi.org/10.1016/j.paerosci.2020.100643.
15. Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F.: Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C. Nov. (2008). http://www.w3.org/TR/xml/
16. Scherer, J., Kohlgrüber, D.: Fuselage structures within the CPACS data format. In: Aircraft Engineering and Aerospace Technology 88.2 (Mar. 2016), pp. 294-302. https://doi.org/10.1108/aeat-02-2015-0056.
17. Charbonnier, D., Vos, J., Shiva Prakasha, P., Mirzoyan, A., Savelyev, A., Della Vecchia, P.: Low Speed Take-Off Aerodynamic Analysis. In: 6TH CEAS Air & Space Conference Aerospace Europe 2017. Oct. (2017)
18. Fröhler, B., Hesse, C., Atanasov, G., Wassink, P.: Disciplinary Sub-Processes to Assess Low-Speed Performance and Noise Characteristics within an Aircraft Design Environment. In: Deutscher Luft- und Raumfahrtkongress 2020. Sept. (2020). https://elib.dlr.de/140229/.
19. Wöhler, S., Atanasov, G., Silberhorn, D., Fröhler, B., Zill, T.: Preliminary Aircraft Design within a Multidisciplinary and Multifidelity Design Environment. In: Aerospace Europe Conference 2020. Apr. (2020). https://elib.dlr.de/135245/
20. Walther, J.-N., Ciampa, P. D., Nagel, B.: Disciplinary Implications of a System Architecting Approach to Collaborative Aircraft Design. In: 14th WCCM-ECCOMAS Congress. CIMNE, (2021). https://doi.org/10.23967/wccm-eccomas.2020.122.
21. Siggel, M., Kleinert, J., Stollenwerk, T., Maierl, R.: TiGL: An Open Source Computational Geometry Library for Parametric Aircraft Design. In: Mathematics in Computer Science 13.3 (July 2019), pp. 367-389. https://doi.org/10.1007/s11786-019-00401-y.
22. Siggel, M., Stollenwerk, T., Kleinert, J., Gruber, B.M., Maierl, R.: TiGL - The TiGL Geometry. Library (2021). https://doi.org/10.5281/ZENODO.858650
23. Nicolosi, F., Marco, A. D., Attanasio, L., Vecchia, P. D.: Development of a Java-Based Framework for Aircraft Preliminary Design and Optimization. In: Journal of Aerospace Information Systems 13.6 (June 2016), pp. 234–242. https://doi.org/10.2514/1.i010404.
24. De Marco, A.: JPAD, A Java Application Programming Interface for Aircraft Design. en. (2018). https://doi.org/10.5281/ZENODO.2064963.
25. Walther, J.-N., Hesse, C., Biedermann, J., Nagel, B.: High Fidelity Digital Cabin Mock-Up based on Preliminary Aircraft Design Data for Virtual Reality Applications and Beyond. In: AIAA AVIATION 2021 FORUM. American Institute of Aeronautics and Astronautics, July (2021). https://doi.org/10.2514/6.2021-2775.
26. Fallside, D. C., Walmsley, P.: XML Schema Part 0: Primer Second Edition. W3C. Oct. (2004). https://www.w3.org/TR/xmlschema-0/
27. Walther, J.-N., Kocacan, B., Hesse, C., Gindorf, A., Nagel, B.: Automatic cabin virtualization based on preliminary aircraft design data. In: CEAS Aeronautical Journal (Jan. 2022). https://doi.org/10.1007/s13272-021-00568-w.
28. European Aviation Safety Agency (EASA). CS-25: Certification Specifications for Large Aeroplanes. (2008)
29. Ayachit, U.: The ParaView Guide: A Parallel Visualization Application. Kitware, Clifton Park, NY (2015). ISBN: 9781930934290
30. Unity Technologies. Unity Real-Time Development Platform | 3D, 2D VR & AR Engine. June (2022). https://unity.com/.
31. Böhnke, D., Nagel, B., Gollnick, V.: An approach to multi-fidelity in conceptual aircraft design in distributed design environments. In,: Aerospace Conference. IEEE, Mar. **2011**,(2011). https://doi.org/10.1109/aero.2011.5747542