TOWARDS A METAMODEL FOR CONCEPTUAL AIRCRAFT DESIGN

Martin Glas
Bauhaus Luftfahrt

Keywords: *conceptual aircraft design, model integration, metamodeling*

Abstract

The development process of aircraft poses the challenge to consistently perform optimizations and assessments of new aircraft concepts in a multidisciplinary design space. Therefore, it is desirable to maintain a consistent and coherent aircraft model throughout the development process which integrates all involved disciplines. However, the conceptual aircraft model is usually divided into different incoherent model segments specialized on the involved disciplines.

This paper identifies several problems that are associated with this situation and proposes an approach for model integration based on a metamodel.

1 Introduction

The design of aircraft concepts is a multidisciplinary process which aims at incorporating state-of-the-art technology and advances in multiple fields of research into new product concepts at an early stage of the development process. Each discipline uses its own tools, methods and experience working on its own model which covers a certain aspect of the design.

One of the challenges of a multidisciplinary design process is to divide the overall aircraft model into manageable segments enabling the application of specialized knowledge. However, the scopes of the disciplines involved in the conceptual design of an aircraft overlap. Accordingly, the respective model segments are inevitably correlated. This means that items of different model segments are coupled or represent the same thing. When model segments are refined in a distributed development process, they are not changed coherently, since there is no automated update mechanism between them. Keeping the model segments consistent and maintaining a common perception of the product is thus another challenge.

Model-based conceptual design promises to increase the effectiveness of the development process and quality of the final product concept. To fully realize the potential of this approach, a continuous integration process is required. Current practice comes short of implementing such a process.

This paper addresses the problem of model segmentation and proposes a metamodel-based solution to the problem of integrating several model segments and is organized as follows: Section 2 describes the problem of integrating multidisciplinary model segments which is illustrated in Section 3 by exemplary scenarios. Section 4 gives a brief overview of the existing literature concerning model integration. Section 5 describes the requirements of a solution and gives a draft for a metamodel which could implement an important contribution to a solution. Furthermore, it is discussed in Section 6 how far the proposed approach can contribute to solving the problem. Section 7 gives a summary and an outlook on further research activities.

2 Problem Definition

During the development process the assessment of new design variants requires an integrated model of the overall aircraft in order to avoid biased and inconsistent design decisions. In a multidisciplinary design process the aircraft is however modeled in several distributed model segments, specific to a discipline. In order to achieve consistency throughout the project, the distributed models have to be integrated. Integration means that changes or refinements of model items are disseminated between model segments. During the integration process the relationships between model items have to be carefully analyzed and made explicit. This analysis is a prerequisite for the decision whether a change or refinement of one model is propagated to the other model or not.

Figure 1 illustrates four basic types of relationships between models. Model segments can overlap (O) which means that they both contain items which represent the same thing. Furthermore, model segments may not overlap but exhibit explicit couplings (EC) between items. There may also be hidden couplings (HC) between model segments which are not registered by the integration process. From the point of view of the overall aircraft model there also may be orphaned items (OI) which are not covered by any model segment. These different kinds of relationships show that the integration process is a complicated and error-prone task.

Especially in order to avoid orphaned items and hidden couplings it is a prerequisite of the integration process to reveal all correlations between models. In detail, this means to identify commonalities and classify the differences on a meaningful level in order to support the change decision process of the respective model owners.

Figure 2 illustrates four types of differences which may occur during a correlation of model *A* and *B* and the required decision by the model owners. In a hierarchical process of integration model *A* stands for any model segment which has to be integrated into the central model *B*. If a central model does not exist, *A* and *B* stand for a pair

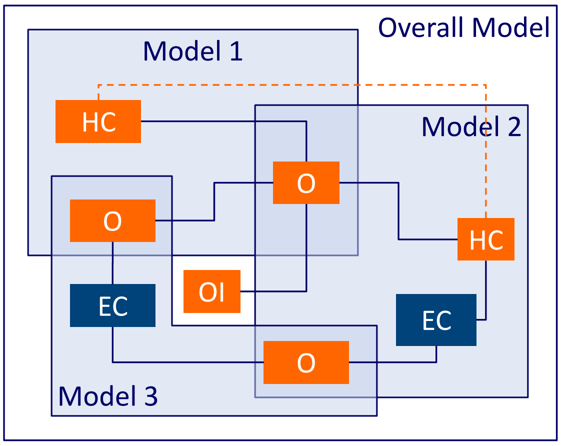

Fig. 1 Model segments show overlapping items (O), explicitly coupled items (EC), hidden couplings (HC) and orphaned items (OI).

of model segments on an equal hierarchical level. The differences between *A* and *B* can be classified in the following types:

Addition An item in model *A* is not contained in model *B*. The reason may be that the element has not been relevant for the focus of model *B*. The owner of model *B* has to decide whether to adopt the item.

Refinement An item in model *A* represents a refinement of an item in *B*. Model *B* may not have contained that refinement, because the level of detail for that particular aspect was not required. The owner of model *B* has to decide whether to adopt the refinement.

Update An item is contained in both models *A* and *B* representing the same thing but in different versions. The owner of model *B* may have committed a change on his model, e.g. a design decision. The owners of *B* has to decide whether to update on the newer version of model *A*.

Alternative An item in model *A* may represent an alternative to an item in model *B*. It is one goal of conceptual aircraft design to trade off design variants. Therefore, the correlation process has to discriminate between alternatives and different

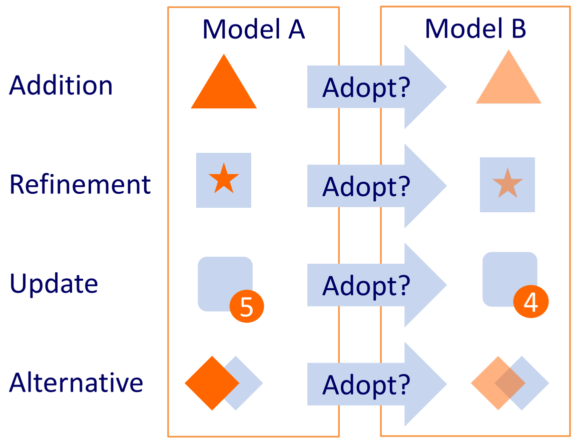

Fig. 2 Types of model differences and the associated decisions which can occur during the integration of Model A and B

versions of the design. If model *A* has generated a plausible design alternative, the owner of model *B* has to decide whether to adopt the alternative.

However, a meaningful level for revealing these correlation between the models is difficult to achieve.

Models usually have different structures and levels of detail. These differences stem from specific paradigms and different practices to implement refinements used by each discipline.

In addition, disciplines have different frequencies for delivering results. In a long meantime between the delivery of results, other model segments may already have changed several times. If the results are based to great extent on an outdated development stage they become obsolete.

More fundamental differences come from peculiar methodologies and concepts of the respective disciplines. This means that the disciplines can have different terms and approaches describing the same thing. Without a common ground of terminology and methodology common model items are hard to correlate in a formal and automatable way.

Furthermore, there are aspects of the aircraft design which are not covered by the specialized model segments but are specific to a general view on the overall aircraft concept. Therefore, there may be no owner for these segments of the model during the multidisciplinary design process who keeps these segments updated.

Without a clear concept for how a meaningful level of correlation is achieved, the integration of models has to rely strongly on manual work, which hardly scales.

As a consequence, the lack of automation and decision support may lead to less frequent integrations which aggravates the problem of inconsistency. Significant parts of the models are not reused by other disciplines but have to be implemented redundantly.

3 Examples

The following section illustrates the described problem by concrete examples. Particularly, it depicts situations where differences between models with respect to structure, level of detail, frequency of results and methodology occur and hinder the integration process.

Conceptual aircraft design starts with the definition of a general system architecture and an initial sizing process. Usually this step is done by a relatively small group of designers using a general purpose conceptual design tool.

After this stage of development the overall model is adopted by several groups focused on certain disciplines like propulsion, structure, aerodynamics and stability&control. These groups develop their own models and manage consistency between each other by means of predefined interfaces and coordination processes. These interfaces are usually based on the initial system architecture and prior experience from earlier developments. This is done to give the design groups freedom in solving the problem without interfering too much with other groups.

However, an integration process based on predefined interfaces may not be sufficient to achieve consistency between models which are continuously refined and restructured. Each discipline decomposes the general layout slightly differently. For example aerodynamics is focused on the outer shape of an aircraft, whereas weight

& balance lays emphasis on the inner construction of an aircraft. The different structures and levels of detail may obscure overlaps and couplings between the models beyond the predefined interface, especially if an unconventional design is pursued.

Another problem arises from different frequencies of the disciplines to deliver results. Depending on the number of degrees of freedom and the need to use numerical methods, this time frame may differ significantly. Therefore, new overall aircraft performance calculations are available more frequently than results from aerodynamics which strongly require the application of time consuming numerical calculations. The different time requirements for calculations alone lead to asynchronous development increments.

Another example are the differences between design of the propulsion systems and the rest of the aircraft. These differences are not limited to scale and level of detail. Whereas an aircraft is analyzed applying principles of structural mechanics and aerodynamics, engine design is based on modeling thermodynamic processes. These differences in the methodology make it very difficult to reveal the couplings and overlaps between models. Usually, the differences are big in such a way that the conceptual design processes of an aircraft and its engine are usually separated from each other.

Generally, the process of correlating the models is usually not supported by tools but relies on project management and the experience of the designers. Advances in supporting a semi-automated integration could help to realize the potential of the multidisciplinary design process.

4 Related Work

Literature about model integration in the context of multidisciplinary development is mostly centered on tool integration.

There is a general overview of the current situation of tool integration in the aircraft industry [10, 9]. Another example for a multidisciplinary integration platform is given from the automotive industry by [4].

Different design patterns for tool integration architectures based on a common domain specific modeling language are described in [6].

The work of [7] depicts a concrete software solution for model based version control which is based on a metamodel[1]. The metamodel in this solution is an abstract description of a class of models and specifies with what elements a model is build, and how these elements can be interrelated to each other. In this work the metamodel also specifies how a model is changed.

A similar concept is described in [8]. This work also incorporates a concept to manage models based on different metamodels and different ontologies. An ontology in information science is a set of concepts of a certain domain of knowledge which are connected by explicit interrelations. Ontologies are used in information science for capturing domain knowledge and for mapping different nomenclatures of models to each other.

In [2] it is stated that current tool chains lack a deep integration of models which hampers reuse and refinement of models. Accordingly, subsequent problems arise like redundancy, inconsistency and lack of automation. The concept of an integrated tool environment must address the different methodologies of the contributing disciplines in order to provide a useful tool to the designer. Therefore, a general framework is proposed to integrate the tool specific aspects on one dimension and methodology specific aspects on the other.

5 Model-based Integration

Tool integration solutions are specific to the respective tool suite of an organization. Tool integration approaches are usually concentrated on mapping the different data models and programming interfaces. In contrast to that, model-based integration is more driven by the context of the subject matter and more focused on the differences between models than on differences between tools. Therefore, this approach is especially adequate for the conceptual design

[1]Greek: $\mu\varepsilon\tau\acute{\alpha}$ = above, beyond

phase which needs to handle topology variations, which are difficult to address by tool integration. Model-based integration requires deep integration of tools in order to facilitate the process. However, this is not in the focus of this paper.

Literature suggests that every technical solution for model integration requires a metamodel which acts as the formal common basis for the integration process. The following section specifies a model-based integration architecture, the requirements of such a metamodel and presents a proposal how it could look like.

5.1 Model-based Integration Architecture

Literally, a metamodel is a “model of a model”. It is a formal specification of how a model is built. If a model is considered a graph then a metamodel describes model elements (nodes) and model links (edges) the graph can be built of. Additionally, a metamodel specifies rules which limit the number of allowed combinations of model elements, and constrains the number of allowed changes to a model.

The importance of a metamodel for the integration process is emphasized by the following architecture of a software system which supports the integration of models. The second important component of this architecture is an integration workbench.

The architecture assumes the existence of several model segments which conform to a metamodel, i.e. they are instances of the same metamodel. One of the models can be the central master model which integrates all other models. Figure 3 depicts the situation when model segments *A* and *B* are integrated. Each model is a data source for the integration workbench. This workbench analyzes the models automatically, helps the model owners to reveal the correlation between the items of the different models and provides support for model integration decisions. After the model owners have determined the correlations between the models and have drawn their change decisions which are supported by the integration workbench, the changes are propagated to the respective model.

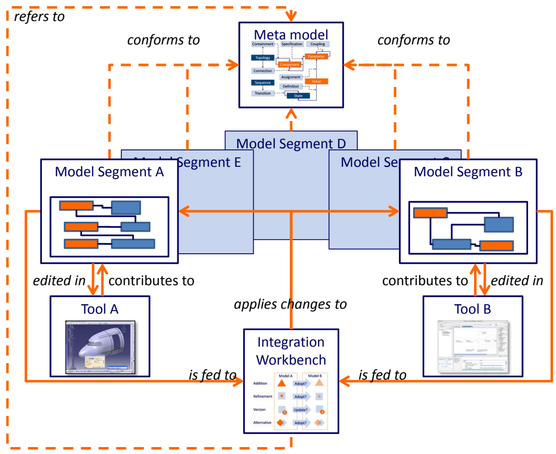

Fig. 3 The general layout of the model integration architecture. The diagram depicts the situation when Model A and model B are integrated. The other model segments can be integrated in the same manner as they all conform to the common metamodel.

This architecture and the associated process are not the only solution for an integration architecture. Section 4 refers to several other approaches. However, a metamodel is an important component used in all approaches and especially supports the semi-automated process in the integration workbench described above. The next section will propose a draft for the metamodel, whereas the concept of an integration workbench will not be explained in more detail.

5.2 Requirements and Draft for a Metamodel

In this section requirements and a draft for the metamodel will be presented. The metamodel needs to fulfill the following requirements in order to address the issues identified in Section 2.

Structure and level of detail The metamodel must provide a standard mechanism for the decomposition of models independent from a discipline specific paradigm.

Versions There must be an explicit standardized way to express the history between versions of a model.

Methodologies The metamodel must define a set of allowed elements and links which enable every discipline to express their specific concepts by a common language.

The draft for the metamodel for conceptual aircraft design is structured as follows:

Model Elements The metamodel specifies different types of node-like items in the model like *component*, *parameter* and *value*.

Model links The edges between model elements are called model links like *containment*, *connection*, *specification*, *assignment*, *definition* and *transition*.

Collections Model links can be accumulated in collections like *topology* (collection of connections and containments), *state* (collection of assignments), *sequence* (collection of transitions) and *component* (collection of collections including model elements).

Constraints Restrictions can be applied on model links to limit them on reasonable combinations between model elements.

Transactions Models can be manipulated by defined transactions like *creation*, *modification*, *deletion* and, if applicable, *rerouting* of model links. In contrast to transitions which interrelate two states of the modeled system to each other, a transaction interrelates two consistent versions of the model to each other.

This paragraph gives a more detailed description of the constituent parts of the metamodel. A *component* represents a physical system. A component can be *connected* to other components and can *contain* other components. The set of all containments and connections of a component is its *topology*. Additionally, a component can have defining attributes called *parameters*. Parameters of a component can be connected by a *coupling*.

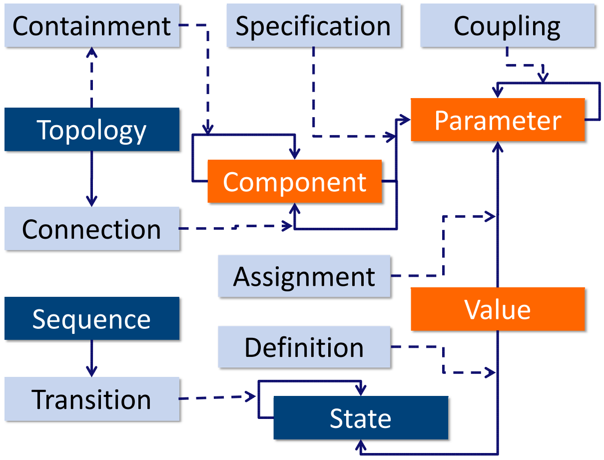

Fig. 4 Sketch of the metamodel in a UML-like notation

The coupling can be expressed by a mathematical expression. A coupling across the boundaries of a component can only be realized via connections and compositions which act as ports. Other rules may be applied to the coupling of parameters in order maintain model consistency, which are not covered in in this paper. Components can be in a certain *state*. A state defines a set of parameters, the so called state variables by certain *values*. States are linked by *transitions*. There can be more than one transition from one state. A specific order of transitions is defined by a *sequence*.

Links between model elements can be subject to *constraints*. For example a constraint can limit the values which can be assigned to a parameter to a certain range. Constraints can also limit the number of component types, which can be connected to another component.

Model *transactions* specify allowed operations on the model. Model elements can be created, modified and deleted. Model links additionally require rerouting. These transactions can also be constrained. For example, a constraint on the modification transaction can express that a modification of a value must lead to a convergent system, otherwise the modification is revoked.

Components	Parameters	...
Aerodynamics		
Profile, shell, fluid ...	Lift, drag, angle of attack, aspect ratio, taper ratio, chord length, lifting area ...	
Thermodynamics		
Compressor, turbine, combustor, propulsor ...	Enthalpy, entropy, SFC, temperature, pressure ratio, efficiency ...	
Structural Mechanics		
Rib, stringer, shell ...	Stress, pressure, stiffness, load ...	
...		

Table 1 Exemplary proposal for domain specific concepts which can be derived from the metamodel.

5.3 Potential Application of the Metamodel

Generally, a metamodel is not applied directly but serves as a formal framework for building models. These models can be automatically validated against the metamodel to check conformity.

In order to make the modeling process more efficient each discipline can derive its own discipline-specific framework from this metamodel in order to provide the designers with a useful toolkit. This discipline-specific framework can be realized as an instantiation of the metamodel. The instantiation mechanism makes sure that models using the model specific frameworks automatically conform to the metamodel. Table 1 gives examples for discipline-specific instances of the metamodel.

Beside the application of a metamodel for modeling clients, a metamodel can be the basis of a version control service as described in [7]. Figure 3 gives an impression how an integration workbench application uses such a metamodel-based version control service.

6 Discussion

The previous section described a draft of a metamodel for conceptual aircraft design as an important component for a model integration architecture. It has been stated that the metamodel especially supports the task of model correlation as all models use the same primitives. The following section will discuss how far the metamodel actually contributes to solving the issues of model integration and which aspects of model integration are not yet encompassed by the metamodel.

6.1 Structure and Level of Detail

According to the requirements, the metamodel should provide a general mechanism for the interconnection and decomposition of model elements in order to support a common structure and the correlation of different levels of detail.

The *containment* relationship between components is the central mechanism of model decomposition, as the component is the only element which can contain elements of one's own kind.

Additionally, components play an important role in the reuse of model parts. As components have defined connections to siblings (same level components), the parent (containing component) and children (contained components), components can be extracted from the model and used as building blocks for other models. The metamodel specifies that cross component couplings have to be realized via connections or compositions acting as ports.

However, a common mechanism for decomposition does not necessarily mean that different disciplines will decompose a system in the same way. For example one discipline may decompose a wing into three components whereas another discipline decomposes it into two. Therefore, a common mechanism for decomposition and modularization may still require the manual correlation and adaption of model items.

6.2 Versions

As the different disciplines usually do not produce new versions synchronously the metamodel must provide means to correlate different versions of the models in order to integrate them.

The proposed metamodel provides different types of allowed transactions on the model. A version is a set of such transactions which define the transformation from the old to the new version.

An unambiguous correlation between two models which represent different stages of development requires a common ancestor of both models and continuous tracking of transactions on both sides. Furthermore, the handling of design alternatives is not yet covered in the metamodel.

6.3 Methodologies

The metamodel should enable the correlation of models which are refined and developed following a specific methodology. The proposed metamodel aims at supporting the modeling of a common subject matter of all disciplines contributing to conceptual design – the physical aircraft. It is assumed that every discipline has its own concepts and methodologies to analyze and design, but considers the physical system as its basis. This implies that every discipline has a bidirectional mapping of the discipline-specific view of the system to the physical system view.

It seems to be current industrial practice to use a geometry-based model as the common master model for all disciplines [10]. This practice is an indication that it is a viable approach to use a physical system oriented model as a common structure for different methodologies as the geometry model is closely related to the physical system.

7 Conclusion and Outlook

This paper described the problem of model integration in a multidisciplinary design process. The correlation of different models was presented as the critical task of the integration process to identify overlapping between models and to classify differences between models on a meaningful level. Furthermore, the concept of a metamodel was presented as an important contribution in solving the problem. It was discussed in how far the metamodel can help to unify the structure of models, to correlate different versions of models and to provide a common framework for the various concepts and methodologies of the different disciplines involved.

As a next step the metamodel will be implemented. Based on this implementation, modeling exercises can give evidence if aircraft concept designers perceive a modeling environment based on a common metamodel as a practical tool. Furthermore, such exercises can elicit requirements for the refinement of a model-based integration environment which helps to cross-fertilize the disciplines in the development of aircraft concepts towards an true interdisciplinary process.

References

[1] Brinkkemper S, Saeki M, and Harmsen F. Metamodelling based assembly techniques for situational method engineering. *Information Systems*, Vol. 24, No 3, pp 209–228, 1999.

[2] Broy M, Feilkas M, Hermannsdoerfer M, Merenda S, and Ratiu D. Seamless model-based development: From isolated tools to integrated model engineering environments. *Proceedings of the IEEE – Special Issue on Aerospace & Automotive, preprint*, 2010.

[3] Dufresne S, Johnson C, and Mavris D. Variable fidelity conceptual design environment for revolutionary unmanned aerial vehicles. *Journal of Aircraft*, Vol. 45, No 4, pp 1405–1418, 2008.

[4] El-khoury J, Redell O, and Torngren M. A tool integration platform for multi-disciplinary development. *Proc Proceedings of the 31st EUROMICRO Conference on Software Engineering and Advanced Applications*, pp 442–450, 2005.

[5] Emerson M and Sztipanovits J. Techniques for metamodel composition. *Proc 6th OOPSLA Workshop on Domain-Specific Modeling (DSM'06)*, p 123, 2006.

[6] Karsai G, Lang A, and Neema S. Design pat-

terns for open tool integration. *Software and Systems Modeling*, Vol. 4, No 2, pp 157–170, 2005.

[7] Kögel M. Towards software configuration management for unified models. *Proc CVSM '08: Proceedings of the 2008 international workshop on Comparison and versioning of software models*, pp 19–24, New York, NY, USA, 2008. Acm.

[8] Kramler G, Kappel G, Reiter T, Kapsammer E, Retschitzegger W, and Schwinger W. Towards a semantic infrastructure supporting model-based tool integration. *Proc GaMMa '06: Proceedings of the 2006 international workshop on Global integrated model management*, pp 43–46, New York, NY, USA, 2006. ACM.

[9] Ledermann C, Hanske C, Wenzel J, Ermanni P, and Kelm R. Associative parametric CAE methods in the aircraft pre-design. *Aerospace Science and Technology*, Vol. 9, No 7, pp 641–651, 2005.

[10] Pardessus T. Concurrent engineering development and practices for aircraft design at airbus. *Proc Proceedings of the 24th ICAS Conf., Yokohama, Japan*, 2004.

[11] Santhanakrishnan D, Parks G. T, Jarrett J. P, and Clarkson P. J. An integrated evaluation system for the conceptual design of space systems. *Proc CSER' 09: Proceedings of the 7th Annual Conference on Systems Engineering Research*, 2009.

[12] Tam W. Improvement opportunities for aerospace design process. *Proc Space 2004 Conference and Exhibit, San Diego, CA*, pp 1405–1418, 2004.

7.1 Copyright Statement