Management of Flexible Data Exchange Processes in Virtual Product Development

Michael Stoye
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
michaelstoye85@yahoo.de

Stephan Vornholt
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
vornholt@iti.cs.uni-magdeburg.de

Ingolf Geist
Faculty of Computer Science
University of Magdeburg
Magdeburg, Germany
ingolf.geist@iti.cs.uni-magdeburg.de

ABSTRACT

To meet the strong requirements in product development many computer-aided systems are used today. Among these systems digital models have to be exchanged to avoid error-prone and time-consuming recreation of already defined product data. During product development computer-aided systems and file formats for data exchange among them can change because of updated versions of systems or new systems. In those changeable environments, for engineers it is hard to know which of the many possibilities to choose for data exchange between computer-aided systems. Another problem is the traceability of executed data exchange processes.

We propose a data model to support the management of flexible data exchange processes. Therefore, the data model describes and connects system- and transfer-specific data. To create an appropriate data model, detailed knowledge about possibilities for data exchange, computer-aided systems, exchange file formats and underlying processes are necessary. Consequently, we analysed frequently used data exchange file formats and an exemplary development process. In this paper the results of these analyses are presented. Furthermore, the developed data model and a prototypical implementation are described.

Categories and Subject Descriptors

H.2.1 [**Database Management**]: Logical Design — Schema and subschema; J.2 [**Computer Applications**]: Physical Sciences and Engineering — Engineering; J.6 [**Computer-aided engineering**]: Computer-aided design (CAD), computer-aided manufacturing (CAM)

Keywords

Virtual Product Development, Virtual Engineering, Data Exchange, Computer-Aided Systems, Workflow Management, Data Model, Product Data Management

1. INTRODUCTION

Companies, which are developing new products, are under a high pressure of the market. Products have to be developed and produced in a short time, with low costs and high quality. To meet these requirements the development is supported by using many different computer-aided systems (CAx systems). Engineers use some of these systems to digitally model products. Other CAx systems use the modelled data for analyses, simulations, production planning or visualisation, for instance. A continuous usage of CAx systems in the product development is also called *virtual product development*.

Today more and more developed products are mechatronic systems which consist of components developed in different domains (e.g. mechanical, electric and software development). Traditionally separated, these domains have been evolved independently of each other. As a result, different methods and software systems exist for developing in the various domains. Consequently, several specialised CAx systems are used to develop a product. Among CAx systems of different domains, but also among systems inside a domain data has to be exchanged. Thus, for data exchange different interfaces between CAx systems exist. That leads to complex CAx system architectures inside a company or among several companies working together. Furthermore, CAx system architectures change over time because CAx systems can be replaced with a new version or a complete new CAx system. It is difficult to keep an overview of the architecture and the changing possibilities for data exchange among CAx systems. Another problem is the traceability of executed data exchange processes. For the described problems, a data model, which describes the required data (e.g. CAx systems, interfaces, process chains, executed process chains) and their relationships to establish a flexible management of data exchange processes was developed in [20]. We present the main results of that work in this paper.

The fundamentals for this paper are presented in section 2, starting with the reasons for data exchange among CAx systems. Hereafter, typical file formats for data exchange are compared and WFM systems as state of the art for process management are described. In section 3, we present an analysed example of a development process focussing on necessary data exchanges. After this, in section 4, a data model for the flexible management of data exchange processes is presented. A prototypical implementation based on the data model is described in section 5. Finally, we conclude the paper and present future work in section 6.

2. FUNDAMENTALS

In the following sections, the necessity of data exchange, typical file formats for data exchange and current approaches to support data exchange processes are presented.

2.1 Data Exchange

Data exchange between CAx systems is necessary to avoid

error-prone and time-consuming recreation of already defined product data. There are research works for data exchange among CAx systems that use *integrated product models* (e.g. [1, 6]). An integrated product model integrates partial models and can be used as a uniform database for several CAx systems. Because this approach is not supported by the CAx system manufacturers, it is currently not common to use integrated product models [19]. However, the growing complexity of products demands the integration of several domains and is in the focus of current research for this topic [7, 22].

This paper focuses on file based data exchange, the most common way to exchange information between CAx systems. One CAx system stores data into a file and another CAx system can read the data from this file. Data exchange often requires a transformation of data from one CAx system in a readable format for another system. File formats have a defined set of supported data that can be represented. Only data which is in the intersecting set of two CAx systems and the exchange file format can be exchanged between both systems (see Figure 1). For example, material data which can be defined in two different CAx systems, can only be transferred between both systems if the exchange file format supports it, too. Therefore, data exchange depends heavily on the right choice of the exchange file format according to a given use case.

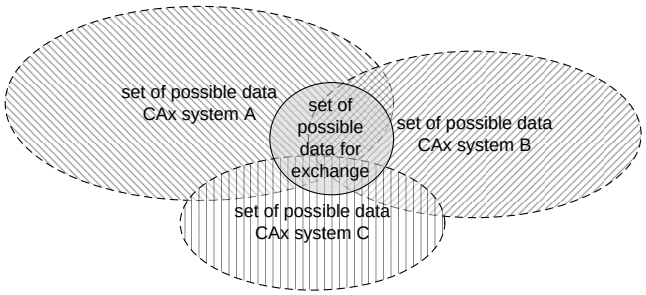

Figure 1: Intersecting sets of supported data that can be exchanged

For the realization of a file based interface there are two different approaches [22]:

- **System specific interfaces** define a specific exchange between two CAx systems. The advantage is that the interface can be optimized in order to support the transfer of many data between both systems. However, to support n CAx systems $n \cdot (n-1)$ specific converters have to be implemented.
- **System neutral interfaces** define a common file format (also called neutral file format) for the exchange between systems. The disadvantage is that the possibilities to exchange data are limited to the supported set of data of the defined file format. The advantage is that the development of interfaces for n CAx systems requires only $2 \cdot n$ converter, which transform data in and from the defined file format.

It depends on a given use case which approach should be used for the realisation of an interface.

2.2 Neutral File Formats

Similar to interfaces, file formats can be separated in *native* and *neutral file formats*. Native file formats are CAx system manufacturer specific and their specification is often not freely available. They are optimised for one CAx system or various CAx systems of one manufacturer and can store all in these CAx systems defined data. Usually, they are not useful for data exchange because only few CAx systems can read them. Thus, a transfer into another native or neutral file format is necessary.

Neutral file formats do not depend on one CAx system and their specification is often freely available. Some neutral file formats are normed. That means they are defined and maintained by a norming institution like the *International Organization for Standardization (ISO)*. We analysed typical standard formats to compare file exchange formats. More and detailed comparisons of neutral file formats are for example given in [2, 17, 21]:

- **Initial Graphics Exchange Specification (IGES)** also known as standard "ANSI Y14.26M" was first released in 1980 and is one of the first developed exchange formats for *computer-aided design (CAD)* systems [14]. Although the development ended in 1996, today IGES is used in many CAx systems especially for the exchange of geometry data.
- **Standard for the Exchange of Product Model Data (STEP)** is a working title for the international "ISO 10303" series of standards [8]. In contrast to many standard formats, STEP does not exclusively focus on geometry data. Many different kinds of product data that are generated in the life cycle of a product can be exchanged using STEP [16]. STEP defines a set of use case specific neutral file formats and can also be used as a construction kit to define new neutral file formats.
- **Jupiter Tessellation (JT)** is an industry standard, which is currently in development to an ISO standard [3]. The version "8.1b" is already available at ISO as a so called *public available specification* ISO/PAS 14306 [10]. JT files contain different representations of 3D models together (tessellated triangle surfaces and parametric models). Additionally, it is possible to define product manufacturing information (e.g. dimensioning, surface quality) and visualisation specific data (e.g. textures, lights, colours). Furthermore, JT has a strong compression algorithm.
- **Drawing Interchange Format (DXF)** is an industry standard developed by the CAx system manufacturer *Autodesk*. The DXF file format is not overloaded and is supported by many CAD systems as an export and import format.
- **Virtual Reality Modeling Language (VRML)** is normed as ISO standard 14772-1 [9]. VRML is a format for the description of 3D scenarios, their geometry, lights, animation and further visualisation related information. In the product development, it is mainly used for the exchange between CAD and *virtual reality* systems.

The paper shows a comparison of the basic properties of the described neutral file formats in Table 1. It describes which neutral file formats are normative standards and which are industry standards. In addition, the type of possible formats for files (ASCII, binary or XML) is shown. Furthermore, the supported content for an exchange is compared.

As conclusion of the comparison: To execute successfully

a data exchange, it is important to know which data can be transferred with a specific file format. The given comparison of the supported content is very compact and has to be refined according to a given use case. For example, if a 3D model with material data has to be transferred between two CAx systems then one has to use a file format that supports both: 3D model data and material data.

Table 1: Comparison of neutral file formats (✓ means can be transferred; - means cannot be transferred)

property	IGES	STEP	JT	DXF	VRML
norm (n) / industry standard (i)	n	n	i	i	n
ASCII (a) / binary (b) / XML (x)	a, b	a, x	b	a	a
2D models	✓	✓	–	✓	–
3D facet models	✓	✓	✓	–	✓
3D parametric models	✓	✓	✓	✓	–
non-geometry data	✓	✓	✓	✓	✓

2.3 State of the Art

WFM systems focus on sending correct information to the right people at the right time. The usage of WFM systems requires the definition of processes. Afterwards instances of those processes are executed and controlled. WFM systems mainly support structured, pre-planned and repetitive processes [13].

Compared to processes in production and logistics, the product design process cannot be pre-planned in many cases. The process often changes during development. Moreover, developing a product is dominated by creative and dynamic processes (e.g. iterations). Predefining the development process would constrain the creative freedom of developers and limit their ability to response. Nevertheless, there are subprocesses in product development that can be pre-planned and automated, for example, release and change processes. Therefore, WFM functionality is often integrated in *product data management (PDM) systems* that are used in product development companies for data management in cooperation with the definition and management of processes [4].

WFM systems can be used for management of data exchange processes but since WFM systems are developed for process management in general they are not suitable to support the management of data exchange processes in detail. The data model presented in this paper could be used to extend a WFM system to meet the special requirements for data exchange processes.

3. ANALYSED DEVELOPMENT PROCESS

We analysed the procedure of a *multibody system (MBS) simulation* as an exemplary data exchange process. Engineers use MBS simulations to analyse the kinematic and dynamic behaviour of a product [18]. In the example process, several CAx systems are used in order to enable a MBS simulation. Data has to be exchanged among these CAx systems. The analysed process is not very complex but it is a typical process in product development environments and contains all problems that motivate the requirements of the traceability of data exchange processes. In Figure 2, a

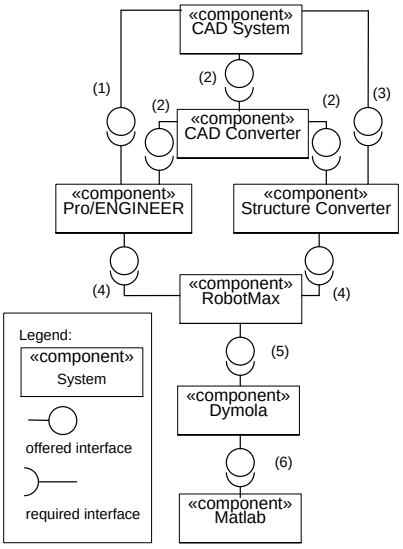

Figure 2: Component diagram showing the involved CAx systems of the analysed MBS simulation process and their interfaces

component diagram illustrates the CAx systems that are involved in the analysed process. In the picture the interfaces are numbered. They also represent process steps for data exchange between the CAx systems. Hereafter, we describe the process steps of the analysed MBS simulation:

As a basis for the MBS simulation a CAD model of a product is necessary. It can be created from any CAD system and saved in a format that can be imported in the CAD system *Pro/ENGINEER* (1). If this is not possible, a CAD converter has to be used to transform the CAD model into a readable format for Pro/ENGINEER (2). As an alternative possibility, the initial CAD model can be saved in the STEP format (3) or transformed into it (2) and be imported in a software system called *Structure Converter*. Pro/ENGINEER or the Structure Converter can create the required data for the *RobotMax system* (4). In this system the data from the CAD model is enriched with electronic components (see [11, 12] for more details). The result is a mechatronic MBS model that is used for a MBS simulation in *Dymola* (5). Dymola is a commercial system for different simulations. It is based on the object-oriented language *Modelica* [5, 15] that can be used for modelling physical systems, which consist of components of different domains. Another way to simulate the MBS system is to create an output file as *C source code* with Dymola for *Matlab* (6). Matlab is also a commercial system that can be used for simulations among other things.

When an instance of the described process has been executed, data is generated. In Figure 3, a UML class diagram is used to illustrate the possibly generated data and their relationships as well as their dependencies. The class *CAD-Data-Model for RobotMax* represents an abstract aggregation of the CAD data that is needed for RobotMax (one or more *VRML files* and an *XML file*). There are two possibilities to create them. One is based on a *STEP-CAD-Model* which might be converted based on another *CAD model*. A CAD model is, according to its format, represented by one or more *CAD files*. For example, for a CAD model in the

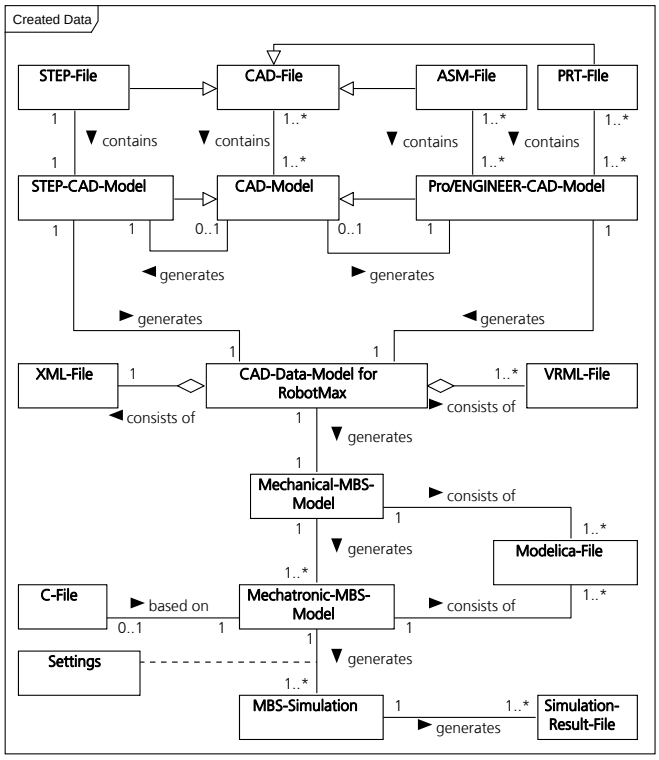

Figure 3: Class diagram with the data and their relationships that are created at the example process

STEP format exists one *STEP file*. The other possibility is a native *Pro/ENGINEER-CAD-Model* represented by one or more PRT and ASM files. Eventually, it can be the result of a conversion from another *CAD model*.

With a given CAD-Data-Model RobotMax creates at first a *Mechanical-MBS-Model*. It is represented by one or more *Modelica-Files*. In RobotMax it is possible to add electronic components, e.g. engines, so that the Mechanical-MBS-Model is extended to a *Mechatronic-MBS-Model*, which is also represented by one or more *Modelica-Files*. Based on one mechanical model different mechatronic models can be created. With a Mechatronic-MBS-Model different simulations can be executed because different settings in the simulation software can be used. A *MBS-Simulation* creates one or more *Simulation-Result-Files*.

It has to be mentioned that in general it is possible to use different settings, when a file is imported or exported in a CAx system. This aspect is not modelled in Figure 3 in order to reduce the complexity and improve the readability of the diagram. An analysis of import and export settings of data in CAx systems is given in [20]. To summarize, the characteristics of the analysed process are:

- different possibilities for the execution of the process
- different file formats are used as input
- data is generated with complex relationships and dependencies
- different settings can be used for importing and exporting files

Based on these characteristics, the requirements for supporting data exchange processes are:

- a description of different possibilities for execution and file formats
- the traceability of executed data exchange processes

The requirement for a description of different possibilities means that a data exchange process should be described in detail. This could be for example, information about the differences between alternate ways for data exchange as well as advantages and disadvantages of file formats in a specific use case. With the help of this information a user can choose the way for his given use case.

The traceability of data exchange processes enables the management of information about executed processes, for instance the chosen exchange steps and file formats, the used settings and also a reference to the files which have been transferred. Based on these requirements we created a data model that is presented in the following section.

4. DATA MODEL FOR FLEXIBLE DATA EXCHANGE PROCESSES

The development of the data model has the goal to enable the management of CAx system architectures and data exchange processes. It meets the demands that we outlined in the last section by the analysis of the exemplary data exchange process. The data model is shown as an UML class diagram in Figure 4. It is separated into two part data models to enhance the comprehensibility and maintainability:

- The first data model described in section 4.1 focuses on the management of CAx system architectures and is represented by the white coloured classes.
- The second data model described in section 4.2 focuses on the management of data exchange processes and is represented by light and dark grey coloured classes.

4.1 System Management

A typical CAx system architecture consists of several CAx systems from different manufacturers, as well as interfaces between them. The part of the data model that focuses on CAx system management allows the description of CAx systems and their versions. Every CAx system version can export and import zero or several file formats. The settings, which can be used by the import and export, can be defined with the class *Settings*. It consists of several separate defined settings. For some settings the user has to choose one of many options. These possible options can be defined with several instances of the class *Choice*. Furthermore, the data model supports the description of interfaces between CAx systems.

The difference between interfaces and file formats for export and import definition of CAx systems is that an interface encapsulates all necessary exchange processes for exchanging data between two CAx systems. For each exchange process different file formats can be used. In the analysed process for example, there is an interface between Pro/ENGINEER and RobotMax. To exchange data between both systems two different exchange processes with different file formats (XML and VRML) have to be executed.

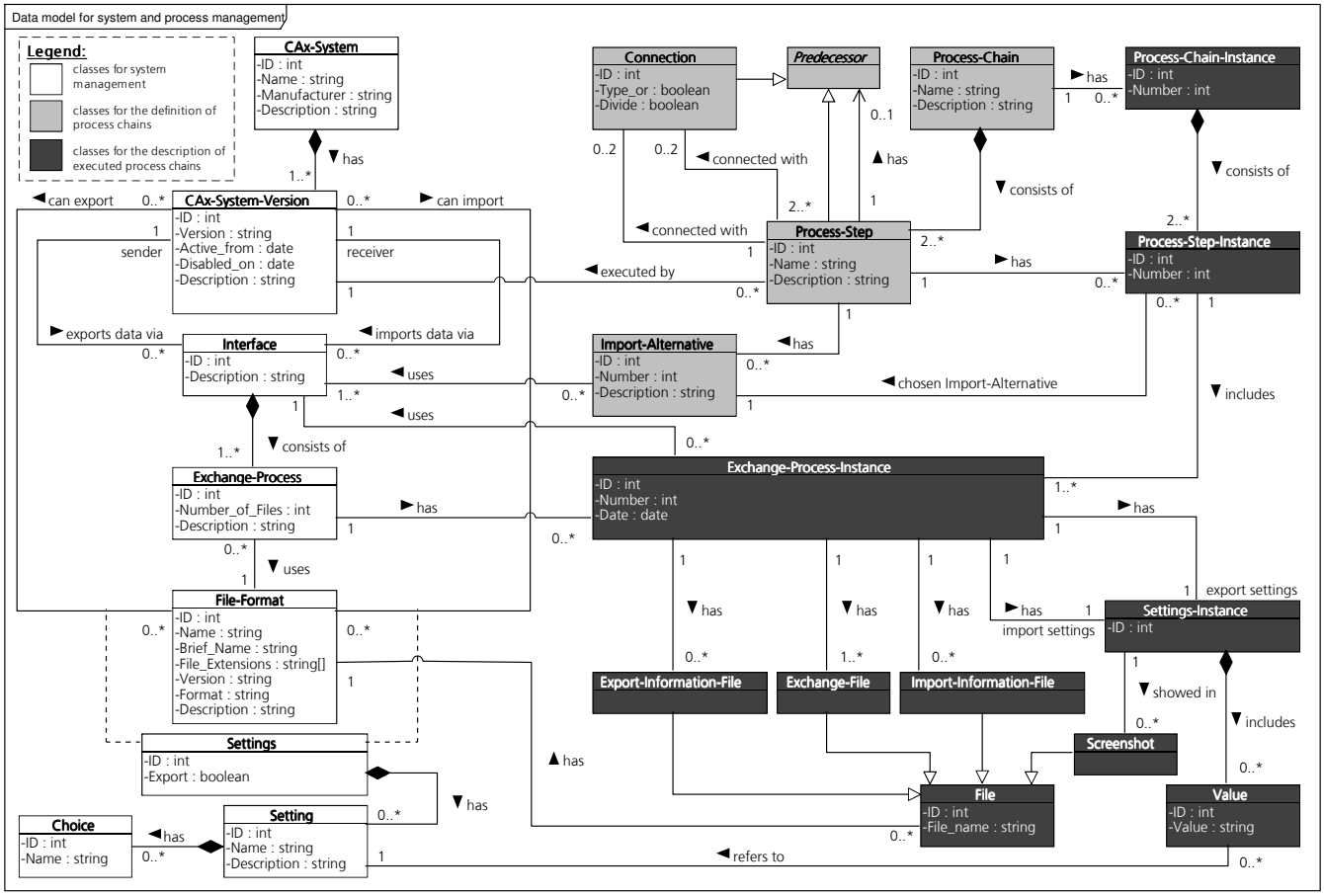

Figure 4: Class diagram showing the data model for system and process management

The defined data model allows the description of a CAx system architecture in a detailed way. It is possible to reconstruct old states of an architecture with the help of the date attributes (*Active_from*, *Disabled_on*). With the data that can be described by the data model, it is possible to create software that can give the following information to the users in an engineering company:

- the CAx systems in the company
- interfaces that can be used to exchange data between CAx systems
- differences or advantages/disadvantages if more than one interface can be used for a data exchange between two CAx systems
- import and export possibilities of CAx systems including settings
- file formats and which can be imported and exported by which CAx system

We use an example to illustrate and explain the data model. The example refers to the analysed process of section 3 of which the interface between the CAx system Pro/ENGINEER and RobotMax was chosen. Figure 5 shows an UML object diagram with instances of the presented data model. Two concrete versions of both CAx systems were defined as well as the related interface, exchange processes and file formats. For an import in RobotMax an XML file and various VRML files are necessary. Pro/ENGINEER can export them by two separate exchange processes.

4.2 Process Management

Based on the proposed data model for system management a data model for process management was defined. It consists of two groups of classes:

- one group focuses on the definition of process chains (light grey classes)
- the other group focuses on the description of executed process chains (dark grey classes)

First, we describe the classes for the definition of process chains. A process chain defines the necessary and possible data exchange processes among several CAx systems. An example for a process chain is the analysed process in section 3. Several data exchange operations have to be executed to realize a MBS simulation. A process chain consists of several process steps, which are connected with each other. Each process step is executed by a defined CAx system version. An example for this is a MBS simulation that is executed by a concrete version of the CAx system Dymola. Each process step, except the first one, has one or more import alternatives. That means, one or different ways to import

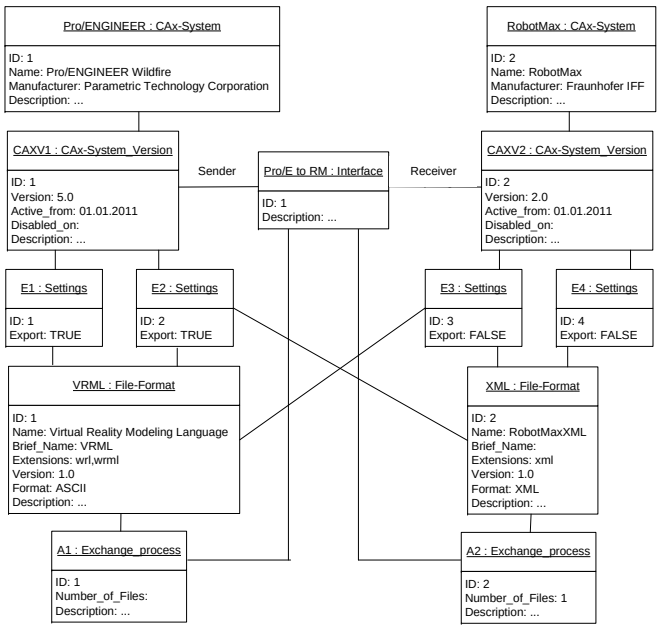

Figure 5: Object diagram for system management (example)

data that is necessary to execute the process step. For example, to generate a mechatronic MBS model in RobotMax one can import data from Pro/ENGINEER or the Structure Converter.

With this part of the data model it is possible to implement software that can be used to define and describe process chains. Every process chain consists of several process steps that are connected by one or different possibilities for data exchange. A user in an engineering company can obtain the following information by defined process chains:

- which possibilities exist to exchange data between two process steps
- what are the differences if there is more than one possibility for an exchange

The second group of classes of the process management data model can be used to describe data that is generated by an executed instance of a process chain. The process chain instance consists of several process step instances and exchange process instances. Every process step instance knows the selected import alternative because in a process chain instance only one of the various possibilities were selected. For each exchange process instance detailed information can be described like the used settings and generated files. This information about an executed process chain makes the data exchange processes traceable. Software that is implemented based on the data model can give a user the following information:

- how a process was executed and which of the possibilities for data exchange were chosen
- the used settings for import and export of files
- exchanged and generated files

5. IMPLEMENTATION

We evaluated the data model by implementing the model in a prototype. The prototype was realized as a web-based client-server application.

The database of the application was implemented as a relational *PostgreSQL* database. Therefore, a relational database schema was defined following the data model as a form of a conceptual schema (see Figure 6). The database schema includes tables to save the data for system and process management. The arrows represent foreign key relations.

The architecture of the implemented web-based application consists of three layers:

- **Presentation Layer:** The user interface is realized by the client side with a web browser. Therefore, several *Hypertext Markup Language (HTML)*, Cascading Style Sheets (CSS) and JavaScript files are provided by the web application.
- **Application Logic Layer:** The application logic is implemented in a *Java web application* that runs on an *Apache Tomcat* web server.
- **Database Layer:** A PostgreSQL database stores all persistent data of the web application and communicates with the web application over the *Java Database Connectivity (JDBC)* interface.

With the implemented prototype it is possible to define a CAx system architecture, process chains and describe executed process chains. Furthermore, the prototype gives access to the defined data.

6. CONCLUSIONS & PERSPECTIVES

Virtual product development includes complex CAx system architectures and between those systems complex data exchange processes. An analysis of typical exchange formats has shown that it is important to choose the right file formats for data exchange processes. Furthermore, based on an example development process we analysed the following requirements for the management of flexible data exchange processes:

- a description of different possibilities for execution and file formats
- the traceability of executed data exchange processes

To meet these requirements we presented a data model in this paper. To enhance the comprehensibility and maintainability it is separated into two data models. One data model describes data for a management of CAx system architectures, for example CAx systems, their versions, interfaces between the CAx systems and file formats that can be used for import and export.

The second data model describes data for the definition of process chains and executed process chains. The definition of process chains allows to describe the necessary and possible data exchange processes among several CAx systems. It is possible to describe different ways for data exchanges in a process chain including, for instance, what advantages and disadvantages for different file formats exist. Based on this information, a user can decide which alternative to choose for data exchange in a given use case. Furthermore, information about executed instances of defined process chains can be described, for example used setting, used alternatives for data exchange and generated files. With this information executed data exchange processes are traceable.

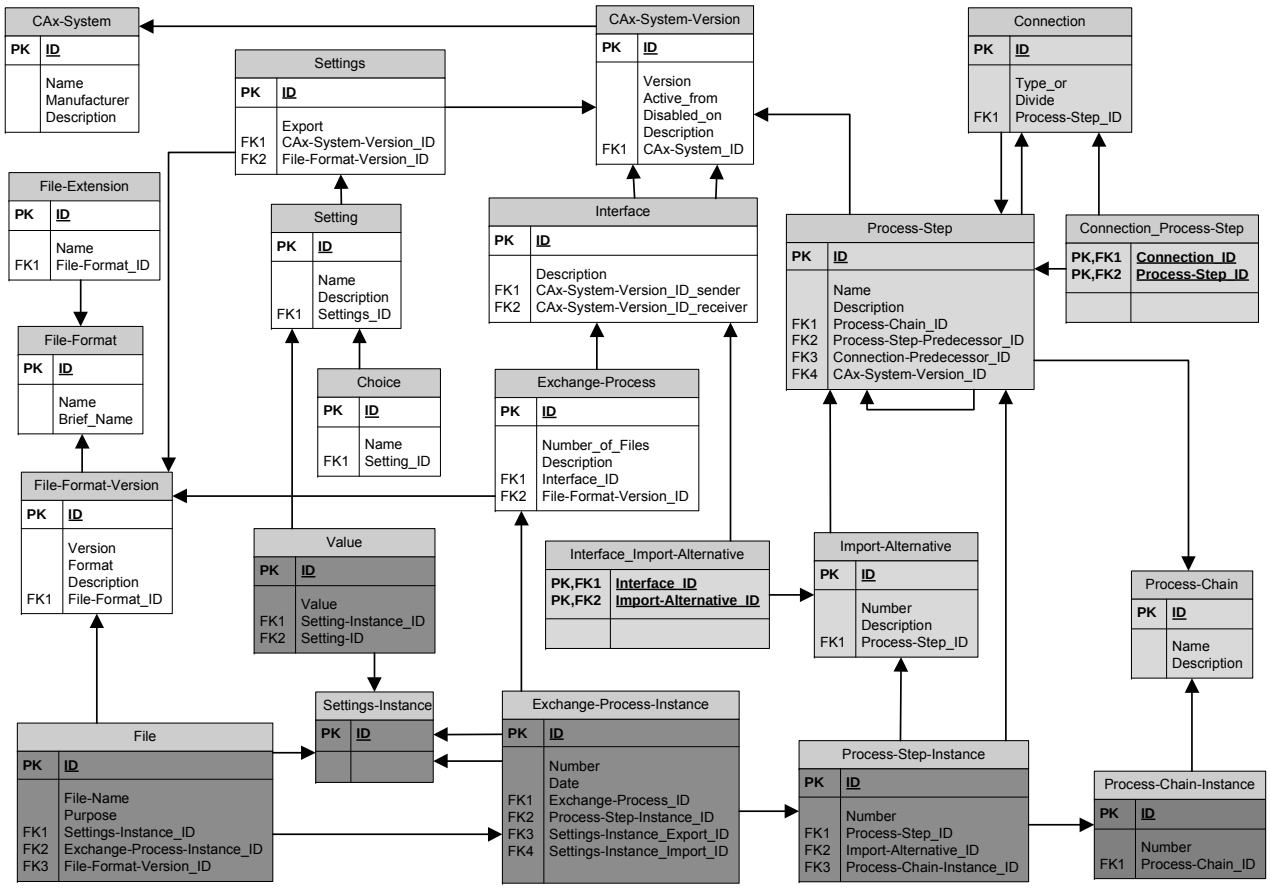

Figure 6: Database schema (white tables = system management data; grey tables = process management data)

To evaluate the data model we implemented a prototype. In the paper the database schema of the prototype was presented and the architecture was described. As a next step, an implementation of the data model as an extension of existing WFM systems or WFM components in PDM systems should be analysed. Furthermore, adjustments of the data model to other domains, and interchange formats of virtual product development are the subject of future research and development.

7. ACKNOWLEDGMENTS

The work described in this paper has been supported by the European Commission: European Regional Development Fund, project "COMO" C1-3201201 and C3-3201201.

8. REFERENCES

[1] G. Brunetti and B. Golob. A feature-based approach towards an integrated product model including conceptual design information. *Computer-Aided Design*, 32(14):pages 877 – 887, 2000.

[2] A. Dyla. *Modell einer durchgängig rechnerbasierten Produktentwicklung*. PhD thesis, Technical University Munich, 2002 (in German).

[3] M. Eigner, S. Handschuh, and F. Gerhardt. Concept to Enrichen Lightweight, Neutral Data Formats with CAD-based Feature Technology. *Computer-Aided Design and Applications*, 7(1):pages 89–99, 2010.

[4] M. Eigner and R. Stelzer. *Product Lifecycle Management: Ein Leitfaden für Product Development und Life Cycle Management*. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2009 (in German).

[5] H. Elmqvist and S. E. Mattsson. An Introduction to the Physical Modeling Language Modelica. In W. Hahn and A. Lehmann, editors, *Proceedings of the 9th European Simulation Symposium ESS'97, 19-23th October, Passau, Germany*, pages 110–114. 1997.

[6] S. J. Fenves, Y. Choi, B. Gurumoorthy, G. M. Mocko, and R. D. Sriram. *Master Product Model for the Support of Tighter Integration of Spatial and Functional Design (NISTIR 7004)*. Technical report, U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, Gaithersburg, MD, 2003.

[7] M. Hirz. An approach of multi disciplinary collaboration in conceptual automotive development. *International Journal of Collaborative Enterprise*, 2(1):pages 39–56, 2011.

[8] ISO 10303-1: *Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles*, 1994.

[9] ISO 14772-1: *Information technology - Computer*

graphics and image processing - The Virtual Reality Modeling Language - Part 1: Functional specification and UTF-8 encoding, 1997.

[10] ISO/PAS 14306: *JT File Format Specification harvesting for 3D Visualization of industrial data*, 2009.

[11] T. Juhász. *Advanced Solutions in Object–Oriented Mechatronic Simulation*. PhD thesis, Budapest University of Technology and Economics, Budapest, 2008.

[12] T. Juhász and U. Schmucker. Automatic Model Conversion to Modelica for Dymola-based Mechatronic Simulation. In B. Bachmann, editor, *Modelica 2008: Proceedings of the 6th International Modelica Conference; 3rd and 4th March, Bielefeld, Germany*, pages 719–726, 2008.

[13] C.-H. Ma, Y.-T. Ko, and D.-B. Luh. A structure-based workflow planning method for new product development management. *International Journal of Management Science and Engineering Management*, 4(2):pages 83–103, 2008.

[14] R. N. Nagel, W. W. Braithwaite, and P. R. Kennicott. *Initial Graphics Exchange Specification (IGES), Version 1.0, N.B.S. Report NBSIR 80-1978 (R)*, 1980.

[15] M. Otter and H. Elmqvist. Modelica - Language, Libraries, Tools, and Conferences. *Simulation News Europe*, December:pages 3–8, 2000.

[16] M. J. Pratt. Introduction to ISO 10303 - the STEP Standard for Product Data Exchange. *Journal of Computing and Information Science in Engineering*, 1(1):pages 102–103, 2001.

[17] S. Schuhmann. *Methoden zur Optimierung des Austausches von Geometrie- und Parametrikdaten*. PhD thesis, Otto-von-Guericke University Magdeburg, 2001 (in German).

[18] A. A. Shabana. *Dynamics of multibody systems*. Cambridge University Press, New York, NY, 2005.

[19] A. Stekolschik. *Ein Beitrag zum ganzheitlichen Qualitätsmanagement von CAD-Modellen in der Produktentstehung*. PhD thesis, Ruhr-University Bochum, 2007 (in German).

[20] M. Stoye. *Entwicklung eines Datenmodells zur Unterstützung des dateibasierten Datenaustauschs in der Produktentwicklung, Diploma thesis, Otto-von-Guericke University Magdeburg*, 2011 (in German).

[21] VDI 2209: *3D product modelling - Technical and organizational requirements - Procedures, tools, and applications - Cost-effective practical use. Beuth Verlag GmbH, Berlin*, 2009.

[22] S. Vornholt, I. Geist, and Y. Li. Categorisation of Data Management Solutions for Heterogeneous Data in Collaborative Virtual Engineering. In G. Saake and V. Köppen, editors, *Proceedings of the First International Workshop on Digital Engineering, IWDE '10, June 14, Magdeburg, Germany*, pages 9–16, 2010.