RECENT ADVANCES IN ESTABLISHING A COMMON LANGUAGE FOR AIRCRAFT DESIGN WITH CPACS

Marko Alder[(1)] and Erwin Moerland[(2)] and Jonas Jepsen[(3)] and Björn Nagel[(4)]
[(1),(2),(3),(4)]*Institute of System Architectures in Aeronautics, German Aerospace Center (DLR)*
Hein-Saß-Weg 22, 21129 Hamburg (Germany)
[(1)]*marko.alder@dlr.de,* [(2)]*erwin.moerland@dlr.de,* [(3)]*jonas.jepsen@dlr.de,* [(4)]*bjoern.nagel@dlr.de*

ABSTRACT

Today's aircraft design projects are characterized by interdisciplinary collaboration between a large amount of heterogeneous disciplines. Each discipline contributes with specialized knowledge and software tools which are integrated in automated simulation workflows. The utilization of a common data model significantly reduces the number of possible interconnections between the modules and ensures a consistent source of information. This paper presents the Common Parametric Aircraft Configuration Schema (CPACS) and discusses the progress of establishing it as a common language for aircraft, spacecraft and rotorcraft design. With increased flexibility of the aerodynamic performance maps, a reformulation of the wing component segments for defining the wing internal structure, a new definition for engine nacelles and pylons and the possibility to include individual tool-specific schemas, the current enhancements and the corresponding development process of the latest CPACS release are presented. The paper concludes by showing that publishing CPACS and its corresponding libraries under an open-source license has been the enabler for a wide adoption of the data format in the aeronautical research and industry.

1. INTRODUCTION

An increasing number of aircraft design projects apply Multidisciplinary Design Optimization (MDO) techniques in decentralized and heterogeneous teams [7]. Aiming at advancing the interdisciplinary collaboration between the various disciplines and realizing decentralized MDO architectures, about a decade ago researchers have postulated that the process efficiency within such a collaborative environment can be enhanced by an open, central data model that serves as a common language [22]. This hypothesis has been proven by applying the Common Parametric Aircraft Configuration Schema (CPACS) [22] in advanced preliminary aircraft design tasks including MDO.

Motivated by these results, subsequent aircraft design projects, such as IDEaliSM [20, 25] or AGILE [7], have driven further development of CPACS and successfully applied the data model within international collaboration environments involving partners from various universities, research establishments and industries. This paper presents the results of this development and addresses the question whether the use of a common language has indeed linked the aircraft design process more closely.

2. DATA MODELING IN AIRCRAFT DESIGN

In this chapter, the challenges on data modeling within state-of-the-art aircraft design processes are discussed. Existing data models aiming to cope with these challenges cover a wide range of aviation disciplines and are all based on standardized data modeling languages, such as the Extensible Markup Language (XML), the Unified Modeling Language (UML) and its derivatives or the Web Ontology Language (OWL). It is not the intention of this paper to provide a thorough overview of available modeling languages and their properties, rather to present the specific implementation of CPACS using XML. For an overview of the properties of modeling languages other than XML, the reader is kindly referred to the extensively available documentation on this topic.

2.1 Challenges on data modeling in collaborative design

As indicated in Sec. 1, today's aircraft design projects must account for the interaction of various disciplines such as structural mechanics, aerodynamics or flight mechanics. One possibility to consider this interaction is to integrate the required sub-disciplines in a monolithic software architecture used to synthesize an aircraft on conceptual and design level by applying sequential iteration methods [32]. From a developer's point of view, the internal data exchange between analysis modules within the monolithic system is advantageous concerning the easiness of resolving data and model inconsistencies. Due to the increased complexity of the design considerations already in early aircraft design stages nowadays however, the process cannot be handled by a single person anymore. Furthermore, if the developer of the monolithic system is not available anymore, a large amount of implicit knowledge required to correctly operate the system is lost. Therefore, today's research on collaborative MDO is often based on tool integration frameworks

which allow to integrate analysis modules in decentralized workflows [7] enabling engineering departments at different sites to be involved in the design process. The open-source software RCE [4, 5] (Remote Component Environment) is an example of such a process integration framework. It enables the connection of analysis modules via a server-client based network infrastructure. Upon workflow execution, the execution of individual modules hosted on their respective server instances is triggered when required and the required data is automatically exchanged. In this construct, only input and output data is exchanged while the tool itself remains under control of the tool owner. A challenge arising within this approach is that the stakeholders might use different data models and vocabulary resulting in $N(N-1)$ possible directions of data exchange between N tools. Within such a simulation process, the consistency among the multiple disciplinary models and different levels of details of the simulations needs to be guaranteed. One solution to this challenge is obtained by introducing a central data exchange format based on common semantics for the whole system to be designed (e.g., full parametrization of an aircraft) which is easy to read and interpret by human. As depicted in Fig. 1, by using this single source of truth the amount of connections reduces to $2N$. This is the main motivation for the development of common aviation data models, from which a selection of the most important ones is presented in the following section.

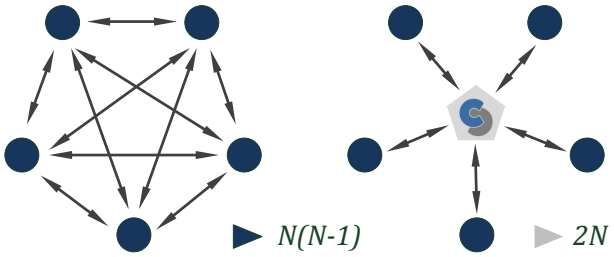

Figure 1: The possible amount of data interfaces reduces from $N(N-1)$ to $2N$ by using a central data format

2.2 Overview of existing aircraft and aviation data models

DAVE-ML is a flight dynamic model exchange format published by a standards working group of the American Institute of Aeronautics and Astronautics (AIAA) [1]. Inspired by the input-files used for the flight dynamics model software library JSBSim [3], DAVE-ML is an XML based markup language. Both DAVE-ML and JSBSim have a strong focus on mathematical modeling, which is achieved by defining individual mathematical operators (JSBSim) or by implementing MathML (DAVE-ML). A higher level of multidisciplinarity is covered by the Aircraft Design Markup Language (ADML) [8]. Developed at the Virginia Polytechnic Institute & State University ADML is as well based on XML and rather represents generic systems by describing its components on a detailed level using abstract mathematical models (this bottom-up approach is explained in Sec. 3.2). In this way ADML primarily aims at modeling unconventional aircraft [8].

Various standards exist for data exchange models covering aircraft operations. Keller [16] summarizes the most widely-used standards, for example the Flight Information Exchange Model (FIXM), the Aeronautical Information Exchange Model (AIXM), the Weather Information Exchange Model (WXXM) or the Maintenance Management Exchange Model (MMIXM) [16]. All of these based on UML.

First attempts to model aviation data using OWL are investigated by Ast et al. [2] focusing on the design of an aircraft system and Keller [16] with respect to general aviation data. However, both are still on a research level and did not yet establish as a widely-used common standard.

3. AN INTRODUCTION TO CPACS

To adequately meet the challenges of collaborative design projects listed in Sec. 2.1, the data model CPACS has been introduced and developed at the German Aerospace Center (DLR) since 2005. The strengths of XML as modeling language will be explained in more detail. On this basis, the modeling approach is presented with regard to a general representation of an aircraft ontology and the corresponding implementation in XML. Finally, the associated software libraries are briefly outlined.

3.1 The strengths of XML as modeling language

XML is an open standard, which is officially coordinated and documented by the Wold Wide Web Consortium (W3C) and nowadays globally accepted in the field of information technology [28]. In contrast to other markup languages, such as HTML with a fixed list of tags, XML has a very generic character and can therefore serve as a computer-processable meta-language [28] enabling the development of an aviation ontology as a markup language itself. Another strength of XML is the separation of the data structure from the actual content. This allows for the definition of complex structural and semantic rules in a separate XML Schema Definition (XSD) file, while users can independently describe data using an exchange format that is easy to read by just using a text editor. In this context Böhnke et al. [6] compare XML with other modeling languages, such as the STandard for the Exchange of Product model data (STEP) or UML, and point

out the importance of the extensibility and simplicity of XML in collaborative design environments.

Relying on an open standard furthermore offers the advantage of choosing from a wide range of more advanced editors and APIs, which are often made available by a large open-source community. Another advantage is that XML is increasingly used to model process information and application configurations [28]. With regard to CPACS, this allows to consistently combine an aviation-specific ontology with tool-specific and process-specific data. Finally, the XML language family comprises a number of additional standards, such as XSL [30], XPath [31] or XLink [29] which can be used to further extend the capabilities of the markup language.

3.2 The applied data modeling approach

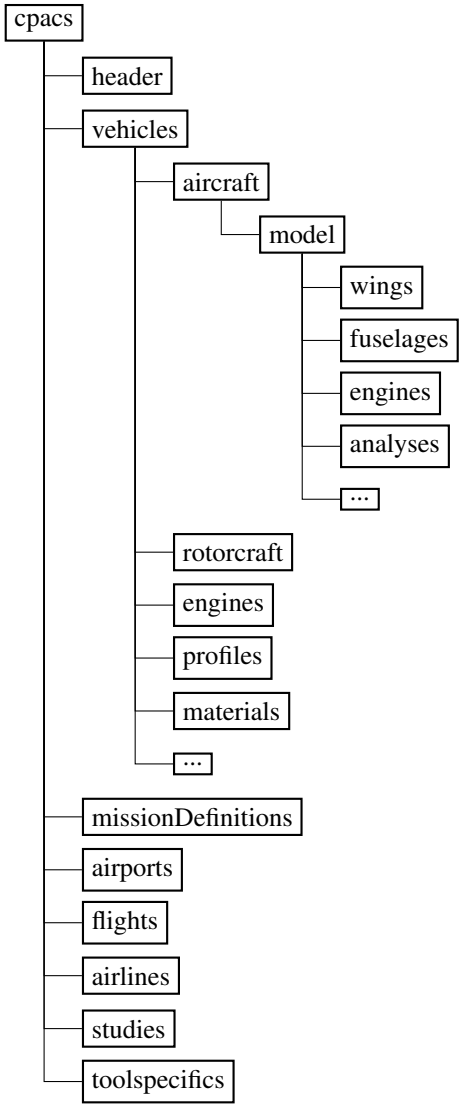

Figure 2: Extract of the hierarchical structure of CPACS

Making use of the hierarchical representation of data in XML the structure of CPACS mainly follows a top-down approach which decomposes a generic concept (e.g., an aircraft) into a more detailed description of its components (see Fig. 2). This originates from the conceptual and preliminary design of aircraft, where the level of detail is initially low and continues to increase as the design process progresses. The hierarchical structure furthermore promotes the simplicity of the exchange format which is required in collaborative design environments (Sec. 2.1) so that the various stakeholders can easily append their results. Fig. 2 shows that CPACS offers a similar approach for other areal vehicles such as helicopters and also for information on aircraft operations such as on airports and airlines, missions and flights as well as process-specific data used for studies and analysis modules. A deeper insight into this structure is given by Liersch and Hepperle [19] and is extensively described within the CPACS documentation [10]. The geometric interpretation of the wing parameterization is furthermore explained in Sec. 3.4 to serve as illustrative example.

For some concepts within CPACS, however, a bottom-up approach is applied where the components are first defined in detail and then linked to each other to formulate a higher-level concept. This is similar to the principles applied within ADML (see Sec. 2.2) and is advantageous when used multiple times within complex systems, such as engines, which only have to be defined once in order to be referenced several times on the aircraft. The combination of these two methodologies is known as middle-out approach and enables the goal to fully parametrize aeronautical systems. In the development of CPACS enhancements the challenge is always to find an appropriate compromise between the very flexible bottom-down approach and the rigid but semantically more meaningful top-down approach.

3.3 XML implementation

To implement the hierarchical structure presented in Sec. 3.2, the majority of CPACS definitions is based on XML standards specified by W3C and implemented as schema in XSD. Simple data types (string, double, boolean, ...) as well as complex data types (containing pre-defined child-elements) are extended by attributes to model recurring meta-data, such as parameters for uncertainty quantification or pointers to external data. These types are introduced as CPACS base-types. Each element in CPACS first inherits the properties from these base-types and extends it with information-specific child-elements and attributes or restricts the allowed content by introducing enumerations of simple values.

Additional base-types provide the means to represent vectors and arrays in a more compact form than specified by the official W3C standard. In aircraft design, large coefficient-based data sets might occur - especially in the

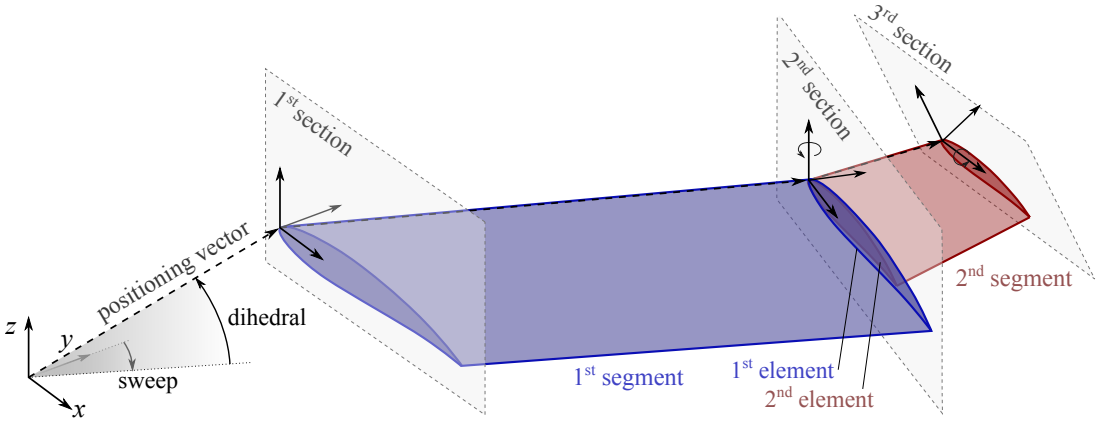

Figure 3: Geometric representation of a wing

field of aerodynamic analyses (e.g. aerodynamic coefficients and their derivatives depending on Mach number, altitude, angle of attack and angle of yaw) and engine performance (e.g. thrust, fuel usage and emission values as function of Mach number, altitude, engine rating, engine condition, shaft speed and power offtake). Within the official standard, vectors are defined as a sequence of elements and multidimensional arrays as a nested list of element sequences. When using this standard to save the coefficient-based data, most of the XML content would consist of tags (XML nodes written in angle brackets). This would be at the expense of clarity and introduce an unnecessary large file size which would slow down the network-based simulation approach. For this reason, CPACS-specific vector and array types are introduced, inheriting from the string type and allowing semicolon-separated vector representations within single tags.

The definitions and conventions used in CPACS are provided by the corresponding documentation written inside the XSD file. This forces consistency between the definitions and its documentation, since these are provided at the same location within the schema. From the documentation entries available within the schema, a HTML-based documentation is derived using an open-source plug-in for the Sandcastle Help File Builder [18]. Using the HTML-based documentation, the user can search through the CPACS hierarchy and obtain the respective information on each tag within the data format.

3.4 Example: parametrization of a wing

An example shall illustrate the geometric representation of aircraft geometries in CPACS. For this purpose, the parametrization of a wing is demonstrated as used for both aircraft and helicopters.

A wing consists of at least two sections, where each section represents a local coordinate system in Cartesian coordinates. Fig. 3 illustrates a wing composed of three sections. CPACS offers two approaches to specify the position of a section in space: (1) by using a positioning vector defined by a length, sweep angle and dihedral angle as well as (2) a transformation through rotation around all three axes and a three-dimensional translation and scaling. As shown in Fig. 3 both approaches can be superimposed. While the first section is only defined by a positioning vector (i.e., the axes are still parallel to each other) the second section is additionally rotated around the z-axis and the third section, representing the wing tip, is rotated around the x-axis.

Each section contains one or more elements which are used to specify the location of two-dimensional airfoils. The elements are again defined as local coordinate system within a section which can be transformed through rotation, translation and scaling. Two elements can be placed in the same section to model a discontinuous jump in span-wise direction as shown in Fig. 3 and explained in more detail by Siggel et al. [24].

Finally, a wing segment is defined as volumetric extrusion connecting two elements of adjacent sections. The optional specification of guide curves furthermore allows to realize curved leading and trailing edges.

This example underlines the potential of XSD to specify repeating properties as complex types that can be used for multiple elements (see Sec. 3.3). It allows in this case to use the same type of wing definition for both aircraft and helicopters or the same type of transformation for the parent coordinate system of the wing, its sections and elements. Furthermore does the use of unique XML identifiers (*uIDs*) allow airfoils from the profile database to be repeatedly referenced (see <*profiles*> node in Fig. 2).

3.5 Program interfaces and libraries

To support the integration of tools in collaborative design environments (see Sec. 2.1) the XML interface library TiXI [9] has been developed at DLR. TiXI is based on Libxml2 [27] and is extended by methods to understand CPACS specific conventions, i.e. reading and writing the

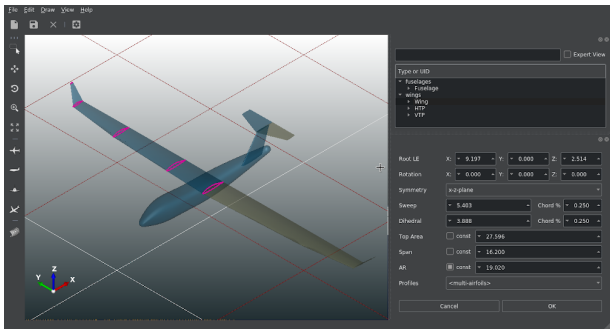

Figure 4: CPACS-Creator GUI showing a CPACS model next to an input deck to adjust the corresponding parameters.

vectors and arrays introduced in Sec. 3.3. The library can be used from applications written in C, C++, Fortran, Java, Matlab and Python [9].

Furthermore, both a Python library generated from the XSD file via generateDS [17] as well as a Java library are delivered with each CPACS release. These libraries provide classes which support the in-memory handling of CPACS data sets. In this context, the hierarchical structure of CPACS proves to be advantageous when used within object-oriented programming languages.

The geometry library TiGL [24] translates the parametric description of aircraft and helicopters to three-dimensional shapes, which various disciplines require especially in the preliminary and higher-fidelity design phases. For this, the open-source CAD kernel OpenCASCADE [23] is used and extended by various geometric modeling features, for example a detailed surface modeling generator using curve network interpolation on the basis of Gordon [13]. This increases the quality (smoothness) of surfaces to a level which is required for high fidelity analysis, such as Computational Fluid Dynamics. The TiGL aircraft component module can create both external aircraft components - such as wings, flaps, fuselages, engines - and internal structures - such as ribs, spars and cutouts [24]. TiGL is written in C++ and provides interfaces for C, C++, Python, Java and MATLAB. In addition, the TiGL Viewer visualizes CPACS data sets in a graphical user interface (GUI) based on 3D OpenGL and allows visualization tasks to be automated by scripting in a console.

Next to the TiGL geometry viewer, a CPACS-Creator has been released. Developed by CFS Engineering, this editor is based on TiGL and allows the parametric modification of aircraft geometries as well as its creation from scratch. CPACS-Creator is published as open-source software as well and its functions can be directly called by analysis modules by using the high-level functions of the corresponding C++ library. A GUI directly integrates in the TiGL Viewer and allows for the synthesis and modification of CPACS data (see Fig. 4).

4. RECENT ENHANCEMENTS OF THE DATA FORMAT AND EXPERIENCES IN ITS UTILIZATION WITHIN COLLABORATIVE DEVELOPMENT PROCESSES

4.1 Major enhancements introduced in CPACS 3

The third major-release version of CPACS was published in July 2018. Some of the experiences gained from previous projects were used to improve and add new elements and types. The most important enhancements introduced in CPACS 3 (including minor releases up to version 3.2) are outlined in more detail below.

Enhanced aerodynamic data sets

One of the modifications concerns the definition of aerodynamic data sets. In previous CPACS releases these so-called aero performance maps consisted of four-dimensional arrays containing the aerodynamic force and moment coefficients as well as damping derivatives of an aircraft depending on Mach number, Reynolds number, angle of yaw and angle of sideslip. It turned out that such a full-factorial representation of aerodynamic data lacks physical relevance, since many combinations of the independent variables describing the aircrafts' state do not represent realistic flight conditions. To allow for a more flexible definition of the aero performance maps, the coefficients and derivatives are now stored as vectors of the same length, whereby entries having the same index together represent a single flight state. Furthermore, the aerodynamic data representation is transformed from the body-fixed into the aerodynamic coordinate system. This allows quick judgments of the correctness of the provided values by the aerodynamic specialists, since they often bring experience in the expected magnitude of coefficients expressed in this coordinate system. Additional elements are added to the performance maps to specify boundary conditions, including atmospheric data as well as settings of movable devices (e.g., landing gear status, control surface deflections, etc.). Operational limits can now be specified in terms of angle of sideslip and angle of attack limitations for a given set of Mach and Reynolds numbers. Maps with delta values of aerodynamic coefficients for incremental control surface deflections are still defined as a CPACS array type as introduced in Sec. 3.3. However, due to the aforementioned reformulation of the basic aerodynamic coefficients these arrays are only two-dimensional and not five-dimensional as in previous versions of the data format. An additional element referencing the state of the flight control system allows to specify the change in aerodynamic characteristics due to simultaneous control surface deflections.

Lessons learned from the development of the aerodynamic coefficients representation is that it requires a large effort to find the right balance between the simplicity of the chosen representation strived for versus the general applicability to all identified use-cases, often requiring complex constructs. With the current representation, the development team believes to have obtained the least complex representation covering all use-cases identified: as simple as possible, as complex as necessary. Next to this - even when the syntax of the representation is fully understood by all users - the semantically correct interpretation of the provided data by engineers from disciplines other than aerodynamics is experienced to not be trivial. To make a first step towards a better alignment of data provision and interpretation among the engineers of different disciplines, an interpolation library will be established and provided next to the TiXI and TiGL libraries described in Sec. 3.5.

Modeling of component segments for defining wing internal structures

Another major modification affects the geometric modeling of component segments within the wing definition. Component segments are areas between certain wing sections containing the wing structure, control surfaces or wing tanks defined in the normalized coordinates ξ (chordwise direction) and η (spanwise direction). In CPACS 3 these coordinates changed from rectangular to body fitted coordinates to prevent the possible definition of points between 0 and 1 being outside the wing. This required significant adaptions of the geometry library TiGL, which is, after the release of TiGL 3, no longer backward compatible to previous CPACS releases.

The possibility to define structural components actually (partly) being outside the wing became apparent when using CPACS to define non-conventional aircraft such as box-wing configurations and by considering more advanced geometric considerations such as aeroelastic deformation of wings and analyzing the corresponding effect on the wing internal structure in detail. CPACS has developed into the current version by testing its applicability within many projects considering a multitude of different configurations. Introducing a major non backward-compatible change provides a large effort - especially if the corresponding libraries need to be overhauled as well. The new release however allows the even more flexible definition of all kinds of aircraft configurations.

Addition of a parametric model of engine nacelles

A parametric description of engine nacelles and pylons, of which a prototype was introduced in the previous release, is now fully implemented in CPACS 3. Fig. 5 shows a nacelle geometry TiGL derived from the new parametrization. The center, core and fan cowl are defined by radial sections in flow direction (two-dimensional profiles in cylindrical coordinates) and guide curves to control the extrusion of these profiles. A comprehensive description of the nacelle and pylon parametrization and its implementation in TiGL is given by Siggel et al. [24].

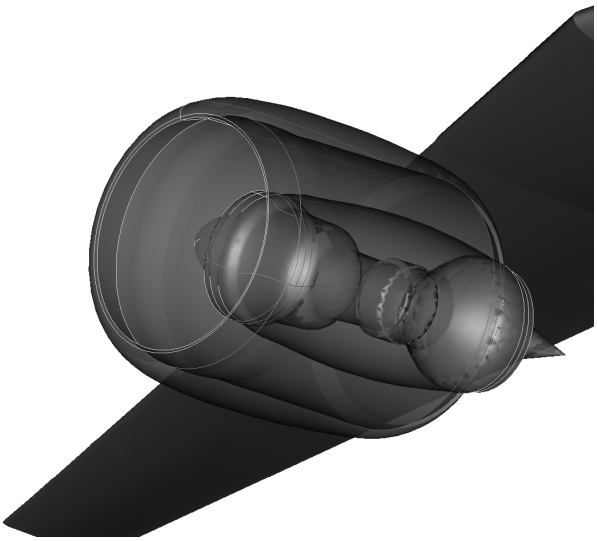

Figure 5: Engine nacelle displayed in TiGL viewer.

Inclusion of a generally applicable mission definition

From applications of previous CPACS versions within aircraft design and analysis tasks, the need arose to include an extensive and generally applicable mission definition within the data format. Next to coping with the relatively straightforward mission definitions used for the design of transport aircraft, the more flexible definition introduced in CPACS 3 can cope with detailed flight paths (flying actual waypoints, applying step-climb profiles, etc.) and with missions for non transport aircraft such as military jets or UAV's and rotorcraft. The new <*missionDefinitions*> node is positioned directly at the root CPACS-node (see Fig.2), so a multitude of missions for which an aircraft or rotorcraft has to be designed can be defined. Within each vehicle definition, a corresponding <*performanceRequirements*> node is introduced, in which the link to the respective mission definition(s) to be fulfilled is established. A mission is built up of a sequence of segments - optionally grouped in segmentBlocks - in which the respective constraints and end conditions on the mission profile can be defined. Furthermore, point performance requirements - such as a service ceiling or sustained turns - can be defined and linked to specific mission segments. Creating the mission definition has been a largely iterative process in which the opinion of multiple DLR internal and external special-

ists in mission simulation and flight performance analyses have been involved. The major challenge has been finding the right balance between the provided flexibility and simplicity of the definition.

Possibility to include individual toolspecific schemas

Fig. 2 shows that CPACS provides a node called *<toolspecifics>* which contains tool-specific process parameters for a given list of tools that can automatically be executed in decentralized workflow environments (see Sec. 2.1). An example is given by Liersch and Hepperle [19] for the aerodynamic analysis tool LIFTING_LINE. Having these parameters together with the aircraft or helicopter model in a consistent data set significantly supports the workflow integration. However, the application of this feature in multidisciplinary projects like IDEaliSM [20] or AGILE [7] revealed that maintaining this part of the XML schema is quite challenging since the development speed of tools usually differs from the CPACS release cycles. To avoid releasing a new CPACS version just for tool-specific updates or slowing down tool development since no direct release of CPACS is planned, it was decided to include these nodes in the schema while using separate namespaces. This is achieved by the XSD *<any>* element. It allows to include an individual namespace but requires a corresponding XML Schema Definition to validate data. The creation and maintainance of this separate schema will be in the responsibility of the tool owner. Thus, the release cycles of the tool-specific schemes can be adapted to the development speed of the corresponding tools, whilst ensuring the validation of all provided input data is possible.

4.2 Establishing a collaborative development process

The CPACS development is coordinated by DLR's Institute of System Architectures in Aeronautics and regularly released under the Apache 2.0 open-source license. It is maintained via the software development platform GitHub, which allows to document the development and discussions in a transparent way using a variety of management methods. Problems are discussed via an issue forum and the status of these issues is tracked in Kanban boards. This provides developers and users an overview of the changes and extensions planned for a certain release and provides the means to keep track of the development process. It also facilitates to involve new stakeholders during the development process and to document possible decisions for future reference.

To ensure acceptance of the proposed updates throughout the entire community, all changes are implemented in a release candidate which is announced at least four weeks before an official release is planned. This announcement is made via a CPACS mailing list and a blog entry on the CPACS homepage [10]. The homepage furthermore provides an online documentation of all release versions including the next release candidate and provides links to the corresponding XSD documents for validation of CPACS data sets. In CPACS 3.2 example files have been added and a unit testing method is implemented to ensure that the examples are always valid with the current schema. Feedback from the community shows that such examples are extremely helpful when learning how to use CPACS.

In addition to passive notifications via GitHub, e-mail and homepage, the authors consider requesting active confirmation or rejection of major changes, so that possible inconsistencies can be discovered well in advance of a planned release. For this purpose, representatives of the participating institutions and disciplines could either make such decisions in the name of their discipline or pass on current developments to the persons concerned.

A very important aspect in the development process is to align CPACS modifications and new features with the library development. As outlined in Sec. 3.5 the geometry library TiGL has become a comprehensive and very important software for the CPACS community, not only because it translates the CPACS parameters into a three-dimensional geometry, also because it is a collection of methods that the community has agreed on in terms of how to interpret the geometry parametrization. When all partners involved in the design process use TiXI and TiGL to wrap their analysis modules, next to the syntactical, also the semantically correct interpretation is guaranteed. Regular online meetings between the corresponding developers help to avoid incompatibilities between TiXI, TiGL and CPACS.

Extensive discussions and agreements take place in the form of regular developer meetings targeted to be held two times per year. Experience has shown that due to the wide range of aspects covered by CPACS, the topics for discussion must be limited. Nonetheless, the on-site exchange is very helpful in accelerating the development process and promotes community building.

4.3 Application of CPACS in collaborative design projects

To provide an insight in the application range of CPACS, the current section lists an overview of some of the projects in which the data format has been applied and extended until the time of writing this paper. In fact, most extensions of the data format are driven by needs from these projects, focusing on the utilization of collaborative design methods to perform complex product design exercises.

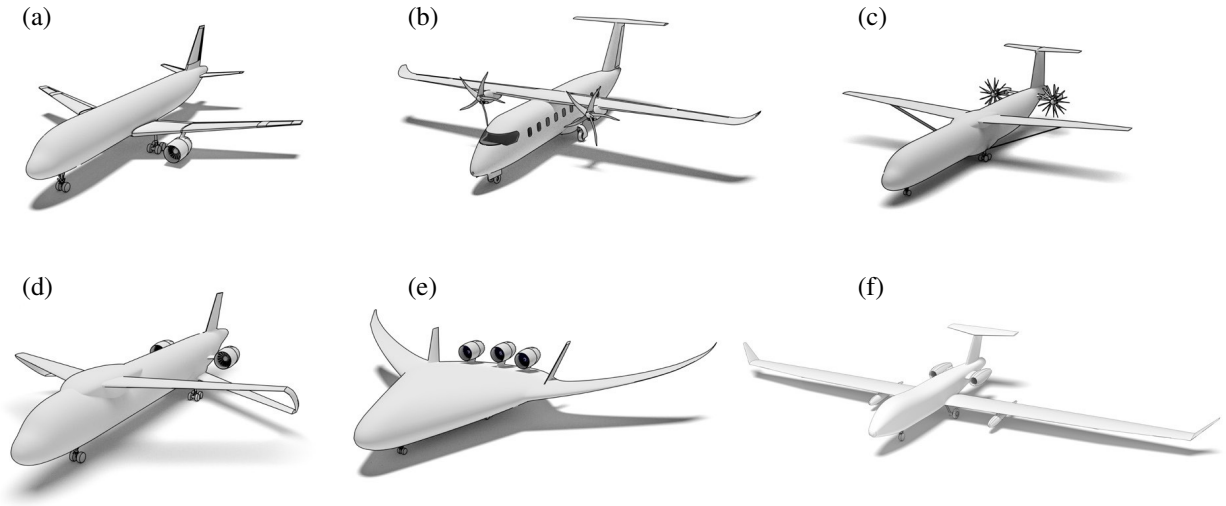

Figure 6: Aircraft configurations investigated in the FrEACs (configurations c and e) and AGILE (all configurations) project.

FrEACs

As fourth in a series of projects focused on building up a design environment for the collaborative and distributed analysis of aircraft configurations within DLR, the *Future Enhanced Aircraft Configurations* (FrEACs) focused on applying the established tools and methodologies to the design of non-conventional aircraft configurations (Fig. 6(c) and (e)). Within an automatic simulation workflow spanning multiple levels of analysis fidelity, the potential of a strut-braced wing configuration as future short- to medium-range aircraft was successfully investigated [21]. Furthermore, the flight mechanical and handling qualities of a blended-wing-body configuration were investigated and tested within DLR's AVES flight simulator [15]. During the project, the team progressively built up and utilized CPACS-based workflows for both configurations including over 24 analysis modules from 11 different DLR institutes. The introduction of design camps - special meetings focused on conducting a specific task, e.g. the downselection of possible design concepts - proved to be a very useful addition to the design process within the project. During the final design camps, the built-up simulation workflow spanning multi fidelity levels was utilized to perform designs of experiments. Focus during these design camps was on comparing results on the three involved fidelity levels and collaborative result interpretation, in which combining the knowledge of all participating team members is of invaluable importance.

ATLAs

Within the *Advanced Technology Long-range Aircraft Concepts* (ATLAs) project, which was successfully completed in December 2019, CPACS served as common language in a collaborative aircraft design project involving 13 DLR institutes. In ATLAs, advanced assessment procedures were employed to evaluate the impact of modern technologies, such as hybrid laminar flow control or a CO_2 management system for cabins - on future aircraft configurations in a more holistic way [14]. The underlying use-case for these studies is the design of an advanced mid-range aircraft. The various disciplines and technologies shown in Fig. 8 underline the importance of not only considering the aircraft itself, but also modeling operational aspects in CPACS, e.g. to assess the climate impact of new designs. Within the ATLAs project, the parametric logic of CPACS has proved to be an enabler for performing technology evaluation studies combining multiple promising technologies in a single configuration.

AVACON

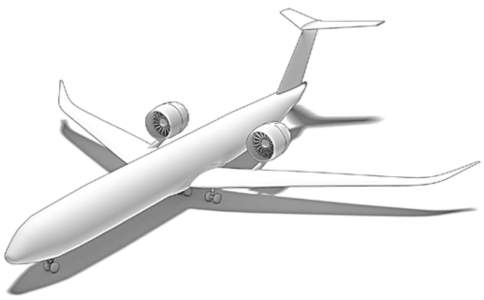

Figure 7: Over-wing nacelle configuration studied in the AVACON project.

Similar to ATLAs the research project AVACON focuses on advanced technologies in collaborative design within a consortium of nine partners from industry, re-

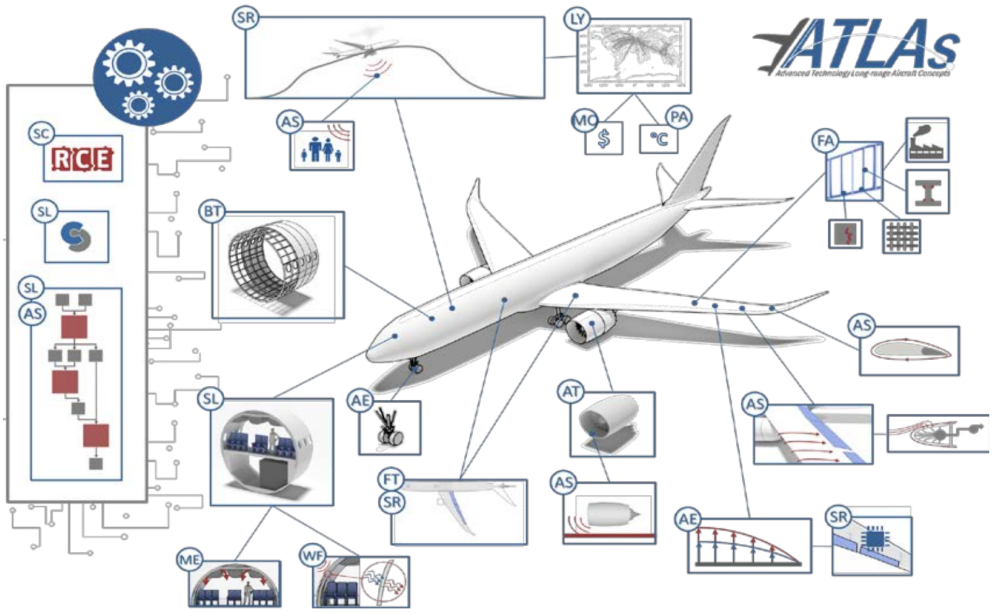

Figure 8: Functional breakdown of disciplines involved in the ATLAs project [14].

search entities and universities funded by the national aeronautic research program LuFo V-3. Starting from the same baseline configuration, the target aircraft configurations and selected technologies differ from ATLAs. For example, a central study within AVACON focuses on the impact of an over-wing engine integration as sketched in Fig. 7 [14]. Within the project, the parametric definition of engines, nacelles, wings and fuselages in CPACS enabled performing aircraft-engine integration studies.

IDEaliSM

Within the project *Integrated and Distributed Engineering Services framework for MDO* (IDEaliSM), a formalized framework for the creation of a service-oriented product development process has been established and successfully utilized to connect an aircraft OEM with a tier-1 and tier-2 supplier for aircraft structural components. The formalized framework serves as basis to perform front-loading of the design process by creating and combining engineering services and connecting these both within and across company borders using automated business and simulation workflows [26]. Upon completion of a conceptual aircraft design task - involving analysis modules progressively adding information to CPACS - its resulting CPACS data deck is uploaded to a central data server within a neutral IT domain [20]. After the Tier-1 supplier receives the corresponding request for proposal - in this case for the design of a vertical tailplane, - the CPACS data deck is downloaded and its geometry definition is automatically converted to the data format natively used at the supplier. The same holds true for the subsequent connection between the tier-1 and tier-2 supplier, the latter providing a single structural component for the structural assembly. The resulting structural assembly and its properties are added to the CPACS data deck and re-uploaded to the central data server. In this way, the aircraft OEM has the ability to take detailed considerations into account already in relatively early design stages. In the described application, the data consistency between the parties involved is ensured by automating data exchange through the central data server and through adoption of common data formats such as CPACS. Both the automation and integration of the design processes lead to a significant reduction in overall process lead time [25], which at its turn allows for executing an increased amount of design iterations.

Agile

The Horizon 2020 project *Aircraft 3rd Generation MDO for Innovative Collaboration of Heterogeneous Teams of Experts* (AGILE) was conducted from 2015 to 2018 addressing multidisciplinary optimization based on distributed analysis frameworks (demonstrated with RCE, see Sec. 2.1). The consortium composed of 19 international partners from EU, Russia and Canada has proven a speed up of 40% in solving realistic MDO problems for conventional, strut-braced, box-wing and BWB configurations (see Fig. 6) compared to today's state-of-the-art approaches in overall aircraft design [7]. Based on CPACS as central data exchange model it has been demonstrated that its current development status allows to model a variety of unconventional aircraft configura-

tions including the corresponding analysis results.

VicToria

The DLR internal research project VicToria (*Virtual Aircraft Technology Integration Platform*) aims at a comprehensive digital description and development of aircraft including flight-physical effects on a high-fidelity level. This is achieved by the development of a distributed collaborative MDO environment which combines high-fidelity analysis for aerodynamics and structures with conceptual aircraft design methods [12]. The use of CPACS in the consortium of 12 DLR institutes and facilities helps to reduce the gap between low and high-fidelity modeling. One of the results is the detailed parametrization of engine pylons and nacelles presented in Sec. 4.1. For selected disciplines, the bridge to use CPACS as initial basis for consecutively performing high-fidelity analyses is established.

TRIAD

A DLR project called *Technologies for Rotorcraft in Integrated and Advanced Design* (TRIAD) has been initiated in January 2018 to investigate the impact of new technologies (e.g., electric propulsion or the design of fiber reinforced materials for fuselage cells using Finite Element Methods) on the overall rotorcraft design considering different rotor, wing and propeller configurations. Five DLR institutes are involved exchanging geometry and aerodynamic performance data using CPACS in a simulation workflow. The application of the helicopter node in such projects ensures consistency between the aircraft and rotorcraft ontology, especially when new or updated definitions - such as the aerodynamic performance map described in Sec. 4.1 - shall be used for both vehicle types.

IFAR-X Challenge

The "IFAR-X Challenge" is a voluntary framework for young engineers and scientists from over 20 countries to collaborate in developing a multi-disciplinary aircraft design process. The common goal is to assess and integrate green technologies in future aircraft. Organized by the Russian Central Aerohydrodynamic Institute (TsAGI) and DLR, both CPACS and other tools focused on collaboration were introduced in a three-day seminar. Such an opportunity allows CPACS developers to understand how the technology can be most efficiently explained to new users. In this context, additional training material (e.g., example files, tutorials) was created and made freely available to the community. Furthermore, the inclusion of new users is a good way of identifying inherent weaknesses in the definitions within the data format which are sometimes overseen by experienced users - within the IFAR-X challenge, certain data interpretations were taken for granted by the experienced users and could have revealed the possible cause of inconsistencies. The IFAR-X challenge significantly contributes to the expansion of the community and the establishment of a common language in aircraft design. Within the challenge, CPACS enables the quick integration of technologies and ideas from research establishments across the globe.

HOLISHIP

The H2020 European Research Project HOLISHIP (*Holistic Optimisation of Ship Design and Operation Life Cycle*; 2016-2020) project gives CPACS developers the opportunity to think outside the box by using the modeling approach described in Sec. 3.2 to support the development of the data exchange format Holispec [11]. During a workshop in Ålesund, Norway, a first data hierarchy for the design process of vessels was developed combining CPACS with previous work from the Maritime Research Institute Netherlands (MARIN). Just as the aforementioned TRIAD project, HOLISHIP shows that the principles of CPACS can be applied to all kinds of collaborative product design initiatives.

5. CURRENT STATUS OF ESTABLISHING CPACS AS A COMMON LANGUAGE IN AIRCRAFT DESIGN

As mentioned in Sec. 3, the development of CPACS started in 2005. Seven years later Nagel et al. [22] evaluated the data model with respect to its application in various collaborative projects asking whether and how it is possible to establish a common language in aircraft design. In this context, CPACS has been used to demonstrate that a standardized data model can have a significant impact on the efficiency of collaborative research and design efforts. The authors also derive a list of requirements which are crucial for the implementation of a standard for communication in aircraft design.

As a summary of the experiences discussed in the present paper a brief up-to-date overview of with respect to the most important requirements is provided below.

Public availability of all technical information

By using GitHub, all adaptations to the XML schema are publicly accessible. Technical reports about current changes have contributed to focus the discussions. Although using the software development platform is helpful in keeping track of all issues and modifications, many features are hesitantly accepted by the community due to the complexity of GitHub. Therefore, the importance of providing a user-friendly homepage that informs users

and provides access to technical details has been recognized. The combination of a development (GitHub) and information platform (Homepage) finally increased the distribution of technical information in the community and in turn decreased maintenance effort for developers.

Availability of training material, training courses and technical help

The automatic generation of documentation based on Microsoft Compiled HTML Help significantly reduced the development effort of CPACS. The compiled CHM files are delivered with every release, but are reluctant to be used by the community due to the less intuitive user interface and, in part, platform-dependent visualization problems. To solve this problem, the documentation is additionally integrated in the CPACS homepage and can be viewed directly in the browser. This feature is used intensively by the community, even though useful extensions such as a search function or a navigation bar are still missing.

Experience has also shown that example files such as described in Sec. 4.2 are the most efficient way to support the introduction to CPACS and to accelerate the integration of own software in CPACS-based workflows. Thus, this aspect will be considered more thoroughly in future releases.

Additional training material was created as part of the IFAR-X design challenge (see Sec. 4.3) and made publicly available on GitHub. It has been observed that a certain momentum has emerged in the community, where users have supported each other and contributed with example programs and tutorials on GitHub.

Finally, it should be mentioned that training courses can contribute significantly to a fast understanding of CPACS. First experiences were gained, for example, with an introduction for doctoral students at TU-Braunschweig or for the participants of the IFAR-X design challenge. The training material for these courses is again freely available on GitHub.

Effort for learning the standard significantly lower compared to the technical problem to be solved

It has been shown that a basic introduction to CPACS via seminars takes about two to three days. Since CPACS is based on XML and thus on an open standard, the comprehensive documentation reduces the additional workload to a few hours (mainly to implement XML processors and APIs), which is usually less than the solution of technical problems addressed in aircraft design. The autodidactic training period is furthermore reduced by sample files provided since CPACS 3.2.

Possibility to use the model without charge

Due to a lack of experience with the usage of proprietary standards, the influence of this factor can only be assessed hypothetically. However, it can be assumed that the acceptance for CPACS would be significantly lower if the use of the file format or the associated software libraries were subject to paid licenses.

Continuity of conventions

In some cases, this aspect is difficult to assess. On the one hand, CPACS is used as a de-facto standard in aircraft design and therefore changes should not be introduced too frequently. On the other hand, a fast development process is often necessary, especially when new definitions are required for upcoming projects. Therefore, two approaches are followed: First, changes are introduced more dynamically when a major release has just been released, as experience has shown that many tools require time to adapt to new CPACS versions. Furthermore, project-specific XML schemas are used in some cases, in which changes are introduced and tested only within a limited number of users before they are incorporated into the public release.

Structured development process with predictable release cycles

A structured development process has been introduced as described in Sec. 4.2 which contributes to the establishment of CPACS as an exchange format. Feedback from the community revealed that predictable release cycles are important.

Establishing a steering committee coordinating the development process

There is a strong interest from DLR to involve Universities and Industry in forming a common steering committee. Exemplary for this are the procedures of W3C in establishing web standards. The framework for the implementation of such a committee are currently being elaborated.

High propagation of the standard through an ecosystem of libraries and applications

CPACS has established as a de-facto standard for DLR internal aircraft design projects and is increasingly used by a large number of national and international research institutions and industrial partners (see Sec. 4.3). The straightforward and proper application of the definitions within CPACS is enabled by the open-source development and availability of the corresponding software libraries TiXI and TiGL. Together with the Remote Component Environment (RCE) as open-source process integration framework, CPACS, TiXi and TiGL form an ad-

vanced ecosystem in support of collaborative aircraft design.

6. CONCLUSION

The present paper highlights that collaborative aircraft design projects involving heterogeneous disciplines may significantly benefit from using a common language for data exchange. It not only supports the integration of decentralized workflows, but also assures that all partners are correctly interpreting the data and working with consistent models. Next to other aviation data models for various disciplines, CPACS provides an extensive parametric ontology for fixed-wing aircraft and rotocraft. Based on the XML Schema Definition CPACS is flexibly expandable and both human readable and computer processable.

Recent enhancements of CPACS include a detailed parametrization of engine nacelles and pylons, more flexible aerodynamic performance maps, body-fitted coordinates for wing component segments and the possibility to include individual tool-specific data. These modifications result from a collaborative development process based on GitHub and on-site meetings with the respective users and developers.

Publishing CPACS and its corresponding libraries under an open source license and applying it in various decentralized aircraft design projects promoted its use in heterogeneous teams and led to a significant community growth. Thus an important step towards a standard for a common language in aircraft design has been achieved. In the course of projects like AGILE or IDEaliSM it has been quantitatively proven that collaborative design tasks based on CPACS can significantly increase in performance and quality. This positively answers the question from Sec. 1 whether the use of a common language has linked the aircraft design process more closely.

Upcoming challenges include a detailed representation of aircraft systems to meet the requirements of various research projects on the climate impact of aviation and the design of sustainable aircraft. To enable the increased complexity and heterogeneity of these projects while ensuring a clear semantic interpretation of data, future research will also focus on combining advanced data modeling methodologies from the field of Semantic Web with state-of-the-art approaches in aircraft design (e.g., Model Based Systems Engineering).

REFERENCES

[1] American Institute of Aeronautics and Astronautics. Flight Dynamics Model Exchange Standard. Technical Report ANSI/AIAA S-119-2011(2016), American National Standard, 2011.

[2] Markus Ast, Martin Glas, and Tobias Roehm. Creating an Ontology for Aircraft Design. In *Deutscher Luft- und Raumfahrtkongress 2013*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2013.

[3] Jon S. Berndt. JSBSim, an open source platform independent flight dynamics model in C++. *JSBSim Reference Manual v1.0*, 2011.

[4] Brigitte Boden, Jan Flink, Robert Mischke, Schaffert Kathrin, Alexander Weinert, Annika Wohlan, and Andreas Schreiber. RCE: An Integration Environment for Engineering and Science. *arXiv:1908.03461*, 2019.

[5] Brigitte Boden, Jan Flink, Robert Mischke, Kathrin Schaffert, Alexander Weinert, Annika Wohlan, Caslav Ilic, Tobias Wunderlich, Carsten M. Liersch, Stefan Goertz, Pier Davide Ciampa, and Erwin Moerland. Distributed Multidisciplinary Optimization and Collaborative Process Development Using RCE. In *AIAA Aviation 2019 Forum*. American Institute of Aeronautics and Astronautics, June 2019.

[6] D. Böhnke, M. Litz, Björn Nagel, and S. Rudolph. Evaluation of Modeling Languages for Preliminary Airplane Design in Multidisciplinary Design Environments. In *Deutscher Luft- und Raumfahrtkongress 2010*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2010.

[7] P. D. Ciampa and Björn Nagel. AGILE the Next Generation of Collaborative MDO: Achievements and Open Challenges. In *2018 Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics, 2018.

[8] Shubhangi Deshpandea, Layne T. Watsona, Nathan J. Lovec, Robert A. Canfield, and Raymond M. Kolonay. ADML: Aircraft Design Markup Language for Multidisciplinary Aircraft Design and Analysis. Technical Report TR-13-07, Computer Science Technical Reports, 2013.

[9] DLR Institute for Simulation and Software Technology. Tixi: fast and simple xml interface library. http://tixi.sourceforge.net/Doc/index.html, 2019. [Online; accessed 12-November-2019].

[10] DLR Institute for System Architectures in Aeronautics. CPACS: Common Language For Aircraft Design. www.cpacs.de, 2019. [Online; accessed 13-November-2019].

[11] Marteen Flikkema, Martin van Hees, Timo Verwoest, and Arno Bons. HOLISPEC/RCE: Virtual

Vessel Simulations. In Apostolos Papanikolaou, editor, *A Holistic Approach to Ship Design*, volume 1, chapter 15, pages 465–485. Springer, 2019.

[12] Stefan Goertz, Caslav Ilic, Jonas Jepsen, Martin Leitner, Matthias Schulze, Andreas Schuster, Julian Scherer, Richard Becker, Sascha Zur, and Michael Petsch. Multi-level MDO of a long-range transport aircraft using a distributed analysis framework. In *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics, June 2017.

[13] William J. Gordon. Spline-Blended Surface Interpolation Through Curve Networks . *Journal of Mathematics and Mechanics*, 18(10):931–952, 1969.

[14] Johannes Hartmann, Till Pfiffer, Berit Breymann, Daniel Silberhorn, Erwin Moerland, Marco Weiss, and Björ Nagel. Collaborative Conceptual Design of a Mid-Range Aircraft Under Consideration of Advanced Methods for Technology Assessment. In *31th Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences, 2018.

[15] Yasim J. Hasan, Jan Flink, Sebastian Freund, Thomas Klimmek, Richard Kuchar, Carsten M. Liersch, Gertjan Looye, Erwin Moerland, Till Pfeiffer, Mario Schrader, and Sebastian Zenkner. Stability and Control Investigations in Early Stages of Aircraft Design. In *2018 Applied Aerodynamics Conference*. American Institute of Aeronautics and Astronautics, June 2018.

[16] Richard M. Keller. Ontologies for Aviation Data Management. In *35th Digital Avionics Systems Conference (DASC)*. IEEE Aerospace & Electronic Systems Society, 2016.

[17] Kuhlmann Kuhlman. generateDS – Generate Data Structures from XML Schema. www.davekuhlman.org/generateDS.html, 2019. [Online; accessed 01-January-2020].

[18] Immo Landwerth. XML Schema documentation plug-in for Sandcastle Help File Builder (SHFB) — GitHub. https://github.com/terrajobst/xsddoc, 2016. [Online; accessed 4-December-2019].

[19] Carsten M. Liersch and Martin Hepperle. A distributed toolbox for multidisciplinary preliminary aircraft design. *CEAS Aeronautical Journal*, 2(1-4):57–68, 2011.

[20] Erwin Moerland, Sebatian Deinert, Daoud Fernaß, Jochen Dornwald, and Björn Nagel. Collaborative aircraft design using an integrated and distributed multidisciplinary product development process. In *30th Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences, 2016.

[21] Erwin Moerland, Till Pfeiffer, Daniel Böhnke, Jonas Jepsen, Sebastian Freund, Carsten M. Liersch, Gabriel Pinho Chiozzotto, Carsten Klein, Julian Scherer, Yasim J. Hasan, and Jan Flink. On the Design of a Strut-Braced Wing Configuration in a Collaborative Design Environment. In *17th AIAA Aviation Technology, Integration, and Operations Conference*. American Institute of Aeronautics and Astronautics, June 2017.

[22] Björn Nagel, Daniel Böhnke, Volker Gollnick, P. Schmollgruber, A. Rizzi, G. La Rocca, and J. J. Alonso. Communication in Aircraft Design: Can We Establish a Common Language? In *28th Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences, 2012.

[23] Open Cascade SAS. OPEN CASCADE part of Capgemini. www.opencascade.com, 2019. [Online; accessed 11-December-2019].

[24] Martin Siggel, Jan Kleinert, Tobias Stollenwerk, and Reinhold Maierl. TiGL: An Open Source Computational Geometry Library for Parametric Aircraft Design. *Mathematics in Computer Science*, 7(1):23, 2019.

[25] Tobie van den Berg, Bastiaan Beijer, and Erwin Moerland. Application of an integrated and distributed multidisciplinary product development framework to a multi-tier aircraft design case. In *AIAA Aviation 2019 Forum*. American Institute of Aeronautics and Astronautics, June 2019.

[26] Stefan van der Elst, Erwin Moerland, Johnny van Lugtenburg, Luc Hootsmans, Sebastian Deinert, Martin Motzer, and Cristina Fernandez. Industrial service-oriented process methodology. Technical Report D2.3, V. 1.0, ITEA 3, 2017.

[27] Daniel Veillard. Release of libxml2-2.9.9. http://mail.gnome.org/archives/xml/2019-January/msg00000.html, 2019. [Online; accessed 11-December-2019].

[28] Helmut Vonhoegen. *Einstieg in XML: Grundlagen, Praxis, Referenz*. Rheinwerk Computing. Rheinwerk Verlag GmbH, Bonn, 8., aktualisierte auflage edition, 2015.

[29] W3C. XML Linking Language (XLink) Version 1.1. www.w3.org/TR/xlink, 2010. [Online; accessed 4-December-2019].

[30] W3C. The Extensible Stylesheet Language Family (XSL). www.w3.org/Style/XSL, 2017. [Online; accessed 4-December-2019].

[31] W3C. XML Path Language (XPath) 3.1. www.w3.org/TR/xpath-31, 2017. [Online; accessed 4-December-2019].

[32] T. Zill, P. D. Ciampa, and B. Nagel. Multidisciplinary Design Optimization in a Collaborative Distributed Aircraft Design System. In *50th AIAA Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 2012.