AVOIDING INCONSISTENCIES BETWEEN DATA MODELS IN COLLABORATIVE AIRCRAFT DESIGN PROCESSES

J. Jepsen, P.D. Ciampa, B. Nagel

Deutsches Zentrum für Luft- und Raumfahrt (DLR), Lufttransportsysteme (LY),
Blohmstraße 18, 21079 Hamburg, Deutschland

Abstract

Setting up a truely holistic aircraft design process considering all the relevant disciplines, requires a new level of collaboration. Design tools from different fields of expertise need to be combined in order to cover each discipline thoroughly. While the use of a central data format reduces the effort necessary to interconnect all the design tools, inconsistencies can remain hidden inside the process. Despite the central data format a conversion needs to be implemented from the central data format to the representation used in each application. Especially in multi-fidelity processes this can be a source of errors. Abstracting from a detailed model to a reduced model and back, requires knowledge which is usually based on assumptions. In order to ensure consistency, any assumption made during the abstraction needs to be valid for the data at hand. In this paper inconsistencies discovered during the development of the design process in Digital-X are used to illustrate the challenge in their setup. From the experience with the implementation of collaborative design processes a generalized method was developed for avoiding inconsistencies. The application of the method on the specific example from the Digital-X project indicates the value for the setup of future design processes.

1. INTRODUCTION

Aircraft design is a complex task in which knowledge of a large number of disciplines needs to be included/combined. Due to its high complexity it is difficult for a single party to deal with all the disciplines in detail. Collaboration between different experts is required for a holistic design considering all the relevant disciplines. Methods for collaboration are currently investigated at DLR Air Transportation Systems. The development of collaborative and distributed design processes is pursued in different projects such as AGILE, Digital-X, IDEaliSM, FrEACs, VicToria. In this paper we show how inconsistencies can pass unnoticed during the assembly of aircraft design tools into a single design process and how to avoid them. As an example a subprocess developed for the Digital-X project is used [4]. One of the goals in Digital-X is the setup of a MDO (Multidisciplinary Design Optimization) process, including high fidelity FEM (Finite Element Method) and CFD (Computational Fluid Dynamics) methods, aero structural coupling and a large number of load cases. The optimization is driven by planform parameters of the wing (as independent design variables). However, in order to ensure the feasibility for each of the design evaluated within the optimization process, a preprocessor was developed which modifies the wing position and the tail sizes according to stability and controllability criteria. The optimization process starts from a reference configuration containing the initial aerodynamic shape, on which planform changes are performed by the optimizer. Taking the reference configuration containing the wing changed by the optimizer, the preprocessor uses low fidelity methods, such as AVL (Athena Vortex Lattice) and handbook methods, for the estimation of component masses and aerodynamic performance. It also changes the wing position and tail sizes according to stability and controllability margins of the original reference aircraft. Due to the interdependencies of the mass estimation, the aerodynamic performance and the stability and controllability criteria, the first two are linked by the latter, the process is run in an iterative loop. Once the wing position and tail sizes are converged, the process deliveres a design consistent with the applied overall aircraft design methods. Generating a consistent aircraft design requires consistency between all the methods and models used by the tools in the process. Although a central data format is used for the exchange of data between the tools, the consistency highly depends on the interpretation of the central data model for each tool. The interpretation usually differs with the discipline and with the level of fidelity. In this paper inconsistencies in the development of the preprocessor and the approached solution are described. From the experiences collected during the project, a suggestion for the implementation of future multi-disciplinary and multi-fidelity processes is presented. In the following section a short overview of the optimization process from the Digital-X project is introduced and the context and structure of the preprocessor is explained. In section 3 examples for inconsistencies recognized during the development of the preprocessor are shown followed by a description of the method used for solving inconsistencies in section 4. Details on how the suggested method was used to achieve consistency between the models are given in section 5.

2. OPTIMIZATION PROCESS

The objective for the optimization process in Digital-X is optimizing the wing shape for minimum fuel burn on a design mission. Included in the process are a large number of disciplinary tools with variing level of fidelity, ranging from handbook methods to detailed FEM and CFD analysis. As an optimization architecture the MDF (multidisciplinary feasible) approach, as described by [11] was chosen [6]. It contains a sequential chain of subprocesses starting from preliminary design methods to high fidelity analysis and sizing. The structure of the optimization process is depicted in FIG 1 [4]. As a data exchange between the different design and analysis tools the central data format CPACS (Common Parametric Aircraft Configuration Schema) is used in the design process [12]. Each of the four design stages

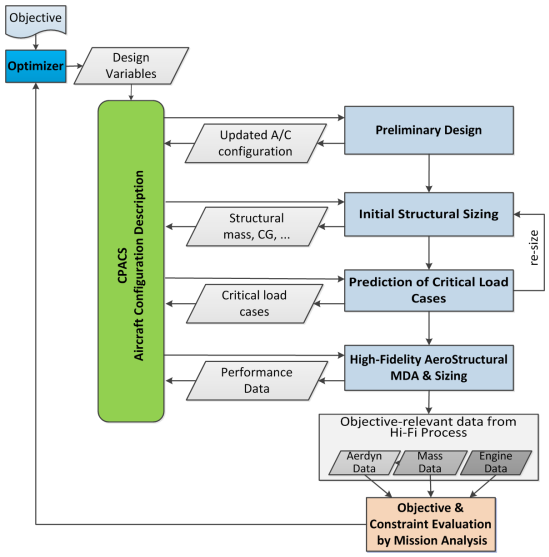


FIG 1: Digital-X gradient free optimization process [4]

Preliminary Design, *Initial Structural Sizing*, *Prediction of Critical Load Cases* and *High-Fidelity AeroStructural MDA & Sizing* writes its results into a CPACS file which is transfered to the next stage. Integrating all the disciplinary tools into an automated design process is done using the integration framework RCE (Remote Component Environment) [9], [1]. The *Preliminary Design* stage is composed of two parts, the preprocessor and the actual predesign. While the task of the preprocessor is the creation of a feasible overall aircraft design based on the reference aircraft and wing shape updated by the optimizer, the predesign evaluates the aircraft geometry created by the preprocessor, using different low fidelity methods. It performs a wing mass estimation using simple beam model under consideration of basic load cases for sizing and handbook methods from LTH (Luftfahrt Technisches Handbuch) [3] for secondary wing mass estimation, a mission simulation and handbook methods for integrating the wing and fuel mass estimations into the overall aircraft masses [2]. In the *Initial Structural Sizing* stage a higher fidelity method for structural sizing of the components is used. Once the *Prediction of Critical Load Cases* stage has identified critical load cases from a vast number of load cases, the *Initial Structural Sizing* stage is triggered again to resize the aircraft components under consideration of all the critical load cases [10], [8]. This loop is continued until convergence is reached and the result is passed to the *High-Fidelity AeroStructural MDA & Sizing* stage. There a final aero-structural analysis and sizing is performed using high fidelity FEM and CFD methods [13]. After the design is completed, the performance of the aircraft is evaluated and the constraints can be evaluated to provide the necessary results for the optimizer.

The described process is a collaboration of eight different facilities from DLR distributed over six sites in Germany. Since the examples described in this paper are based on experiences in the development of the preprocessor, the following paragraphs describe the structure of the preprocessor in more detail. For information on the other parts consider reading the individual publications.

3. PREPROCESSOR

FIG 2 shows a flow chart of the preprocessors structure as part of the complete process. As an input to the preprocessor an aircraft shape containing a wing designed by the optimizer and the top level aircraft requirements (TLAR) are used. From the aerodynamic shape of the aircraft and the TLARs two parallel processes are triggered. On the one side the aerodynamic performance of the aircraft and its components is evaluated and on the other hand the initial mass estimation is performed. For the aerodynamic performance evaluation AVL with additional corrections for transonic flow is used. Since this method can only estimate the lift and lift induced drag, handbook methods based on Raymer are used to add a zero lift drag prediction. The estimations of the overall design masses are performed using the open source conceptual aircraft design software VAMPzero, which incorporates a large number of handbook methods from different sources. From the geometric description of the wing and tails the volume of the fuel tanks is determined. In total the volume and center of volume of four separate fuel tanks are evaluated, namely the center fuel tank, the inner fuel tank, the outer fuel tank and a trim tank located in the horizontal tail plane. Together with the mass and location of the payload, the most forward and most backward center of gravity (COG) for the complete aircraft can be estimated. The minimun and maximum COG are then used to iteratively calculate the wing position and tail size according to the stability and controllability margin given for the reference aircraft using a method derived from [5]. Once a new wing position and tail size is found, the geometric changes are applied to the geoometry in the CPACS file. This process is repeated until the wing position and tail size are converged. Finally, the full aircraft configuration including the adapted wing position and tail sizes, as well as the aerodynamic performance and initial mass estimation are returned.

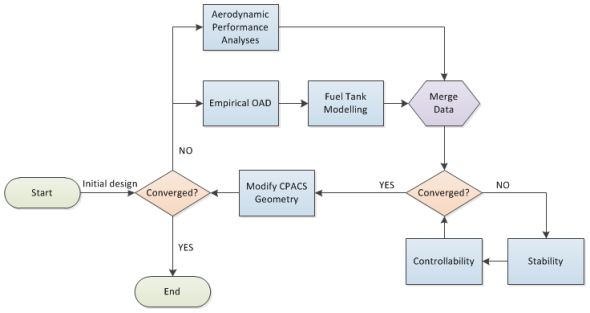


FIG 2: Preprocessor structure

The given description does not contain all the details of the preprocessor but gives a slightly simplified view. Some parts which are not relevant for the method described in this paper, such as the engine placement and the analysis of the one engine inoperative criteria were left out for the sake of simplicity.

Although CPACS is used for the data exchange throughout the preprocessor an interpretation of the aircraft model described by the data needs to be performed for every tool. In the described process, methods of different fidelity were used. And although their internal models are based on the same central model, there can still be inconsistencies be-

tween them. In the next section an example for inconsistencies which were identified during the development of the preprocessor is given.

3.1. Hidden Inconsistencies Revealed

In the previously described design process, tools of different level of fidelity are used raising consistency discrepancies. As an example, the NASA CRM (Common Research Model) aircraft model is taken into consideration, and difference in the interpretations of the wing models between the conceptual design tool VAMPzero and the implemented AVL method are presented. Since the AVL wing model uses all the wing segments defined in the CPACS file, for the remaining part of the paper, the CPACS geometry is used as representative for the AVL wing, neglecting the details for the transformation from CPACS to AVL. The CPACS file of the NASA CRM was generated from the step file available at `https://commonresearchmodel.larc.nasa.gov` and additional information such as the reference area was taken from [14]. The generated CPACS file contains a total of 20 wing segments. While the panels of the wing for AVL are created using the detailed shape reflecting all the individual segments, VAMPzeros internal representation consists only of a single trapezoid wing. Thus the detailed shape needs to be processed to find a suitable representation. VAMPzero contains an import method, which will read the parameters from the CPACS geometry definition. A look at the source code reveals the methods used for the import of the following wing (planform) parameters:

wing sweep The wing sweep is calculated as the angle between the line spanned from the quarter chord of the second wing segments root to the quarter chord of the last segments tip and the yz plane (pitch and yaw axis).

root chord The root chord is calculated as the distance between the leading and trailing edge of the second segments inner section.

tip chord The tip chord is calculated as the distance between the leading and trailing edge of the last segments outer section.

span The span is defined as twice the distance between the leading edges of the first segments inner section and the last segments outer section.

wing reference area The reference area of the wing is taken not from the wing shape, but from the reference nodes of CPACS.

wing position The wing position or more precisely the leading edge position of the wings root chord is determined as the leading edge position of the first wing segments inner section.

The following assumptions made by VAMPzero on the CPACS wing geometry can be extracted from the wing parameters import methods.

1. The calculation of the wing sweep assumes that the wing must consist of at least two segments.
2. The second segments inner section defines the root chord length.
3. The fact that the first segment is not considered for the wing sweep calculation could mean that it is assumed to represent the wing segment lying inside of the fuselage and thus should not be considered for determining the sweep angle.
4. The implementation also relies on the order of the segment nodes in the CPACS file. It should be noted that the CPACS schema does not force any order on the segments of a wing in a CPACS file. Thus the order can be arbitrarily shuffled, making the implemented method infeasible for such cases.
5. The outer section of the last wing segment defines the wing tip.
6. The wing is defined as half a wing with symmetry.
7. The inner section of the first segment lies on the wings symmetry plane.

All of these assumptions can be erroneous, depending on the definition of the wing geometry in the input CPACS file. The examplatory wing geometry of the NASA CRM already violates some of the assumptions considered by the conceptual tool.

2 The wing is defined starting at the fuselage intersection. Thus the second segment is not the first after the fuselage intersection as intended.

3 The first segment does not lie within the fuselage and thus should be considered for the sweep angle.

7 The inner section of the first segment does not lie on the wings symmetry plane, thus resulting in a wrong value for the wing span.

The violation of these assumptions leads to an abstraction of the wing shape which is inconsistent with the original shape. FIG 3 shows the result for applying VAMPzeros internal interpretation to the NASA CRM wing shape. It is

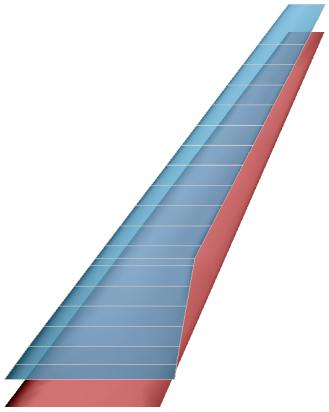


FIG 3: NASA CRM wing (blue) and its VAMPzero interpretation (red)

obvious that the simplified wing does not properly reflect the original wings position and span. When considering the fuselage shape this would result in a considerable difference in the exposed area of the wing. These two differences can mainly be explained by the offset of the

CRM wing from the symmetry plane. When aligning the two wings, as shown in FIG 4, the shapes compare quite well. The leading edge sweep, the root chord and the tip chord of

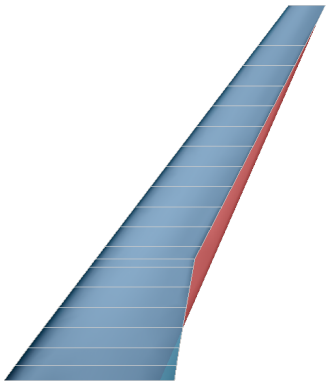


FIG 4: Aligned wing shapes of NASA CRM (blue) and its VAMPzero interpretation (red)

the simplified wing are reasonable when compared to the original values. After the alignment it also becomes more obvious that there is a difference in the wings planform areas. When comparing the calculated planform area of the NASA CRM wing shape with the reference area used for the aerodynamic coefficients, a difference of $24.01m$ can be observed ($S_{planform} = 359.68m^2$; $S_{reference} = 383.69m^2$ [14]) . This shows, that taking the reference area value, which is used for the normalization of aerodynamic coefficients, to represent the geometric wing planform area does not neccessarily reflect the wings planform area as calculated from the shape.

3.2. Achieving Consistency

There generally are two approaches in which the inconsistencies can be removed. The violated assumptions can either be fixed by changing the method to make valid assumptions or by providing an input for which the assumptions made are valid. Both methods have advantages and disadvantages. On the one hand, changing the data should be relatively easy to do, but doing so could have an undesirable impact on other parts of the process. Additionally, the results can differ depending on the changes in the data model and the changes have to repeated every time the process is used for a different configuration. On the other hand, changing the import method of the tool requires access to and familiarity with the code and thus can be more difficult. The main advantage of the second approach is, that as long as the changes to the import method are chosen carefully, using only assumptions valid for most expected input data, allows the method to work on a large variety of inputs. In the following paragraphs we will first take a closer look at the “Changing the Data” approach, followed by considerations on how the “Changing the Method” approach can be applied.

Changing the Data

The main issue of the wing shape is that it does not start at the symmetry plane. The discrepancy in the wing planform

(a) Method A (b) Method B

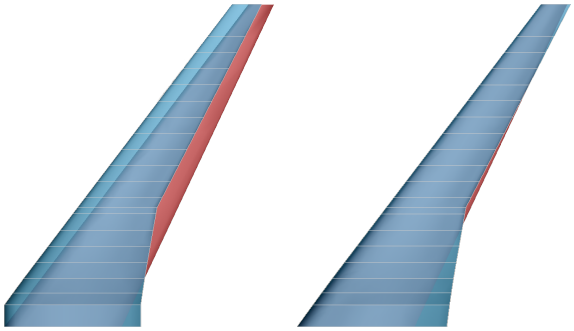
FIG 5: Extended wings

and reference area of the NASA CRM is that the win reference area is usually caclulated from a wing shape which is continued unti the symmetry plane. In the following the same wing object is represented by two approaches, which are commonly used to extract wing properties such as the reference area:

Method A The wing shape is continued by adding a rectangular segment with a straight/unswept leading and trailing edge.

Method B The wing shape is continued by extending the leading and trailing edges with the direction from the inner segment.

For method A the root section is duplicated, placed orthogonally on the symmetry plane and then connected with the original section. The wing shape resulting from this method is shown in FIG 5a.
Applying method B means continuing the leading and trailing edge shape of the inner segment towards the symmetry plane. From the intersection points of the symmetry plane with the extended edges the new root section parameters such as chord length and position are calculated. The resulting shape from applying method B is shown in FIG 5b.
Although these two methods represent the same wing product, the difference in representation has an impact on the interpretation of the conceptual tool. For both resulting shapes the assumptions violated with the original shape become valid. Applying VAMPzeros import methods to both of them results in the wing shapes shown in FIG 6. Although the resulting shapes of both wings are the the same, their position is not. Due to way in which the wing position is determined (as described above), the position of the wing created using method A does not well reflect the original wing position. This is obvious when comparing the leading edges of the wings. In contrast, the leading edges of the wing obtained using method B and its simplified shape align nicely.
When additionally changing the area in the reference node of the CPACS file the boeing wing can be consistently converted into the internal simplified representation as shown in FIG 7.
Since the changes on the input data need to be valid for all tools involved in the design process it might lead to problems with other tools. In this case the change in reference area affects the values of the aerodynamic coeffi-



(a) Method A (b) Method B

FIG 6: Comparing wing shape (blue) with VAMPzero interpretation (red)

cients. Therefore changing the import method is considered in the next paragraph.

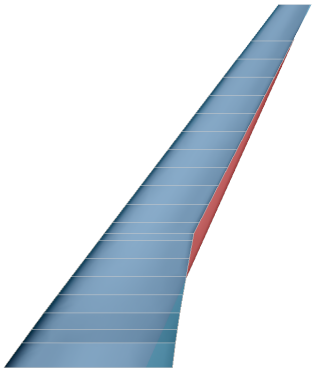


FIG 7: Consistent conversion of Method B wing (blue) to VAMPzero interpretation (red)

Changing the Method

Finding a generic approach for the simplification of the geometry which will derive a feasible interpretation for all possible inputs can be extremely difficult if not impossible. The use of assumptions can simplify the implementation of the conversion between two models considerably and hence is still commonly used. But it should always be assured that assumptions are only used where applicable. For the sake of confirmability the following points should be considered when implementing the conversion between two models.
Firstly, for the purpose of raising awareness for their existence, assumptions should be stated as explicit as possible. Secondly if the software is used for another use case it should be straight forward to change the assumptions, in order to provide flexibility. Separating the interpretation of the model from the application code supports both points, by enabling us to develop and implement an interpretation using case specific assumptions. Whenever a configuration does not satisfy all the assumptions made in one interpretation, it can be replaced by a another implementation suitable for the respective configuration. This way only the interpretation needs to be adapted or exchanged depending on the use case, but the main tool can stay the same - no need to provide multiple versions of the same tool.

When developing a complex design workflow by connecting multiple tools together there are many concerns to be taken care of. A central data format such as CPACS simplifies the connection of tools by providing a common language, but application specific “glue code” is still needed in order to ensure consistency between the different interpretations of the central model. Placing all the assumptions on the central data model in the “glue code” while simultaneously keeping the code as precise and short as possible, decreases the effort of adopting the process to new configurations and thereby increases its flexibility. In oder to support the fast implementation of required “glue code”, a method based on disciplinary concepts was developed serving as a bridge between the central data model and the application specific data model.

4. Concept Interface Layer

A common language such as CPACS is fundamental to a distributed multi-disciplinary design process including competences of many different partners. As evident from the example in the previous section, even when using a common data format as a central data storage, the consistency between the models is not automatically achieved. Usually the tools use the product description from the central data format and convert it into an internal representation of the data. When combining aircraft design and analysis tools into a design process, CPACS is but the first step. It ensures that the tools can read from the data structure regardless of the tool used to create the data and that they can always be fed with the most recent model. Though whether a connection of two tools is feasible also depends on the compatibility of their internal representation generated from the central model.
In order to create a feasible design process two main requirements need to be satisfied:

1. All used methods need to be in appropriate for the problem. (Appropriate internal model of the problem domain and appropriate physics.)

2. The collection of all assumptions made during the interpretation of the central data by all tools may not contain contradicting statements.

The first criteria is important for the selection of tools to be integrated into the design process. Using a supersonic aerodynamic analysis method for the design of a glider with a subsonic design mach number is usually not a feasible approach. The internal model of the supersonic analysis tool includes assumptions which are not in line with the problem at hand and thus the method is not compatible with the problem task.
The second criteria reflects the inconsistency example from the previous section. Checking this criteria is usually less straight forward. Ensuring it requires to keep track of all assumptions made by the involved tools when interpreting the central product model. For dealing with these less obvious compatibility issues a layered approach for the setup of a complex multi-disciplinary and multi-fidelity design process is presented which supports the user in avoiding hidden inconsistencies.

Accessing CPACS files can be done in many different

ways. Some of them are more comfortable than others. Since CPACS is based on XML (Extensible Markup Language) and thus a text based format, CPACS files could be accessed through plain text processing. But the use of an XML library especially for scripted access is far more convenient and less error-prone. In the same way XML libraries simplify the access to XML files, CPACS specific libraries can simplify the access to CPACS files and hence enable a more effective processing of CPACS data, which is desirable for the “glue code” as described in the previous section. For accessing CPACS data a layered approach as shown in FIG 8 was developed. One can find

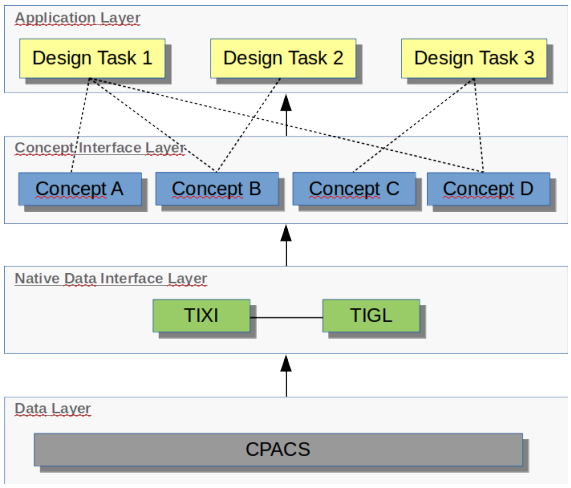


FIG 8: Concept Interface Layer

layered approaches in many complex fields, the OSI-model (Open Systems Interconnection Model) being particularly popular. The layered approach has proven valuable for separation of concerns and modularization of systems. Connecting an application to an external data format requires a mapping between the internal and external data model. Depending on the similarities or dissimilarities between the two representations, a more or less extensive conversion/interpretation is needed. The approach for connecting applications to CPACS can be described by the four layers shown in FIG 8.

Data Layer (DL) is the description of the model as a valid CPACS definition

Native Data Interface Layer (NDIL) provides access to the CPACS related properties

Concept Interface Layer (CIL) provides the abstraction from the CPACS model properties to the model representation properties ad-hoc for the application level

Application Layer (AL) is the description of the model as required by the application/tool

The layers can be passed through either way, top to bottom for converting from the application data to CPACS, or bottom to top for converting CPACS data into the applications internal representation.
At the bottom of FIG 8 the CPACS data format represents the data layer. It is the structure in which the data is described. For CPACS the structure is the schema definition.
On top of the Data Layer there is the Native Data Interface Layer. It consists of accessor libraries which provide direct access to the elements of the data structure such as TIXI (TIVA XML Interface) and TIGL (TIVA Geometric Library). They are used to directly access the data as represented in the CPACS definition.
At the top of FIG 8 there is the Application Layer. It represents the place where the application codes or wrappers to the applications are implemented. For VAMPzero and the previous example this would be the parameters for the single trapezoid wing representation.
Inbetween the Native Data Interface Layer and the Application Layer, the Concept Interface Layer serves as a bridge connecting the two. Here the disciplinary concepts are defined which serve as abstractions from the data structure for more intuitive interface enabling a more straight forward implementation of knowledge. To this day there is no way for an application to include the real world with all of its details and presumably there never will be. Applications consist of knowledge which is described using simplified concepts for real world objects or relations. Common to all simplification approaches is, disregarding everything which is not relevant for solving the problem. Which parts of the real world are actually relevant highly depends on the problem domain or discipline. Therefore the concepts used are usually tailored to the problem domains. When two problem domains are related they may use similar concepts but similar does not necessarily mean that they are compatible. In some cases the concepts even uses the same named parameters and thus seem to be the same but due to differences in their definition they actually are not. One example is the wing span of a c-wing as described in [7]. When assembling tools into a complex design process using similar but not identic concepts, some consistency issues can easily pass unnoticed. Often without beeing aware of it such processes contain inconsistent models, which can have an unforseen effect on the result. When using VAMPzero for designing the horizontal tail plane, an error in wing position and hence in its mean aerodynamic center results in an error in lever and by that in tail size. Therefore a clear and explicit definition of the concepts used in each sub process/tool is necessary in order to provide a way to uncover such discrepancies thus leading to more consistent design processes.
At DLR Air Transportation Systems the cpacsPy library serves the purpose of the intermediate Concept Interface Layer for CPACS based systems. A clear definition and adequate documentation of the concepts are essential to keep track of the assumptions made for the concept parameters. Details on the cpacsPy and the definition and application of concepts are given in [7]. In the next section an example is given on how the concepts from the cpacsPy are used to create an abstraction module as a solution for the consistency problem presented in section 3.

5. Resolving Inconsistencies

For the preprocessor in Digital-X the inconsistencies between the detailed NASA CRM wing in CPACS (Data Layer) and VAMPzeros wing representation of a single trapezoid wing (Application Layer) were solved by implementing an abstraction script linking the properties of the two representations/models. Before turning to the implementation,

the method for abstracting a single trapezoid wing from a detailed CPACS wing geometry needs to be defined in a generic way. Therefore some decisions have to be made. Firstly it is decided, that the properties of the simplified wing should be defined by the exposed wing shape. This way the abstractions of the method A wing, the method B wing and the original NASA CRM wing shape should result in the same single trapezoid wing. Secondly it should be possible to exclude some parts of the detailed wing for the determination of the parameters. This is required because for some of the parameters, such as the taper ratio, there is no single distinct/universal definition on a complex wing shape. Especially for rounded of wing tips as shown in FIG 9 the tip chord can only be estimated. Thus defining a taper ratio parameter which behaves as expected for any arbitrary wing shape can be extremely difficult. One could try defining rules for identifying the wing tip chord such as:

FIG 9: Rounded wing tip

- wing tip location should be within the outer 10% of the wing span.
- When the relative change in chord length with respect to spanwise position is greater 20% (or another threshhold).

This might work well for conventional wings but for boxwing aircraft this will probably not work as expected, depending on e.g. whether it is defined as a single wing or three separate wings. On the one hand the exclusion of some parts of the wing allows to ignore some of the details resulting in a more simple and clean definition of the parameters, while on the other hand requiring additional user input on the start and end segments for the definition of the parameters.

The wing concept from the wingAeroShape.wingLib module of the cpacsPy library includes the option to give identifiers for the start and end segment for the determination of the wing parameters. It can be used to calculate mean values for the most common wing parameters such as sweep and determine the wing reference area, aspect ratio and taper ratio for parts of the wing starting from one segment following the topologic tree until the end segment is reached.
Based on the above mentioned considerations, the abstraction script is developed using a wing concept from the cpacsPy. We will walk through the process step by step. At first the aspect ratio, the taper ratio, the sweep, the dihedral and the reference area are calculated using the respective methods from the wing concept by providing the start and end segment of the exposed wing. The parameters define the exposed part of the single trapezoid wing as shown in FIG 10. The corresponding values for the wing on the CIL are presented in Table 1. This wing is a representation of a wing using the “Wing” concept provided by the *wingAeroShape.wingLib* module of the cpacsPy. For positioning, the simplified wing is placed in a way that the x-position of the mean aerodynamic center of the exposed wing is met. From the spanwise position of the root chord (of the exposed wing) and the sweep angle of the leading and trailing edge, the intersections with the symmetry plane and from that the actual root chord is calculated. In a next step the taper ratio of the complete single trapezoid wing can be calculated from the root and tip chord. Also the complete wing planform area needs to be calculated. The consistent parameters for the application wing shape are given in Table 1 and pictured in FIG 11. This wing shape is the representation of the wing on the application level.

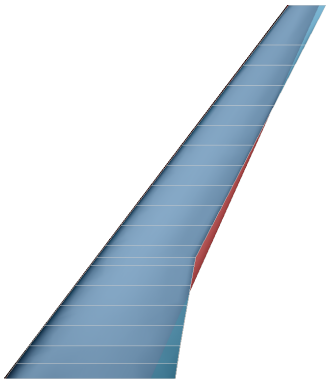


FIG 10: Comparison of exposed NASA CRM wing (blue) and wing at CIL (red)

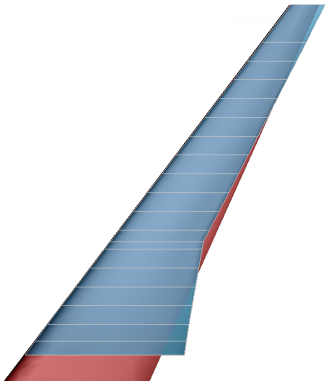


FIG 11: Comparison of the NASA CRM wing (blue) and the wing on AL (red)

Running the abstraction for the geometry from both method A and method B result in the very same wing shape.
Using the concept provided by the cpacsPy, developing a consistent abstraction of the NASA CRM wing for use in VAMPzero was relatively simple. All assumptions for the abstraction are explicitly stated in the short script and the documentation of the used concept. The implementated abstraction is expected to be generic enough to be valid for most anticipated inputs. If at some point this should not be the case, it can be easily adapted.

Of course the implemented method could be imple-

TAB 1: Parameter values for simplified wing shapes

Parameter	Wing at CIL	Wing at AL
Aspect Ratio	8.34	8.55
Taper Ratio	0.22	0.24
Planform Sweep	$37.2°$	$37.2°$
Dihedral	$4.7°$	$4.7°$
Reference Area	$359.7m^2$	$407.5m^2$
Root Chord	$10.8m$	$11.4m$
Tip Chord	$2.4m$	$2.4m$
X Position (Mean Aerodynamic Center)	$34.4m$	$33.5m$
X Position Root Chord Leading Edge	$24.3m$	$22.8m$

mented in VAMPzero but by keeping it separate from the tool, it can be reused for other applications and used for communicating assumptions. When using the presented method for setting up a complex design process, the assumptions made become far more visible and thereby increase awareness/conciousness for them. Another benefit is that the abstraction can be easily adapted or exchanged if required for a new configuration. No additional version of VAMPzero and no deeper knowledge of VAMPzeros code basis is required.

6. CONCLUSIONS

In collaborative environments data needs to be exchanged frequently between the participants. Tools from different disciplines need to be combined into a complex design process. When developing/implementing a collaborative design process as a combination of disciplinary tools, consistency in the used methods needs to be ensured. In fact not only the consistency between the methods, but also consistency in the assumptions on the data exchange of all involved tools needs to be guaranteed. Using a central data format can simplify the data exchange by reducing the number of interfaces required for collaboration but it does not automatically lead to consistency between the model representation of the involved tools. It can generally be said, that every separate interpretation of data can lead to inconsistencies between models. Therefore using a central data format such as CPACS is not enough. In this paper an example for inconsistencies due to assumptions made while converting the central data model into an application specific representation was given. It underlined the importance of beeing aware of all the assumptions made when exchanging data with the central data model.
As a solution an additional abstraction layer, the Concept Interface Layer, was introduced. For the CPACS environment this layer was realized as part of the cpacsPy library, where disciplinary concepts are implemented and documented. It was shown how the use of concepts from the cpacsPy supported the development of a interpretation of the central data model consistent with the actual configuration at hand. Although this approach requires some additional information, e.g. valid assumptions for the central model, it is the authors opinion that explicitly stating all assumptions is essential for the setup of a truely consistent design process. Admittedly this seems to increase the effort of interfacing between the different disciplinary models, but when provided with a suitable disciplinary concept the effort is actually very limited and the confidence in the process is increased.
Therefore extending CPACS as a central data model by a library for common, disciplin specific interpretation of the data is pursued in form of the cpacsPy library. Supporting a common understanding through the explicit definition and use of concepts is assessed to be of high value for the CPACS user community. With the development of the cpacsPy library the first step towards an enhanced common language has been established.

REFERENCES

[1] BÖHNKE, D. ; MOERLAND, E. ; SEIDER, D. ; KUNDE, M. ; LITZ, M. ; ZIEMER, S. ; STENZ, G.: Challenges for collaborative data management in an MDAO process. (2012). – Deutscher Luft- und Raumfahrtkongress (DLRK)

[2] CIAMPA, P.D. ; NAGEL, B.: Preliminary Design for Flexible Aircraft in a Collaborative Environment. In: *4th CEAS Air & Space Conference* (2013), S. 270–281

[3] DORBATH, F. ; VEEN, L. van ; GAIDA, U.: Wing Secondary Structure - Large Civil Jet Transport (MTOM > 40t) - Statistical Mass Estimation. In: *Luftfahrtechnisches Handbuch (LTH)* (2012)

[4] GÖRTZ, S. ; ILIĆ, Č. ; ABU-ZURAYK, M. ; LIEPELT, R. ; JEPSEN, J. ; FÜHRER, T. ; BECKER, R. ; SCHERER, J. ; KIER, T. ; SIGGEL, M.: Collaborative Multi-Level MDO Process Development and Application to Long-Range Transpoart Aircraft. In: *28th International Congress of the Aeronautical Sciences* (2016)

[5] HAFER, X. ; SACHS, G.: *Flugmechanik: Moderne Flugzeugentwurfs- und Steuerungskonzepte*. 3. Springer-Verlag Berlin Heidelberg, 1993. – ISBN 978–3–642–86730–9

[6] ILIC, C. ; JEPSEN, J. ; LIEPELT, R. ; LEITNER, M. ; SCHUSTER, A. ; SCHERER, J. ; BECKER, R.-G. ; SEIDER, D. ; WUNDERLICH, T. ; KEYE, S. ; BACH, T. ; STOLLENWERK, T.: Demonstration of a Collaborative Multi-Level MDO Process on a Long-Range Transport Aircraft. (2016). – Deutscher Luft- und Raumfahrtkongress (DLRK)

[7] JEPSEN, J. ; CIAMPA, P.D. ; NAGEL, B. ; GOLLNICK, V.: Design of a Common Library to Simplify the Implementation of Aircraft Studies in CPACS. (2015). – DLRK

[8] KAISER, C. ; FRIEDEWALD, D. ; QUERO, D. ; NITZSCHE, J.: Aeroelastic Gust Load Prediction Based on Time-Linearized RANS Solutions. (2016). – Deutscher Luft- und Raumfahrtkongress (DLRK)

[9] KUNDE, M. ; SCHREIBER, A.: Advantages of an Integrated Simulation Environment. In: *4th CEAS Air & Space Conference* (2013)

[10] LEITNER, M. ; LIEPELT, R. ; KIER, T. ; KLIMMEK, T. ; MÜLLER, R. ; SCHULZE, M.: A Fully Automatic Structural Optimization Framework to Determine Critical Design Loads. (2016). – Deutscher Luft- und Raumfahrtkongress (DLRK)

[11] MARTINS, Joaquim R. R. a. ; LAMBE, Andrew B.: Multidisciplinary Design Optimization: A Survey of Architectures. In: *AIAA Journal* 51 (2013), Nr. 9, 2049–2075. http://dx.doi.org/10.2514/1.J051895. – DOI 10.2514/1.J051895. – ISBN 10.2514/1.J051895

[12] NAGEL, B. ; BÖHNKE, D. ; GOLLNICK, V. ; SCHMOLLGRUBER, P. ; RIZZI, A. ; ROCCA, G. L. ; ALONSO, J. J.: Communication in aircraft design: Can we establish a common language? In: *28th Congress of the International Council of the Aeronautical Sciences* (2012)

[13] SCHUSTER, A. ; SCHERER, J. ; FÜHRER, T. ; BACH, T. ; KOHLGRÜBER, D.: Automated Sizing Process of a Complete Aircraft Structure for the Usage within a MDO Process. (2016). – Deutscher Luft- und Raumfahrtkongress (DLRK)

[14] VASSBERG, J. C. ; DEHAAN, M. A. ; RIVERS, S. M. ; WAHLS, R. A.: Development of a Common Research Model for Applied CFD Validation Studies. In: *26th AIAA Applied Aerodynamics Conference* (2008)