EVALUATION OF MODELING LANGUAGES FOR PRELIMINARY AIRPLANE DESIGN IN MULTIDISCIPLINARY DESIGN ENVIRONMENTS

D. Böhnke*, M. Litz†, B. Nagel*, S. Rudolph‡

Abstract

To enhance the quality of design processes the collaboration and integration of disciplinary specialists into a holistic, multidisciplinary design team must be granted. The resulting demand for communication leads to the desire for a common language or namespace for the exchange of information. As the collaboration includes the capability to link analysis tools, communication must also be established on the level of computer code. Due to the fact that the number of interfaces is the critical factor for communication the common language must be coded in a single information model. The key to a multidisciplinary approach is therefore a central model that contains all data and is accessible for all domains. The goal of this work is to analyze three different models in respect to their benefits for information exchange in preliminary airplane design. These models are CPACS, developed at the DLR, STEP, published by the ISO, and the UML, released by the OMG. Several methods for the classification of information are introduced and requirements for information models are set up. The main part of the work is contributed to the analysis of the different models. Subsequently, a prototype for a converter tool, that processes CPACS data to STEP is developed. Additionally, a conversion from the CPACS modeling language XSD to UML is shown.

1 INTRODUCTION

The coupling of physical effects and the rising complexity of modern aircraft necessitate an intense collaboration of disciplinary specialists in preliminary airplane design. Additionally, a growing number of suppliers and outsourced design activities aggravate the design processes even further. Novel multidisciplinary design environments intend to enable specialists to better integrate analysis codes and constitute the consistent technical basis for their, often dislocated, cooperation. In this manner different technical aspects can be evaluated quickly, since the data update and analysis capabilities are available throughout the design team. The exchange of information is therefore of crucial importance in multidisciplinary design.

Due to the fact that in data exchange the number of interfaces is the critical factor for the flexibility of a design environment, a central information model is a key feature. The central information model reflects among other things the common namespace of the design team and can be seen as the meta-model for all of the deduced analysis models. The architecture of such a novel design environment is strongly linked to principles of model-based architectures and the chosen underlying software engineering techniques influence strongly the efficiency of the resulting design processes.

At the German Aerospace Center (DLR) efforts are made for combining data for preliminary airplane design in a single information model. The *Common Parametric Aircraft Configuration Scheme* (CPACS) enables project partners to adjoin and share data from one single source [25, 30].

Along with CPACS there are other information models such as the *STandard for the Exchange of Product model data* (STEP) published by the International Organization for Standardization (ISO). STEP is widely spread in today's industry, especially for the exchange of geometric data. Usage of STEP is made in aerospace, automobile and ship engineering [15].

The *Unified Modeling Language (UML)*, having its origins in software engineering, being distributed by the Object Management Group (OMG), and its successor the *Systems Engineering Language (SysML)* have also been used to model airplane data [9, 31, 45].

The goal of this work is to analyze and compare these information models. Furthermore, the goal is to find out how the different information models can help to increase speed and quality of future aircraft design projects.

2 REQUIREMENTS

This section will give an outlook on some of the requirements and benchmarks that are important for the evaluation of modeling languages. At first, the scope of the application for the modeling, namely preliminary airplane design, must be declared. Without going into to much detail three different stages of airplane design can be distinguished. These are

- conceptual design,
- preliminary design and
- detail design.

Conceptual design so far mostly consists of history-based empirics increasing the design space from a set of requirements to a first concept for a new design. The number of parameters is generally small and analogous models are

* Lufttransportkonzepte und Technologiebewertung, DLR e.V., Blohmstrasse 18, 21079 Hamburg, Deutschland
† Simulations- und Softwaretechnik, DLR e.V., Linder Höhe, 51147 Köln, Deutschland
‡ Institut für Statik und Dynamik der Luft- und Raumfahrtkonstruktionen, Universität Stuttgart, Pfaffenwaldring 27, 70569 Stuttgart, Deutschland

applied. The models used in preliminary design are usually lower order physic-based models. The application of these models mostly requires a geometry definition as input, while runtimes and the amount of data are still sparse. Finally, in this work detailed design is seen as the phase where high level models are used to run the analysis. Due to the fact that these models require long runtimes and create great amounts of data the exchange of this data falls out of the scope of the modeling approaches that are explained in this work.

Lee states that a quality information model is described as a model that is: "*complete, sharable, stable, extensible, well-structured, precise, and unambiguous*", [28]. In their introduction to STEP Anderl and Trippner outline the central functions of a product model as the exchange, storage, archive and transformation of product data [4]. While the list can probably be continued beyond this point, we try to give an outlook on some of the most important aspects of information models.

Several possibilities of attributes for a quality information model are named in the literature and are outlined in this research. These attributes or requirements include important aspects like holism, accessibility, transparency and ordering mechanisms. Additionally, several abstraction methods need to be taken into account that are mostly based on principles of object oriented modeling.

The first and foremost important aspect of an information model in preliminary airplane design is holism. The model does not only need to include all engineering domains but also other departments like marketing, sales and production. The holistic approach can be subdivided into two sections. The first and most obvious requirement is the multidisciplinarity of the model. As the data has to be within the model, it also has to be available for design work. Hence, accessibility is another issue that can be linked with holism.

2.1 Multidisciplinarity

A multidisciplinary model can be defined as a model that holds a common subset of parameters and objects that have a relation to more than one domain concerned with the design of the product. This model definition for example goes beyond the point of dual disciplinary models that are used for fluid-structure interaction. These models are related to specific tools and are from an abstract point of view handled as one domain.

Working in a multidisciplinary environment is an uncommon task for a design engineer who usually is concerned with single domain solutions. The design engineer should therefore be able to still work with his known single domain tools, as can be found by Hoofman et al. in [20]. Working in a common environment is one reason for this organization of tools. Additionally, there is no need to re-implement parts of a model that are not related to other domains. A multidisciplinary model does not replace single domain specialists but makes parts of their knowledge available to others in the design process. The goal of a multidisciplinary model is not to create an engineer that has full knowledge in all accessed domains. The key is to provide the information that influences the global design operations and hence reflects the emergence of the integrated models. This brings up two major issues:

- identifying parameters that are relevant for a multidisciplinary model
- announcing domains accessing/influencing a parameter

The first issue can only be handled by modeling specialists defining the meta model for the multidisciplinary model in combination with single domain design experts. The modeling specialist needs to know about the requirements for several design tasks in single domain environments and their output and impact on the model. From this information he needs to extract common subsets and make them available in the holistic model.

Introducing a parameter to a model is however not enough. A designer working on a model needs to know which design task does influence a parameter. Only in this way changes to parameters in the model are re-traceable, and only in this way an engineer is informed that his changes may influence other domains of the model. This is a major task when checking if a model is valid, e.g. all calculations have converged.

2.2 Accessibility

As the model is built up as a multidisciplinary model it needs to be accessible for all domains contributing to it. At this point it must be seen that while the model should be set up by a modeling expert it is usually accessed by a design engineer coming from a single domain background. The design engineer reads data from the model, processes it and writes it back. Hence the model should be accessible as easy as possible. In a study [16] about this issue Fidel and Green find the most frequent accessibility factors for documentary sources as:

- Saves time
- Has the right format
- Is physically close . . .

These points are probably some of the most important. Domain engineers first need to be convinced that a holistic approach to airplane design can result in major benefits for the product. The academic benefit from integrating a tool into a process chain in opposite is unquantifiable. Nevertheless, static boundary conditions in disciplinary analysis are replaced by dynamic input from other domains. Therefore the integration into and work with a new model should be as easy as possible. Note that the people working on the tools are in the first line engineers, mathematicians or natural scientists. They are experts in their domains and inconclusively software engineers.

2.3 Reuseability

Modeling product data is a time and cost intensive task. As a result this task should not be repeated for every product or product derivate. In the automobile industry products are distributed in different configurations (coupe and convertible) as well as in the aerospace industry (extended range

or freighter). The created information goes through the process of: specification, development, production, test, modification, use, storage, etc, as stated by Stark in [43]. Enabling the reuse of this information is a major task for the information model.

The creation of a detailed meta-model (e.g. well defined classes) allows the reuse of information. It is important to note that the information model should hold this information in form of applied modeling mechanisms and not in form of documentation. Documentation can encourage the understanding of a model but is often used to replace elements that could be modeled as well. Furthermore documentation may be ignored as it makes limitations to the model that can not be validated.

Rule-based systems can offer operations to build up a model. When the set of rules (production system) that was used to create a model is known, a reconstruction of the model or variants of it can easily be achieved by using a similar set of rules. Offering a model creation process can eliminate *copy&paste* approaches and therefore reduce erroneous.

2.4 Abstraction Methods

The process of modeling is divided into two parts. On the one hand there is the specific or *instance* model of a product. This model carries detailed information about the product, in our case the airplane. For example this model gives information about the span of the wing as well as information about the number of seats in the business class section.

On the other hand the information that is detailed in the instance model is placed as abstract information in the meta-model. The meta-model describes the structure and types that can be used to instantiate the specific model. The metamodel describes the wingspan as an attribute of the wing and the number of seats as an attribute of the fuselage section. For the span it allows positive values from type meter. The number of seats must be an integer value. In this work we examined languages for meta-models that are mostly based on object oriented mechanisms. The fact that future product models should use object oriented modeling has been stated by Stark in [43]. In their paper [8] Bertino and Martino describe the basic concepts for such a model:

- each real world entity is modeled by an object
- each object has a set of instance attributes and methods
- the attribute values represent the object's status
- objects sharing the same structure and behavior are grouped into classes
- a class can be a specialized version of one or more classes

Additional information on object oriented modeling is given in the language specific literature such as in [40] by Schenk and Wilson on EXPRESS or in [34] by Oestereich on the UML.

3 MODELING LANGUAGES

An information model is defined in [32] as "*a collection of symbol structure types [...] and a collection of general integrity rules*". Symbol structure types describe entities that can be produced inside the information model. General integrity rules check for the consistency of the produced entities inside the model. A typical information model is described by the relational database model from [13]. The model is established from the three different symbol structures: table, tuple and domain. The integrity rule for the relational database is defined as: No two tuples within a table can have the same key.

Another more extensive definition for an information model is given in [28], "*an information model is a representation of concepts, relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse*". One of the first attempts to create modern information models was established by the creation of the Entity Relationship Model described in [11].

The creation process of an information model is described in the literature as well. During the development process of an information model, four different worlds are distinguished by [23]:

- **subject world:** subject matter of the information system
- **system world:** information system itself
- **usage world:** organizational environment
- **development world:** describes the creation of the information system

An information model for a design environment as described above consists of two aspects. On the one hand the elements, attributes and their structure need to be defined in a schema definition. On the other hand the explicit content has to be stored in a data set which conforms to the schema definition. Whereas the data set is mainly used for the exchange of information, the schema definition is utilized for documentation, model validation and model generation.

In the following sections the already named modeling languages and standards are described in further detail. The scope of this paper is however limited, so that only some of the aspects of our research can be included. Detailed reviews on e.g. the syntax of the underlying languages are therefore disclaimed.

3.1 STEP

The ISO 10303 Product data representation and exchange is also known as **ST**andard for the **E**xchange of **P**roduct **D**ata. It is a set of standards for the modeling of product data. The following sections give an outlook on the requirements that were set up during the development process of STEP. The history of STEP along with some of its ancestors is presented. Subsequently, the structure of the standard along with its operational area and some of the main tools that have implemented STEP is illustrated.

At the beginning of the STEP development several requirements were determined that describe the scope of the

evolving standard. Fowler elaborates these requirements in [17] as:

- The creation of a single international standard, covering all aspects of CAD/CAM data exchange
- The implementation and acceptance of this standard by industry, superseding various national and *de facto* standards and specifications
- The standardization of a mechanism for describing product data, throughout the life of a product, and independent of any particular system
- The separation of the description of product data from its implementation, so that the standard would not only be suitable for neutral file exchange, but also provide the basis for shared product databases, and for long-term archiving

Ongoing from these requirements the development of STEP started within the International Organization for Standardization (ISO) in 1984. Other neutral file formats of the time did not comply with the challenges.

The most famous ancestor of STEP is the Initial Graphics Exchange Specification (IGES). IGES has been developed in the 1970s by the IGES/PDES Organization and was published as ANSI standard. It is still used for the exchange of geometric information and most major CAD-Kernels offer translators for IGES although it has been officially replaced by STEP [17, 21].

The fact that no parametric information is translated by STEP is in someway appreciated by industrial customers [12]. In this way only pure geometric information *en bloc* is exchanged. This is common practice between OEMs and their suppliers. Additional features of STEP are implemented within some parts of CATIA and the software packages like JSDAI and IDA-STEP. A more extensive list of tools that use STEP can be found in the STEP Application Handbook [1].

Due to the long and intense standardization process of STEP its bandwidth is broad enough to attract the big companies in the aircraft industry. Boeing has started working with STEP early on AP 210 as stated by Smith in [41]. In 2004 the *Value Improvement through a Virtual Aeronautical Collaborative Enterprise* (VIVACE) project started. For the developed *Engineering Data Management* (EDM) framework STEP was also used, [44].

Generally, all software tools that implement STEP APs can choose from a subset of conformance classes. A conformance class covers a set of entities from the AP that can be interpreted semantically. This however makes it difficult for a user who works with a certain AP to know if his work can be translated into different systems. Due to the ongoing development of STEP and its surrounding tools a milestone system seems to be an adequate measure, but the implementation level has to be found for each tool separately [1].

EXPRESS is a language for information modeling. It was developed during the first years of STEP after the initial work was undertaken at McDonnell Douglas Information Systems [24]. Ancestors of EXPRESS are the Natural language Information Analysis Method (NIAM) also known as Object Role Model (ORM) by Njissen [33] and the IDEF1X language developed by the U.S. Air Force [2].

By now, the language is accepted by the Object Management Group (OMG) and is made one of the OMG languages [15]. EXPRESS is part of the description methods in STEP. The language is used to define the entities in the integrated resources. The data itself is stored within formats described in more detail in the parts 21 and 28.

Part 28 was introduced as a *new item* to the TC184/SC4 committee in 1999. It is supposed to substitute part 21. The reasons to choose XML are expressed by Kimber in [26]:

- defines a generic, robust, and time-tested character syntax for representing structured data objects
- facilities for the syntactic validation are available
- inherently extensible and flexible
- supported in Web browsers
- tied to an existing ISO standard, ISO 8879 (SGML)

A similar list of arguments can be found from Peak et al. in [36]. The implementation of XML in STEP however is only limited to the instance level.

AP 214 was designed to support the management of product data in the automobile industry. The development was initiated by the Verband Deutscher Automobilhersteller (VDA). Todays work on AP 214 is sponsored by the shareholders of the ProSTEP Association and the BMWi (german federal ministry of economy and technology). Some encouragement comes from members of the SASIG (STEP Automotive Special Interest Group) [18]. The development of AP 214 began during the 1990s, the latest version [22] was published in 2003.

There are no APs that are related to the aircraft industry explicitly. Some shipbuilding APs are released, but they lack the extent of AP 214. As long as no production related decomposition of the product is intended the approached tools can be dropped for preliminary airplane design. As a matter of course, the representation of the product data in various CAD formats (e.g. B-Rep, CSG) is part of the standard. AP 214 was therefore chosen for the comparison. STEP APs benefit from the large number of tools supporting the standard. It must however be noted that the standard covers a huge bandwidth and is as a matter of course inflexible to adjustments from smaller design teams.

3.2 UML

The development of the **U**nified **M**odeling **L**anguage began in 1990. The three authors Booch, Rumbaugh and Jacobsen joined their work for an object oriented modeling language in [10]. UML2 was published in 2005. The language is published by the *Object Management Group*(OMG). Primarily, the UML has been developed to support the automatic generation and description of object oriented software. Through its graphic design layout it enables a lucid representation of contents. Currently the UML is extended to other domains such as systems engineering through the *Systems Modeling Language*(SysML). Further information regarding development using the UML can be found by Oestereich in [34].

As well as a high bandwidth of modeling mechanisms, the UML offers several views on models. These views can come from different approaches, examples are a class or

a use-case diagram. Through these views the model gets reduced to only some aspects. This mechanism allows a better overview, which is necessary in bigger models. Complexity can arise quickly in UML as Oestereich states.

Apart from STEP and CPACS there is as of yet not a common standard for preliminary design in the UML. It is shown in this work that from a technical point of view a translation from these modeling languages into the UML is not a problem. However the following outlook will only display advances for preliminary design modeling that uses UML as a native format.

Lu proposes in [31] requirements for a Next Generation Aircraft Conceptual Design Software Environment (NextADE). The work offers a detailed overview of modeling techniques and implementation methods. The implemented version of NextADE is coded in Java with the help of UML. Several UML views are used for displaying the different development matters.

An application for high fidelity multidisciplinary optimization using the UML can be found by Alonso et al. in [3]. The modules of the software framework for aircraft analysis are modeled as UML classes. The modeling is strictly object oriented and hence also includes methods of the modeled objects.

Oh and Yan [35] use the UML in combination with PDM Systems. In their approach the UML is used to create a mapping from product data stored in STEP files into a commercial PDM System. The transfered STEP is limited to CAD-Systems and conform to AP 203. STEP and UML are brought into relation more and more often. Besides the official Part 25: *EXPRESS to OMG XMI binding*. An overview on the connections between XML, UML and STEP is given by Peak et al. in [36]. In this work the UML is more seen as an implementation model. In the author's opinion the UML holds the most powerful modeling mechanisms of the presented languages.

Van der Laan introduces the use of Knowledge Based Engineering (KBE) for airplane design in [45] with the UML. He describes the design process using KBE and outlines some tools that are used for this process. The step from an informal model to a formal is made using the UML. This work is linked to the research activities at the TU Delft.

3.3 Design Languages based on UML

As already mentioned in the previous section on requirements for modeling languages the aspect of reuseability is a major factor. The linkage of the UML to software engineering and its advanced modeling mechanisms have led to a number of approaches for model generation that are highlighted within this section.

Graph-based design languages for the support of conceptual engineering design are now around for more than one decade [6]. As one of several special features, design languages do not only support the construction of a central model, but allow to store and replay the whole design process history [27, 37]. Furthermore, since from the theory of modeling it follows that any model serves a specific purpose [42], multiple models can be generated through dedicated interfaces (plugins) appropriately. This feature of design languages eases therefore consistent multi-disciplinary modeling and has been demonstrated in the early conceptual design of complex space systems such as satellites [39, 19] and airplanes [9].

More recently, some implementations of the concept of graph-based design languages [37] have adopted the current trend in computer science and use the UML as means of design information representation [9]. As a result, many commercial and open-source tools such as Eclipse[1] originally conceived for software design and development can also be used now for classical engineering design and development purposes.

A specific example of this approach using design languages based on the UML is the design compiler 43[2] for the compilation of the appropriate models. This has been demonstrated in the application case of preliminary airplane design and can be found in [9]. In this work airplane geometries are described by symbolic expressions and can be generated via a rulebased system. The output of the model is then transferred to CATIA as a waterproof surface that can be used for e.g. CFD analysis.

However, in order to successfully work with design languages, a high level of understanding of both the engineering and the computer science world is still required. It is hoped that this user complexity will drop to an acceptable level in the future when more and more libraries for complex engineering subsystem models become better available and usable on a 'LEGO'-like component and system design level.

3.4 CPACS

CPACS stands for **C**ommon **P**arametric **A**ircraft **C**onfiguration **S**cheme. CPACS came up during the first TIVA project at the German Aerospace Center (DLR). The DLR carries out major research in aerospace related technology. Due to its wide spread structure, many different tools are developed separately. These tools come from domain experts and vary in their level of detail from simple stochastic and analytic approaches up to full numeric simulations. While these tools have been tested in single domain projects, the future goal is the integration into multidisciplinary design workflows covering the whole bandwidth of the DLR's aircraft research topics. Tools that need to be integrated are either under development or commercial (or tools with code that can not be accessed for other reason).

CPACS is based on XML technologies that are used also in other projects related to aircraft design. These can be found for example in a framework of aircraft conceptual design including environmental performance studies by Antoine et al. in [5]. The data exchange in the presented CAFFE framework is done via XML files.

TIVA I started in 2005. Evolving from this first project CPACS has been used and extended in the following projects:

- **TIVA I & II**
  Technology Integration for the Virtual Aircraft
- **CATS**
  Climate optimized Air Transport System

[1] www.eclipse.org

[2] www.iils.de Design Compiler 43 is a trademark of IILS Ingenieurgesellschaft für Intelligente Lösungen und Systeme GmbH

- **EVITA**
  EValuation of Innovative Turbine Engines
- **UCAV 2010**
  Unmanned Combat Air Vehicle

The scopes of these projects are spread from analysis of climate data over the preliminary design of airplanes to multidisciplinary approaches in jet engine design. Of course, the detail level in each project is different. While for example EVITA contains extensive data on jet engines, the engine data in TIVA is displayed on a lower granularity. TIVA II ended mid 2009 and is followed by VAMP. This project continues the integration of tools and establishes further process chains in preliminary design. A more detailed description of the project's scope and the relevance of CPACS can be found in [7] by Bachmann et al. and by Liersch et al. in [29].

The tools are integrated using a framework that is based on a Server/Client approach. A user chooses a CPACS data set and connects the data set via a process chain to several tools. The tools are running on servers that are set up at the different institutes of the DLR. While executing the process chain the data is transferred to and from the servers.

The integration of one specific tool is established by a mapping structure. This way a tool designer only has to build one XML-file for input and one for output. Via the mapping this data is linked to entities in the CPACS-file. A change in the CPACS-file only leads to a change in the mapping files. The tool itself does not need to be altered. The mapping is also established using XML-files.

It should be noted that XML does not more than markup data. It does not help in any way to interpret or process the data as stated in [26]. Only putting data into XML syntax can not result in a persistent data model. The creation of a metamodel (e.g. in XML-Schema) is therefore an irreplaceable step. It can be found in [38] by Sachers, that there are no guidelines or examples how to use XML in product data management yet. The current work on CPACS may help to close this gap.

The content-model of CPACS is created using the XML-Schema Definition. XML-Schema is explained by Duckett et al. in [14] and by van der Vlist in [46].

XML-Schema enables the user to create extensive documentation of the created data via annotations. The annotations can be used for the documentation of the saved data as well as annotations for applications processing the data.

Additionally to the XML-Schema elements, the developers of CPACS included own mechanisms for the work with the content model. To allow the creation of these relationships in CPACS as well, a new element for identification was introduced. These unique identifiers (`uIDs`) are introduced as attributes. They enable the libraries to *jump* inside the model. An element like an airfoil for example is tagged by an `uID`. A wing segment can then reference the airfoil data by the `uID`.

Including all data into one single XML-File can lead to very big files. This can cause trouble with the limitations of simple parsers as well as with the readability of the file. Therefore an `externaldata` node was introduced into CPACS. This element allows the import of data from other XML-files. The structure is redundant and hence huge XML-trees can be constructed from several simple files. In this way all parts of the model's external data can be attached to a leaf.

Generally speaking, XML Schema offers a broad variety of modeling mechanisms. In depth these mechanisms show some weak points. The creation of relationships is not very lucid. Even though graphical representations of the model can be created, only the tree structure is recognizable. The inheritance mechanisms are suitable for easy modeling activities but e.g. do not enable multi-inheritance.

XML-Schema offers simple mechanisms for the creation of a content model. The standard is widely spread due to the extensive use in other application areas.

4 MODEL CONVERSIONS

4.1 CPACS to STEP

Although the CPACS standard is spread more and more through the DLR and it is written in XML, one of the most wide spread data exchange formats, one of the goals of this work was to increase its compatibility. STEP is a standard that can be interpreted by all more advanced CAD-Kernels. It was our goal to write a converter tool that enabled us to put out all data from CPACS to STEP. This included all the non geometric information in CPACS as well, and can therefore not be handled by a geometry converter.

As the class structure is given, it is not known whether any or how many objects are instantiated in the CPACS tree. Changes can come up for every CPACS file. A fuselage element can for example carry fifty or hundred points. In this case the method must be modular so that it can act with an arbitrary number of objects. Another issue might come up with changes in the CPACS schema. Updates from the schema are followed by changes in the compiled classes. As the compilation can be automated, the method should adapt to these changes by itself. The method that accesses the CPACS objects is therefore recursive and reflexive.

A recursive method calls itself as long as no break condition becomes true. The method designed to process the CPACS data starts at the top-level CPACS element and checks all lower classes. If a class holds an object it calls a `process2Step` method that converts the object. Afterwards the method calls itself and processes all sub elements of the converted class. The break condition becomes true if it finds an object of one of the base types.

While recursion is a term well known in software development, reflexion came up with the use of object oriented programming languages. Reflexion allows to gather knowledge about classes and objects during runtime. This way the central code of the converter can be kept modular. Even if the CPACS version changes, meaning that the compiled class structure changes, the methods still work because the class structure is not hard coded. The method already described therefore runs through the whole CPACS data and picks out all classes and objects while processing them. This way it can be guaranteed that all information stored in the model is found. The converter tool knows two different sorts of classes:

- elements and attributes that are translated generally, because there is no semantic equivalent in STEP

- elements and attributes that need to be translated semantically correct, because they are interpreted by STEP processors

An approach for the conversion of the first sort of classes is described by Jaroš in [24]. In his work he describes the mapping to the elements `item`, `specific item relationship`. These elements are part of the ARM. In the AIM the equivalent entities are `product` and `alternate product relationship`. The idea behind this mapping method is that elements and attributes are written to the `product` entities and the tree structure is reconstructed using `alternate product relationship` entities.

For the second set of classes specific methods have to be created that process the data to STEP. Each new class owns a method that processes objects of this class to STEP. These methods are of course specific for each class. The second type of classes only comes up with geometric entities that need to be processed from CPACS to STEP. Figure 1 shows a graphical representation of the translation process.

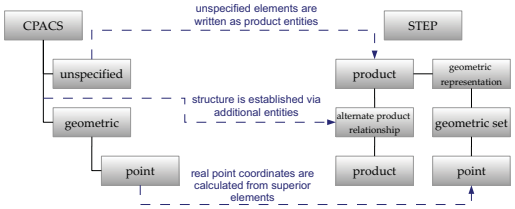

Figure 1: Converter Structure

4.2 CPACS to UML

The above section showed the conversion from CPACS to STEP. As the meta-model in STEP is defined by the ISO standards only the content model could be created in respect to these standards. As already stated there is as of yet no standard for preliminary design in the UML. A conversion from CPACS to the UML can therefore also include a creation of the meta-model. The following figure 2 shows the CPACS structure in a UML class diagram.

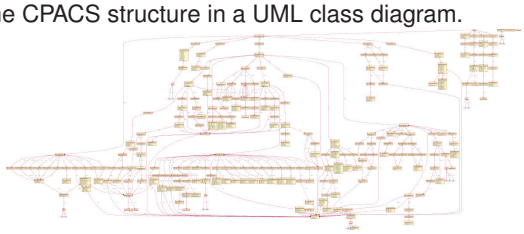

Figure 2: CPACS to UML Class Diagram

Reduced to the software engineering point of view the conversion can be seen as the import of a XML Schema Definition to UML. Several software packages are available for this task. During the research *hypermodel*[3] provided the most valuable results. As content and meta model are integrated within the UML the creation of new models is easily possible. Interpretations for this conversion are however not available as there are no adjacent tools. A conversion to the UML can result in major benefits for connecting to other modeling languages, e.g. ModelicaML[4]. This enables the connection of analysis modules and model definitions in an integrated model.

5 DISCUSSION

In this work requirements for modeling languages have been set up. These requirements are then reflected upon some selected standards.

STEP is the standard within the industry for the exchange of among others geometric information. One of the drawbacks for preliminary design hereby is the explicit modeling and hence the hampering of parametric studies. EXPRESS offers the basic modeling mechanisms but is in some points weaker compared to modern languages.

CPACS is still growing and as of yet only in use in an academic environment. The number of disciplines is currently increasing, whereas the number of tools has exceeded more than 20. The content model in XML is easily exchangeable and can be handled by all developers. Conversions from XSD to other modeling mechanisms have been shown within this work.

The UML turns out to be the most powerful of the outlined modeling languages. Applications with a need for a smaller bandwidth may make use of the SysML. Several applications in cooperation with design languages have shown the value of a high level modeling language for the generation of new models. Standards using UML still need to be established.

While examining the different information models it became obvious that in some cases the responsible organizations try to use synergy effects. It can be seen that a lot of the development is heading in the direction of UML / XML and that generally speaking the exchangebility between models is growing.

- EXPRESS is adopted as an OMG language
- STEP develops parts for systems engineering similar to SysML
- exff[5] maps EXPRESS to the UML
- STEP utilizes XML as content file format

While working on models in preliminary design it became obvious that for larger scope projects the integration of several partners accessibility seems to be the key factor. The drawback from these models however is that advanced processes, e.g. model generation can suffer from the simplicity of the model. A combination of simple content models with advanced modeling available in the meta model may therefore result in an enhanced modeling in preliminary airplane design. Additionally, high level modeling languages may feature the integration of further calculation modules to create integrated design models.

[3] www.xmlmodeling.com/hypermodel
[4] http://openmodelica.org/index.php/developer/tools/134
[5] www.exff.org

References

[1] STEP Application Handbook ISO 10303. *SCRA* (2006).

[2] AFWAL /MLTC. Integrated Information Support System(IISS), Common Data Model Subsystem, Part 4: Information Modeling Manual - IDEF1X. *AFWAL-TR-86-4006 5* (1985).

[3] ALONSO, J. J., LEGREASLEY, P., VAN DER WEIDE, E., MARTINS, J. R. R. A., AND REUTHER, J. J. pyMDO: A Framework for High-Fidelity Multi-Disciplinary Optimization. *10 AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA 2004-4480 (30 August - 1 Septemper 2004).

[4] ANDERL, R., AND TRIPPNER, D. STEP- Eine Einführung in die Entwicklung, Implementierung und industrielle Nutzung der Normenreihe ISO 10303. *B.G. Teubner Stuttgart* (2000).

[5] ANTOINE, N., KROO, I., WILLCOX, K., AND BARTER, G. A Framework for Aircraft Conceptual Design and Environmental Performance Studies. *10th AUAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA 2004-4314 (30 August- 1 September 2004).

[6] ANTONSSON, E., AND CAGAN, J. Formal engineering design synthesis. *Cambridge University Press* (2001).

[7] BACHMANN, A., KUNDE, M., LITZ, M., AND SCHREIBER, A. A Dynamic Data Integration Approach to Build Scientific Workflow Systems. *Grid and Pervasive Computing Conference* (2009), 27–33.

[8] BERTINO, E., AND MARTINO, L. Object-oriented database management systems: Concepts and issues. *Computer 24*, 4 (1991), 33–47.

[9] BOEHNKE, D., REICHWEIN, A., AND RUDOLPH, S. Design Language for Airplane Geometries using the Unified Modeling Language. *ASME Int. Design Engineering Technical Conferences (IDETC) & Computers and Information in Engineering Conference (CIE)* (2009).

[10] BOOCH, G., RUMBAUGH, J., AND JACOBSEN, I. The Unified Modeling Language User Guide. *Addison-Wesley* (1999).

[11] CHEN, P. P.-S. The entity-relationship model - toward a unified view of data. *ACM Trans. Database Syst. 1*, 1 (1976), 9–36.

[12] CHOI, G., MUN, D., AND HAN, S. Exchange of CAD Part Models Based on the Macro-Parametric Approach. *Int. J. of CAD/CAM 1* (2002), 13–21.

[13] CODD, E. F. A relational model of data for large shared data banks. *Commun. ACM 13*, 6 (1970), 377–387.

[14] DUCKETT, J., GRIFFITN, O., MOHR, S., NORTON, F., STOKES-REES, I., WILLIAMS, K., CAGLE, K., OZU, N., AND TENNISON, J. Professional XML Schemas.

[15] FEENEY, A., AND PRICE, D. FutureSTEP Project. 11th NASA/ESA Workshop on Product Data Exchange.

[16] FIDEL, R., AND GREEN, M. The many faces of accessibility: engineers' perception of information sources. *Information Processing and Management 40*, 3 (2004), 563–581.

[17] FOWLER, J. STEP for Data Management, Exchange and Sharing. *Technology Appraisals Ltd.* (1995).

[18] GEISSEN, M. Einen STEP vorraus. *Digital Engineering Magazin 2* (2005), 36–37.

[19] HARMSEN, N., KORMEIER, T., AND RUDOLPH, S. Structural Dynamic Conceptual Design of Satellites By Design Language. *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Honolulu, USA (April 23-26 2007).

[20] HOOFMAN, J., MULYAR, N., AND POSTA, L. Coupling Simulink and UML Models. *Formal Methods for Automation and Safety in Railway and Automotive Systems 1* (2004), 304–311.

[21] IGES/PDES ORGANIZATION. Initial Graphics Exchange Specification 5.3. *U.S. Product Data Association* (1996).

[22] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. Industrial automation systems and integration - Product data representation and exchange, Part 214: Application protocol: Core data for automotive mechanical design processes. Second edition.

[23] JARKE, M., MYLOPOULOS, J., SCHMIDT, J. W., AND VASSILIOU, Y. DAIDA: an environment for evolving information systems. *ACM Trans. Inf. Syst. 10*, 1 (1992), 1–50.

[24] JAROŠ, M. Integration des STEP-Produktmodells in den Getriebeentwicklungsprozess. *Technische Universität München* (2006).

[25] KERSKEN, H.-P., LITZ, M., CORNELSEN, H., AND SCHREIBER, A. A Software Environment for multidisciplinary Simulation in Aircraft Predesign. *to be published*.

[26] KIMBER, W. XML Representation Methods for EXPRESS-Driven Data. NIST GCR 99-781.

[27] KRÖPLIN, B., AND RUDOLPH, S. Entwurfsgrammatiken - Ein Paradigmenwechsel? *Der Prüfingenieur 26* (2005), 34–43.

[28] LEE, Y. T. Information Modeling from Design to Implementation. *Manufacturing Systems Integration Division, National Institute of Standards and Technology (NIST)*.

[29] LIERSCH, C., AND HEPPERLE, M. A Unified Approach for Multidisciplinary Aircraft Design. *CEAS European Air and Space Conference* (2009).

[30] LITZ, M., CORNELSEN, H., AND KERSKEN, H. Software Tools and Data Formats for Data Exchange in Airplane Predesign. PDE 2008.

[31] LU, Z. Data Management in an Object-Oriented Distributed Aircraft conceptual Design Enviroment. *PhD Thesis Georgia Institute of Technology* (2007).

[32] MYLOPOULOS, J. Information Modeling in the Time of the Revolution. *Information Systems 3-4* (1998).

[33] NIJSSEN, G., AND HALPIN, T. Conceptual Schema and Relational Database Design: A Fact Oriented Approach. *Prentice Hall* (1989).

[34] OESTEREICH, B. Analyse und Design mit UML 2.0 - Objektorientierte Softwareentwicklung. *Oldenburg* (2005).

[35] OH, Y., HAN, S., AND SUH, H. Mapping product structures between CAD and PDM systems using UML. *Computer Aided Design 33* (2001), 521–529.

[36] PEAK, R. S., LUBELL, J., SRINIVASAN, V., AND WATERBURY, S. C. STEP, XML, and UML: Complementary Technologies. *Journal of Computing and Information Science in Engineering 4*, 4 (2004), 379–390.

[37] RUDOLPH, S. Know-How Reuse in the Conceptual Design Phase of Complex Engineering Products – Or:Are you still constructing manually or do you already generate automatically? *Proceedings Conference on Integrated Design and Manufacture in Mechanical Engineering 2006 (IDMME 2006)* (May 17-19 2006).

[38] SACHERS, M. White Paper for PDM-Integration of OEM and Supplier in the Automotive Industry.

[39] SCHAEFER, J., AND RUDOLPH, S. Satellite Design by Design Grammars. *Satellite Design by Design Grammars, Aerospace Science and Technology 9* (2005), 81–91.

[40] SCHENK, D., AND WILSON, P. Information Modeling: The EXPRESS Way. *Oxford University Press, Inc.* (1994).

[41] SMITH, G. Utilization of STEP AP 210 at the Boeing Company. *Computer-Aided Design 34* (2002), 1055–1062.

[42] STACHOWIAK, H. Allgemeine Modelltheorie. *Springer Verlag* (1974).

[43] STARK, J. Product Lifecycle Management , 21st Century Paradigm for Product Realisation. *Springer London* (2005).

[44] VAN, T. N., FÉRU, F., GUELLEC, P., AND YANNOU, B. Engineering data management for extended enterprise - context of the european vivace project. *Product Lifecycle Management PLM-SP2* (2006), 338–348.

[45] VAN DER LAAN, T. Knowledge based engineering support for aircraft component design. *TU Delft* (2008).

[46] VAN DER VLIST, E. XML Schema: The W3C's Object-Oriented Descriptions for XML. *O'Reilly* (2002).