Research on Feature-Based Data Exchange Between Tribon and UG

Wei Liu, Xiong-hui Zhou, and Xue-yu Ruan

Abstract For shipbuilding and auto industry, using different CAD platform to fulfill one project is very popular. Engineers have to deal with so many data exchange works. While the main problem is that many raw modeling features have been lost during this process and could not be inherited by another system. So from this point of view, how to realize the modeling data exchange or called geometry data seamless integration between heterogeneous CAD systems is very important. In this paper, we propose a new method to realize such function. We trace ship hull modeling history in Tribon and then via the XML neutral file, a same 3d model will be regenerated in another system UG. The experiments show that this data exchange method is fairly high efficiency and all modeling features have been inherited. This method can also be widely used in other feature-based CAD platforms such as Solidworks, CADDS5 and CATIA.

Keywords 3d model regeneration • Data exchange • Feature-based modeling • XML

1 Introduction

With the development of computer science, more and more modern CAD systems have the ability to create parametric models. In a parametric model, each entity, such as a line or circle in a wireframe, or an extrude operation, has parameters associated with it. These parameters control the various geometric properties of the entity, such as the length, radius or height of a cylinder. In order to create the desired part, these parameters may be changed by the operator if necessary. Parametric modelers that

W. Liu (✉) • X. Zhou • X. Ruan
National Engineering Research Center of Die & Mold CAD, Shanghai Jiao tong University, Shanghai, China
e-mail: claption@sjtu.edu.cn

E. Qi et al. (eds.), *Proceedings of 2013 4th International Asia Conference on Industrial Engineering and Management Innovation (IEMI2013)*,
DOI 10.1007/978-3-642-40060-5_5, © Springer-Verlag Berlin Heidelberg 2014

use a history-based method keep a record of how the model was built. When the operator changes parameters in the model and regenerates the part, the program repeats the operations from the history, using the new parameters, to create the new solid [1].

However, different CAD systems have different modeling ways even for the feature definition. A parametric model that was generated in one CAD system will not be recognized or called feature inherited in another CAD platform. In fact, this is very popular in shipbuilding and auto industry [1]. As you know, shipbuilding is an extremely complex business. The time between order and delivery is very short, which means that a huge amount of tasks must be performed in parallel, or in some other cooperation ways [2]. This due to the whole ship model may come from different platform such as Tribon, CADDS5 and UG. How to realize ship data exchange between those heterogeneous systems has become a main problem that engineer have to face. At present, the common way is to use standard file style, such as STEP, IGES that was provided by most CAD systems. While this traditional data exchange method did not took feature parameter and modeling history into account [3]. That means after such exchanging, the original modeling features have been lost and model could not be regenerated in the destination CAD system. To edit or change the model parameters will become impossible.

The objective of this research is to fulfill the data exchange of modeling data of hull between Tribon and UG systems, two important CAD systems in shipbuilding and auto industry, via the XML neutral file.

2 Feature-Based Data Exchanging

By analyzing the modeling framework of Tribon and UG systems, we got to know that the traditional data exchange method by STEP or IGES is useless for feature inherited. Se we define a special neutral file style of our own base on XM. This neutral file can be used to save the ship hull data exported from Tribon. On the other hand, it can be also read by the destination CAD platform after customized development. In this method, any feature-based model can be trace back to its modeling history, so we can realize data exchanging process by rebuilding modeling history tree in different CAD system.

2.1 Frame Structure

Feature-based data exchanging aims at feature information sharing between different CAD systems. The whole exchanging process can be divided into two parts as shown as follows Fig. 1.

There are two steps for this exchanging process. One is pre-processor and another is post-processor. Pre-processor mainly used to read modeling data from local system as far as Tribon is concerned. During this process, these extracted data will

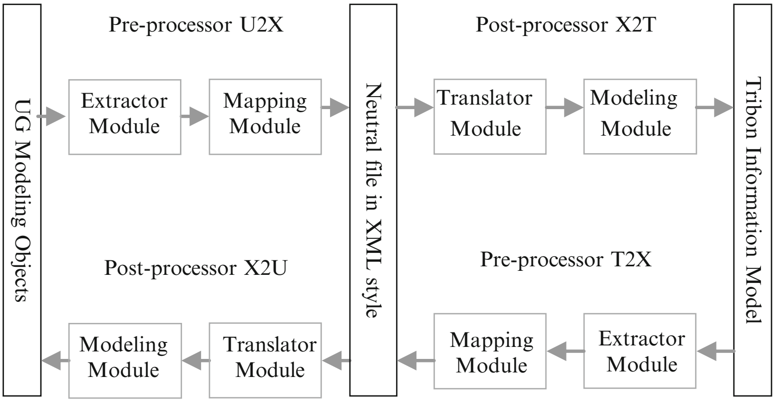


Fig. 1 Feature-based data exchanging framework

be organized according to a mapping module and then exported in a neutral XML file. By contraries, post-processor mainly used to translate these neutral files and fulfill the model regeneration in the target system.

2.2 Development Tool

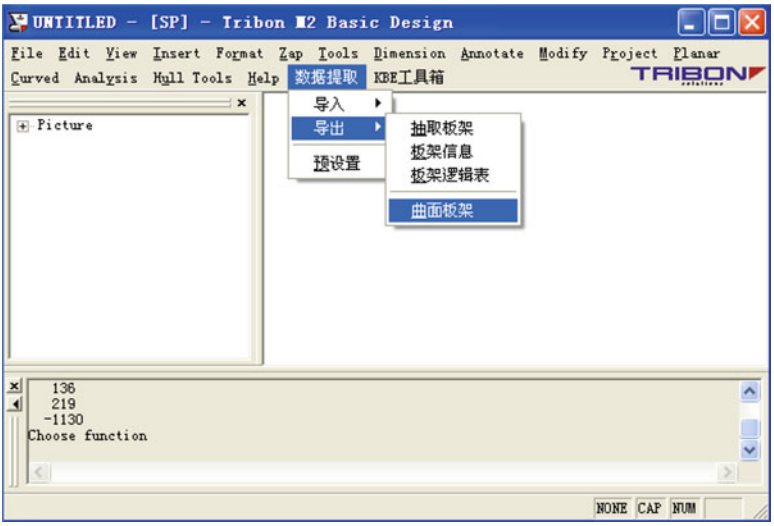
Like many modern CAD systems, Tribon provide user a built-in programming tool Vitesse [1]. Tribon Vitesse is a productive way to create customized, user-developed macros or programs in an object-oriented language. These macros are then available within the interactive Tribon application through a special function on the menu. A customized interface that we developed by this tool can be shown as follows Fig. 2.

Tribon Vitesse incorporates the programming language Python [1]. This syntax includes all normal features in a programming language such as branching, loops, functions, etc., as well as full support for the object-oriented programming. The Python language itself gives the use the full freedom to work in a traditional function-oriented fashion. Its implementation in Vitesse, however, requires the programmer to gain also the basic skills in the object–oriented approach, because the Tribon entities and activities have been modeled in the Python language as classes as its methods.

By calling the corresponding utility functions that was encapsulated by Vitesse, the programmer can easy access Tribon product information model and its database.

2.3 Ship Hull Exchanging Objects

Tribon product information model can be regarded as a ship database containing all information about a specific project. The database is object-oriented in the sense


UNTITLED - [SP] - Tribon M2 Basic Design
File Edit View Insert Format Zap Tools Dimension Annotate Modify Project Planar
Curved Analysis Hull Tools Help 数据提取 KBE工具箱
TRIBON
Picture
导入
导出
预设置
抽取板架
板架信息
板架逻辑表
曲面板架
NONE CAP NUM

```
 136
 219
-1130
Choose function
```

Fig. 2 Screenshot of the T2U client developed by Tribon Vitesse tool

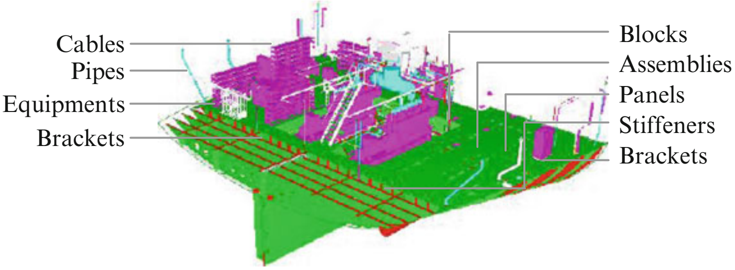


Fig. 3 Ship hull and outfitting topological structure

that all design and production data is stored as “objects”. These “objects” are all the types of physical items found in shipbuilding, e.g. [4] (Fig. 3).

There is a hierarchy within the product information model, which helps the user keep track of the information and produces production information more efficiently [5]. An example of this hierarchy is shown in the following diagram.

2.4 *XML Neutral Files*

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a certain tag format that is both human-readable and machine-readable. It is a data format used extensively to describe structured data [6].

Table 1 A glance at XML neutral file for ship hull

Keywords (Class1, Class2, Class3, Class4, Class5)	Feature
Ship Hull Block Panel Tran matrix	
<Ship name="dc_18000">	Seam
<hull>	
<block name="SP505" id="1">	Contour
<panel name="SP505-41" id="1">	Notch
<tran_matrix>[0, 3, 0,1,0...]</tran_matrix>	
<main_plate>	Hole
<thickness>20.0</thickness>	Cutout
<nseg>4</nseg>	
<seg id="1">	Bracket
<start_point>5,1,0</start_point>	
<end_point>5,1,0</end_point>	Flange
<amplititude>0,0</amplititude>	Pillar
</seg>	
</main_plate>	Stiffener
</panel>	Plate
</block>	
</hull>	Sub panel
</Ship>	

Since the XML documents are standard test files, it is virtually possible to handle them, using standard Python abilities of handling files. This becomes, however extremely difficult with the growing complexity of the document [7]. Therefore, we have to develop a translation module to facilitate the handling of the XML. For example, we can use the following code to obtain an existing XML document in TRIBON development circumstance [8].

From xml.dom.minidom import parse

```
try:
     doc=parse(“D:\\test.xml”) #Build XML document
try:
              ... #Work on the document
     finally:
              doc.unlink() #Prepare for cleanup
except:
     kcs_ui.message_confirm(“Error reading file!”) [1]
```

In this paper, we use a kind of XML neutral file to record ship hull modeling history data. These neutral files are composed by some well-regulated macro unites [9]. In this method, Modeling data are described by modeling macro (modeling history tree) instead of geometry element, such as point, line and surface. Macros unite is a data structure with two-dimensional characteristic. Each macro unites will give expression to a certain hull structure: parallel unites describe the structure that lies in the same class, while longitudinal unites describe the structure that lies in different class [10]. We put the ship hull data into five classes: Ship, Hull, Block, Panel, Tran matrix. This can be shown in the following Table 1.

2.5 Co-ordinate Translation

In the ship hull creation process, we adopt a variety of positioning ways. For example, we work with three kinds of co-ordinate systems in Tribon platform. Drawing co-ordinate system will be used under screech circumstance, global co-ordinate system and panel co-ordinate system will be used when we need to create a solid feature [11]. The coordinate variables shown as follows:

- drawing co-ordinate system (U, V)
- global co-ordinate system (X, Y, Z)
- panel co-ordinate system (U, V, W)

In addition, Tribon and UG NX differ from the definition of the relative coordinate system. So, during the model regeneration process in the destination platform UN NX, we have to deal with the co-ordinate translation problems. If the coordinate transformation is linear, we need to change source coordinates from (x, y) to a new pair of coordinates (x′, y′). The conversion express can be written as a matrix equation. The most general linear coordinate transformation of R^2 has the following form:

$$\begin{aligned} x' &= \mathrm{a}x + \mathrm{b}y \\ y' &= \mathrm{c}x + \mathrm{d}y \end{aligned} \tag{1}$$

Where a, b, c, and d are real-valued constants with $\mathrm{ad} - \mathrm{bc} \neq 0$. Now we can easily see that this transformation may be expressed as a matrix equation as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{c} & \mathrm{d} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2}$$

What if we want to return to the old coordinates? Then we apply the inverse coordinate transformation, given by:

$$\begin{bmatrix} x \\ y \end{bmatrix} = (\mathrm{ad} - \mathrm{bc})^{-1} \begin{bmatrix} \mathrm{d} & -\mathrm{b} \\ -\mathrm{c} & \mathrm{a} \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{3}$$

These equations may be written symbolically as $\mathrm{x}' = \mathrm{Ax}$ and $\mathrm{x} = \mathrm{A}^{-1}\mathrm{x}'$. It is easy to see that the first of these equations implies the second; just multiply both sides on the left by A^{-1}. When written out as a system of linear equations, the above matrix equation becomes:

$$\begin{aligned} \mathrm{x} &= \left(\mathrm{dx}' - \mathrm{by}'\right) / (\mathrm{ad} - \mathrm{bc}) \\ \mathrm{y} &= \left(-\mathrm{cx}' + \mathrm{ay}'\right) / (\mathrm{ad} - \mathrm{bc}) \end{aligned} \tag{4}$$

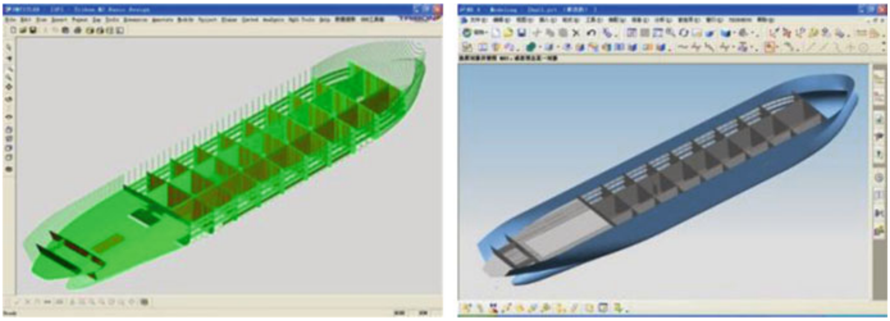


Fig. 4 Comparison for Tribon hull model and regenerated model in NX

2.6 Application Examples

In this paper, we use the programming language Python which has been integrated within Tribon as the customized development tools. The syntax includes all normal features in a programming language such as branching, loops, functions, etc., as well as full support for the object-oriented programming [12].

Within the research project, two customized clients are implemented as prototypes. These are used for demonstration and special design purposes. Both are implemented in Python. T2U client is used for visualization and to generate the user interface of hull data export [13] (Fig. 4).

For the usage in hull structural design, this client is integrated into the Tribon hull-drafting module using the Vitesse application-programming framework. This client program is used as named, i.e. [14]. It provides an interface or called tools to the knowledge based design and related information stored in the standards database. Basic workflows like the standards conformant construction of brackets and cutouts are implemented. For a design task like the definition of a bracket attribute values are entered automatically as far as possible thus reducing the number of user inputs required [15].

Due to the open design and the consolidation of the complete application logic into the server the development of additional clients that integrate CAD–systems or further applications into the system is possible [16].

3 Discussion

In this paper, we talk about the possibility for feature-based geometry data seamless integration between heterogeneous CAD systems. The author trace ship hull modeling history in Tribon and via the XML neutral file, a same 3d model has been regenerated in another system UG. In case of ship industry, this method was proved to be effective.

Currently, the data exchange method apply is a first prototype focused on design, storage and retrieval of standardized parts. It is a highly configurable and flexible system. By using the data model described, engineers have a unique data source for the steel structural design process. The access to additional information using entity information derived classes allows for a further consolidation and a tighter integration of all relevant pieces of information [17]. Further work is currently carried out on the closer integration of a 12 rule–based computation system for the dynamic evaluation of parameters or conditions. Advanced workflows for a guided wizard-based design are developed.

4 Conclusion

By feature-based data exchanging method, a ship hull model has been regenerated successfully in UG. The experiments show that this data exchange method is fairly high efficiency and all modeling features have been inherited in the destination system. This method can also be widely used in other feature-based CAD platforms such as Solidworks, CADDS5 and CATIA [18].

Acknowledgment The work presented in this paper is supported by the China National Engineering Research Center of Die & Mold CAD, Shanghai Jiao tong University. This research project would not have been possible without the support of many people. The author wishes to express his gratitude to Prof. Zhou who was abundantly helpful and offered invaluable assistance, support and guidance. Deepest gratitude is also due to Prof. Ruan and all project group members.

References

1. Hyeong–cheol Kim et al (2005) Introduction to DSM hull modelling system “COSMOS” based on TRIBON M3, ICASS 2005, Busan
2. Wang Y, Chen L, Huang Z, Wu J (2000) A history-independent modelling-oriented approach to solve geometric, constraints between features in 3D space. Int J Adv Manuf Technol 25:334–342
3. Keyser J, Culver T, Foskey M, Krishnan S, Manocha D (2002) ESOLID—a system for exact boundary evaluation. Comput Aided Des 36:175–193
4. Young–Soon Yang (2005) A study on the web–based distributed design application in the preliminary ship design, ICASS 2005, Busan
5. Rao Nalluri SRP (1993) Knowledge-based gluing operators for feature-based modeling. Comput Ind 23:129–138
6. Antkiewicz M (2004) Feature modeling plug-in for eclipse. In: Proceedings of the workshop on eclipse technology eXchange, Vancouver, pp 67–72
7. Ferraiolo D, Kuhn D, Chandramouli R (2007) Role-based access control, 2nd edn. Artech House, Boston
8. Triantaphyllou E, Mann S (1995) Using the analytic hierarchy process for decision making in engineering applications: some challenges. Int’l J Ind Eng Appl Pract 2(1):35–44

9. Osborn S, Sandhu R, Munawer Q (2000) Configuring role-based access control to enforce mandatory and discretionary access control policies. ACM Trans ISS 3(2):85–106
10. Lee SH (2005) A CAD–CAE integration approach using feature-based multi-resolution and multi-abstraction modeling techniques. Comput Aided Des 37(9):941–955
11. Alvares AJ, Ferreira JCE (2008) A system for the design and manufacture of feature-based parts through the Internet. Int J Adv Manuf Technol 35(7–8):646–664
12. Van den Berg E, Bronsvoort WF, Vergeest JSM (2002) Freeform feature modelling: concepts and prospects. Comput Ind 49(2):217–233
13. Borutzky W (1999) Bond graph modeling from an object oriented modeling point of view. Simul Pract Theory 7(5–6):439–446
14. Isoda S (2001) Object-oriented real-world modeling revisited. J Syst Softw 59(2):153–162
15. Eelco VDB, Hilderick AVDM, Bronsvoort WF (2003) Specification of freeform features. In: International conference on shape modeling, ACM, New York, pp 56–64
16. Allen B, Curless B, Popović Z (2003) The space of human body shapes: reconstruction and parameterization from range scans. ACM Trans Graph 22(3):587–594
17. Wang CCL, Tang K, Yeung BML (2005) Freeform surface flattening by fitting a woven mesh model. Comput Aided Des 37(8):799–814
18. Wang CCL, Wang Y, Yuen MMF (2005) Design automation for customized apparel products. Comput Aided Des 37(7):675–691