Feature-Based Design Modification in Co-Assembly Design

C. Lu[1], J. Y. H. Fuh[1], Y. S. Wong[1], W. D. Li[2] and Y. Q. Lu[2]

[1]National University of Singapore, mpefuhyh@nus.edu.sg
[2] Singapore Institute of Manufacturing Technology

ABSTRACT

This paper discusses the feature-based design modification in a collaborative assembly (co-assembly) design environment. A hierarchical co-assembly representation model has been proposed and a proposed assembly feature design scheme has been given to resolve the co-assembly design issue. In order to realize the design modification propagation control, an XML schema was developed to transfer the assembly design information by defining each feature using the XML format based on the co-assembly representation model proposed. The detailed design modification propagation control mechanism has been demonstrated through an example case study. Furthermore, a system framework suitable for realizing the co-assembly design modification is also proposed and developed.

Keywords: Collaborative assembly design; assembly feature; XML; design modification propagation

1. INTRODUCTION

During product design, assembly design enables designers to provide a complete concept of a product that usually consists of many different parts. Generally, in traditional computer-aided assembly design, each part is designed in a standalone computer system and then assembled into a sub-assembly or a more complex product by one or a group of designers in the same location. With the development of the Internet and communication technology, more and more products are designed and manufactured in different locations to meet the fast-changing market requirements. Rezayat [1] reported that about 50-80% of the components in a product from Original Equipment Manufacturers (OEMS) are outsourced to external suppliers geographically dispersed. Hence, products are usually divided into several sub-assemblies or even more detailed parts and are assigned to many designers located in different geographical sites to speed up the design process.

In such a collaborative design environment, when every designer finishes designing his parts according to the initial design requirements, those parts should be assembled together correctly. However, if a designer modifies his design after the assembly process is finished, he does not know how the modification can affect the other parts designed by other designers because the whole assembly relationship with other associated parts are not known to him. Hence, it is unavoidable that some conflicts arise during the co-assembly design process. Therefore, it is imperative to develop a methodology to support the proper design modification of each part when the mated parts have been modified in a co-assembly design environment. In order to address this problem, this paper proposes a novel feature-based hierarchical data representation for co-assembly design, gives a new definition of assembly features, and studies the design modification propagation control mechanism.

2. LITERATURE REVIEW

In a co-assembly design process, one of the key researches is to develop a proper assembly representation approach to specify the relationship between different parts.

Conventionally, the assembly feature is used to represent the assembly relationship between different parts in an assembly, but the collaboration between different designers is not considered [2-8]. Therefore, these developed assembly representations are not adaptive to the assembly design in a collaborative design environment.

In order to address this problem, some researchers proposed new definition of assembly features. Shyamsundar and Gadh [9] defined an assembly feature as a property of an assembly unit with respect to other components. In addition, the authors proposed the interface assembly features as a subset of the assembly features. These interface assembly features are

considered as hard constraints and cannot be modified unilaterally by the designer. It can only be changed through the negotiation with other designers. Therefore, modifications of an assembly feature can only be executed when all corresponding designers in different geographical locations are working simultaneously, and the real-time design modification cannot be realized when some designers are absent sometime.

Chen et al. [10] proposed a co-assembly representation including Master Assembly Model (MAM) and Slave Assembly Model (SAM). MAM is a complete representation stored in the server, and SAM is a simplified version of MAM used for visualization in the client. The MAM includes the composite component information, atomic component information and link entity information. This representation can realize the co-assembly modeling, but it cannot realize the real-time design modification in a collaborative design environment either.

For some research works related to the real-time design modification in a collaborative design environment, Bidarra et al. [11] and Noort et al. [12] presented a multiple-view feature modeling approach to integrate part design and assembly design. This approach integrates a part's detailed design view and the assembly design view by linking the part model with the associated components in an assembly model, thus enables the system to update a part's detailed design when the associated component modified, and vice versa. However, it focuses on the modification propagation between a part's detailed design and associated component design in the assembly design environment, but does not consider how the design modification of one part affects the other parts designed by other partners in the co-assembly design environment. It does not consider the network-based working relationship between different designers.

Mori and Cutkosky [13] proposed an agent-based architecture and a set of algorithms to coordinate the actions of different design agents using the theory of Pareto optimality. The agents are reactive and they can track and respond to changes in the state of the design when one designer changes his design and thus brings in the conflicts. In each design agent, there is a design process manager which is responsible for recording the design process, and manages rule-based knowledge to coordinate and control the actions of agents. However, the communicating protocol to exchange information between the design agents is simple and limited, so that this architecture is not suitable for the more complex co-assembly design.

3. AN ASSEMBLY REPRESENTATION MODEL FOR COLLABORATIVE DESIGN

3.1. Feature-based Hierarchical Co-assembly Representation

In co-assembly design process, how to represent the assembly is very important to realize the real-time design modification and communication between different designers geographically dispersed. It requires represent not only the assembly relationship between features of different parts, but also the network-based working relationship between different designers.

Based on the above requirement, a feature-based hierarchical co-assembly representation has been proposed as shown in Figure 1. This hierarchical data structure organizes an assembly as a compound of sub-assemblies, and the sub-assemblies are composed of several parts. The parts can be divided into a number of form features that are composed of boundary entities using Boolean algorithms. In addition, a part has one element "client ID" which indicates the designer for this part. In the following classification, each form feature has two basic elements: "modification attribute" and "mating constraints". The modification attribute includes two states: "changeable" and "constant". The changeable state means the geometrical shape of this form feature or its position in the assembly can be changed after the assembly design finished, and the constant state means the both above must be kept the same. Typically, the constant attribute is often used in some critical and standard parts in the assembly. The "ClientID with rights to modify" indicates the designers with rights to modify the feature. The other basic element, i.e. "mating constraints", has two attributes: "no" and "yes". If "no", this form feature does not have any assembly relationship with other form features; otherwise, "yes" means the feature has the assembly relationships. If a feature has assembly relationships, the "mating condition" of this feature further includes sub-elements: "feature ID mated with", "geometric constraints", "degrees of freedom" and "motion limits". The "feature ID mated with" points to the form feature mated with it, and through this form feature, the corresponding part ID and client ID can be searched and retrieved.

This hierarchical data structure not only represents the longitudinal "part-of" relationships, but also the latitudinal "mating" relationships between different form features belonging to different parts which are designed by the different designers geographically dispersed. In addition, by assigning the modification attribute and modification rights of each form feature, the design modification propagation routes can be built up when the design of a form feature is modified in the co-assembly design environment.

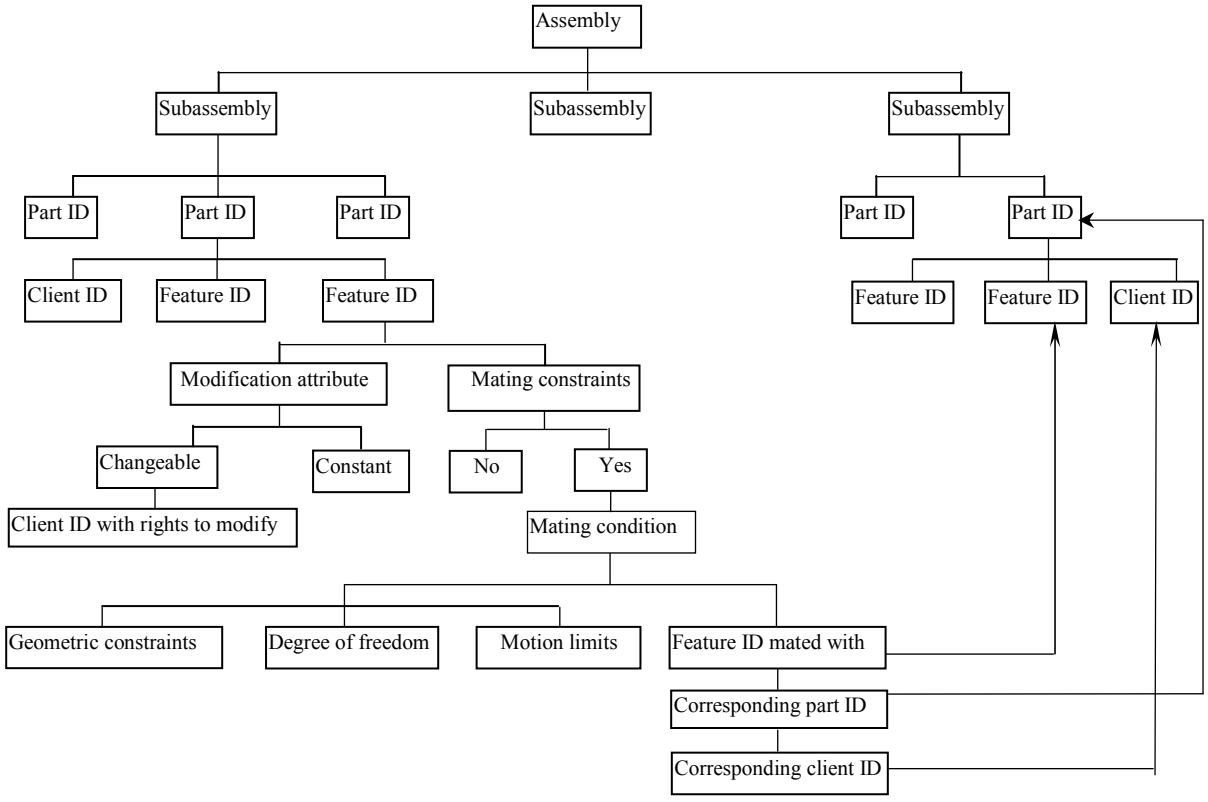


Fig. 1. Feature-based hierarchical co-assembly representation

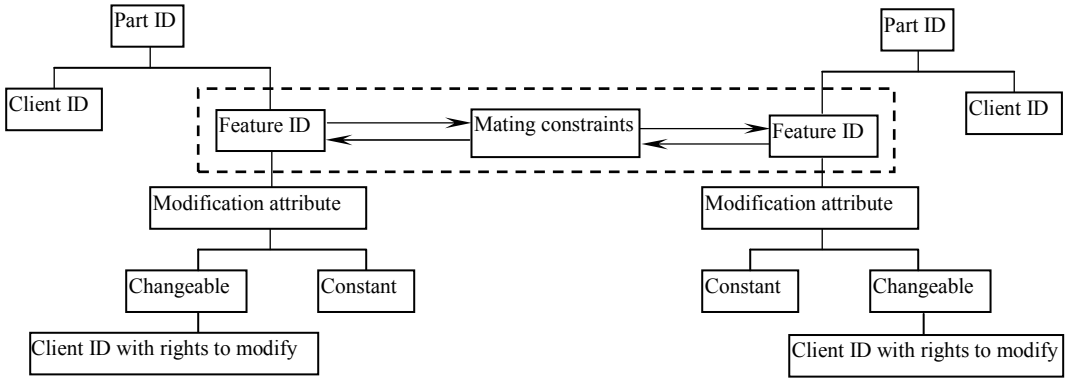


Fig. 2. Structure of assembly features

3.2. A Definition of Assembly Feature for Collaborative Design

The previous works are not suitable for the real-time design modification in co-assembly design environment. In our assembly representation approach, the defined assembly feature includes not only the relationship between two form features, but also the relationship between two different designers in different geographical locations. From Figure 1, the definition of assembly features is extracted and shown in Figure 2. Only the form feature that has the mating relationship with others can be used to combine into the assembly feature.

In our new definition of assembly features, the assembly feature is divided into two portions — internal assembly feature and external assembly feature. The internal assembly feature lies within the dashed rectangle (see Figure 2), and it represents the assembly relationships between two form features through mating constraints, which often include geometric constraints, degree of freedom and the motion limits. The assembly feature outside the dashed rectangle is the external assembly feature that includes the modification attributes, corresponding part ID and client ID of the form features.

Basically, the internal assembly feature is the same as the traditional assembly feature, and the main function is to define the assembly relationships between form features. However, the external assembly feature defines the other two important factors in the co-assembly design environment: (1) corresponding part ID and client ID, and (2) modification attributes.

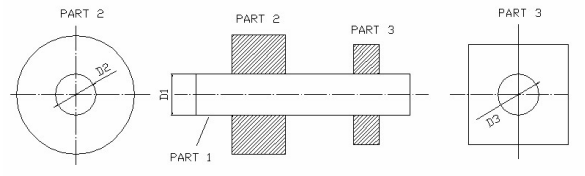


Fig. 3. Assembly consisting of thee parts

4. ROLE OF CO-ASSEMBLY REPRESENTATION IN REAL-TIME DESIGN MODIFICATION

In this section, an example will be given to illustrate the role of this new assembly representation model for real-time design modification in the co-assembly design.

In Figure 3, a simple assembly consists of 3 parts - Part1, Part2 and Part3, and each part is to be designed by designers geographical dispersed, i.e. Client1, Client2 and Client3, respectively. The cylinder feature of Part1 has the mating relationship with the hole feature of Part2 and the hole feature of Part3 respectively. Hence, there are two assembly features in this assembly: one between Part1 and Part2, and the other between Part1 and Part3. The designer of Part1 sets the modification attribute of the cylinder feature of Part1 as "changeable". The designer of Part2 also sets the modification attribute of the hole feature of Part2 as "changeable". The designer of Part3 sets the modification attribute of the hole feature of Part3 as "constant". These designers do not assign the modification rights to each other, then the assembly feature between Part1 and Part2 can be represented in Figure 4, and the assembly feature between Part1 and Part3 can be represented in Figure 5.

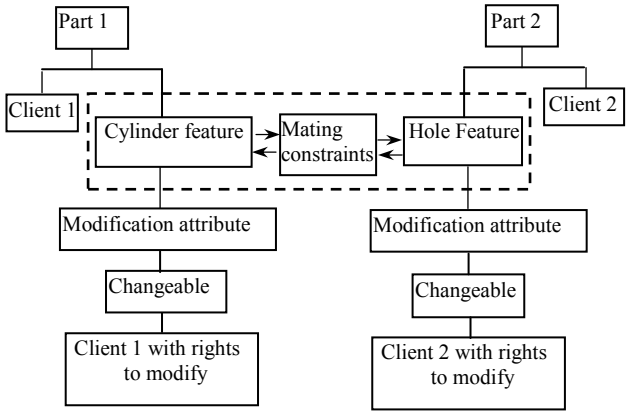


Figure 4. Assembly feature between Part1 and Part2

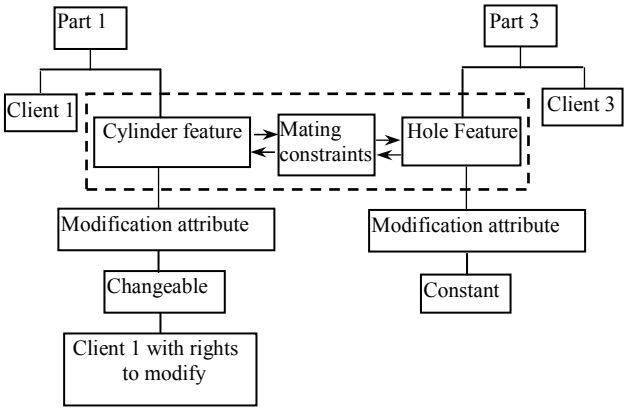


Figure 5. Assembly feature between Part1 and Part3

In the co-assembly design process, after the assembly modeling of these three parts is completed, if the designer in Client2 modifies the design of the hole feature of Part2 by increasing the diameter of hole D2 to D2', through the assembly information and the working relationship defined in the assembly feature between Part1 and Part2 in shown Figure 4, this change should first be propagated to the cylinder feature of Part1, and the diameter of the cylinder D1 should be increased to D1'. Then, through the assembly information and the working relationship defined in the assembly feature between Part1 and Part3 in shown Figure 5, the modification of the cylinder feature should be propagated to the hole feature of Part3, and its diameter should also be increased. However, since the modification attribute of this hole feature is assigned "constant", the diameter D3 must maintain constant. Finally, the modification in Part2 and the two assembly features defined jointly decide the design modification as follows:

(1) The designer in Client1 should modify Part1 into a step cylinder with the diameter D1' and D1.
(2) The designer in Client3 should keep Part3 remain unchanged.
The design modification results are illustrated in Figure 6.

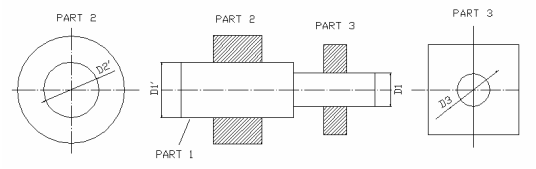


Figure 6. The design modification results (1)

Another condition defined by designers is that the assembly feature between Part1 and Part2 keeps constant as in shown Figure 4, but the assembly

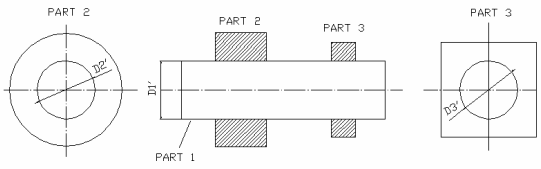


Figure 7. The design modification results (2)

feature between Part1 and Part3 is changed, as the modification attribute of the hole feature in part3 is set "changeable" by the designer in Client 3.

Then the modification in Part2 and the two defined assembly features should jointly decide two design modification schemes. The first one is the same as that shown in Figure 6:

(1) The designer in Client1 should modify Part1 into a step cylinder with the diameter D1' and D1.

(2) The designer in Client3 should keep Part3 remain unchanged.

The second scheme is shown in Figure 7:

(1) The designer in Client1 should modify Part1 into a cylinder with the increased diameter D1'.

(2) The designer in Client3 should increase the hole diameter of Part 3 to D3'.

From this example, we can see that through different definitions of assembly features including internal and external assembly feature, the design modification of the form feature of one part can trigger different change propagation routes in the co-assembly design, and we can get the different design modification results of the other parts designed by other designers geographical dispersed.

5. DESIGN MODIFICATION PROPAGATION CONTROL MECHANISM

5.1 XML Representation

Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere [14].

Since XML data format is flexible to be specified by the user, it is suitable to embed and transfer some kinds of information across the Internet.

5.2. Using XML File to Exchange Information

In order to control the design modification propagation and realize real-time design modification, two kinds of XML file formats have been adopted to define different contents. Based on them, the design parameters and assembly information of features can be exchanged during the co-assembly design process.

Format 1: XML file for defining the design parameters of each feature

We use XML file to define the design parameters of each feature. When a designer input the parameters of each feature in a feature-based design working model in his client, the XML file defining the parameters can be written through the XML writer in the client. The

example of XML file defining the parameters of each feature of Part 2 shown in Figure 3 is illustrated in List 1, which includes two features: a hole feature and a cylinder feature.

Format 2: XML file for defining each feature and the assembly information

In order to transfer the assembly information using an XML file, we define each feature using the XML file format according to the feature-based hierarchical co-assembly representation proposed in Section 3. The XML file defining each feature and the assembly information are written through the XML writer in the client when assembly modeling is completed.

List 2 is an example of XML file that defines each feature and their assembly information in Part 2 shown in Figure 3.

5.3 XML File Parsing Process

When the feature defined in the above XML file- <featureID> "201" is modified, through the XML parser implemented we can extract the value of node <value> in the parent node <mating_constraints>; since it is "yes", it is an assembly feature with assembly relationship with others. Then we further extract the value of the node <feature_ID_mated_with>, which is the feature that has the assembly relationship with the modified feature and will be affected by the design modification. Otherwise, if the value of the node <value> in the parent node <mating_constraints> is "no", e.g. <featureID> "202", it is a feature without assembly relationship, and the modification of it cannot affect other features. All this process can be executed by the XML parser when the designer sends his XSL file to the parser according to his requirement. The parsing result- a HTML file including the modified feature information and the mated feature information, can be generated automatically when the parsing process is finished. Figure 8 is the parsing result of List 2 when <featureID> "201" is modified, and the result is displayed as a Web page.

```
<?xml version="1.0"?>
<part>
<feature>
<featureID>201</featureID>
<name>hole</name>
<length>50</length>
<diameter>30</diameter>
</feature>
<feature>
<featureID>202</featureID>
<name>cylinder</name>
<length>50</length>
<diameter>100</diameter>
</feature>
```

List 1. XML file defining design parameters of each feature in Part 2

```
<?xml version="1.0"?>
<part>
 <feature>
  <ID>
    <featureID>201</featureID>
    <partID>2</partID>
    <clientID>2</clientID>
  </ID>
  <modification_attribute>
   <value>changeable</value>
   <clientID_with_rights_to_modify>2</clientID_ with
_rights_to_modify>
  </modification_attribute>
  <mating_constraints>
   <value>yes</value>
  </mating_constraints>
  <mating_condition>
   <featureID_mated_with>"101"</featureID_mated_with
>
   <mating_type>"fit"</mating_type>
  </mating_condition>
 </feature>
<feature>
 ……
 </feature>
</part>
```

List 2. XML file defining each feature and the assembly information in Part 2

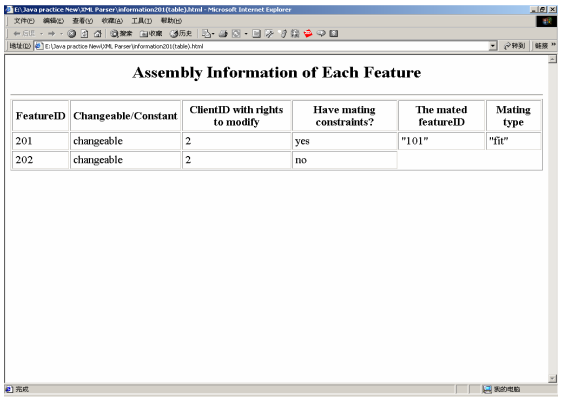


Fig. 8. Parsing result of XML file (List 2) when <featureID> "201" is modified

<featureID_mated_with> which is "The mated featureID" of the modified <featureID> "201" in the web page (Figure 8), we can search the corresponding XML files defining "The mated featureID", and then parse these files in the same way. In this example, the feature with the <featureID> "101" is affected by the modification of <featureID> "201", then the XML file defining <featureID> "101" will be parsed and the other corresponding XML files are searched. The flowchart of the whole parsing process is given in Figure 9.

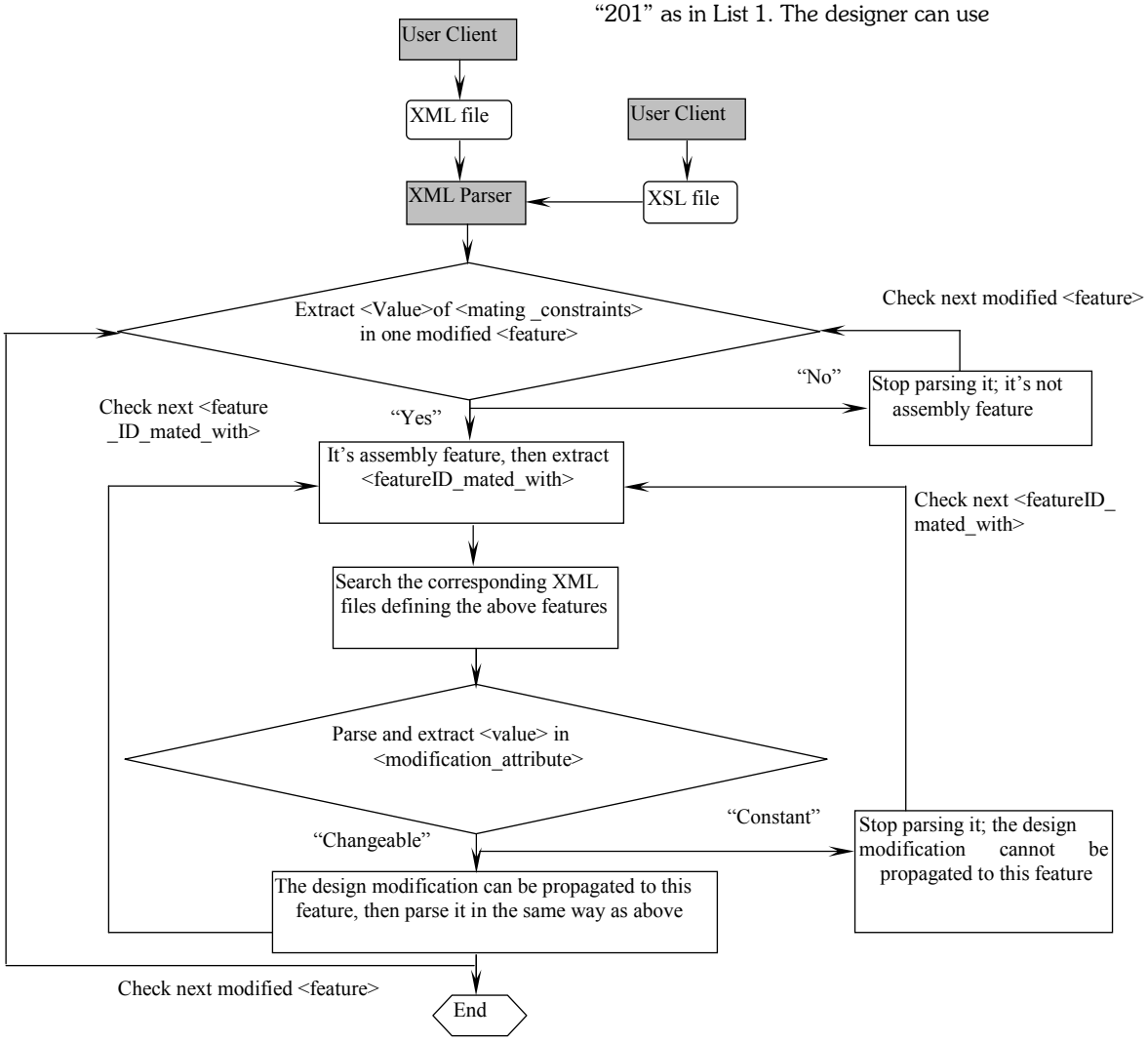


Figure 9. Flowchart of XML file whole parsing process

The XML file defining the <featureID> "101" is shown in Figure 10. This file defines each feature (only one feature) and their assembly information in Part 1 shown in Figure 3. Because the <value> in <modification_attribute> is "changeable", the <featureID> "101" would be affected by the modification, and this XML file should be parsed in the same way. (On the contrary, if <value> in <modification_attribute> is "constant", this feature cannot be modified. That is, we do not need to parse this file further, and the design modification cannot propagate through it to others.) The client who designed the <featureID> "101" will receive the XML file that defines the updated design parameters of <featureID> "201" as in List 1. The designer can use the new information in List 1 to modify the affected <featureID> "101". Using the same parsing process, the modification of <featureID> "101" can be propagated

to <featureID> “301” in Part 3 based on the assembly information defined in the XML file (Figure10).

The parsing result of the XML file defining the <featureID> “101” (see Figure 10) when <featureID> “101” is modified is displayed in Figure 11.

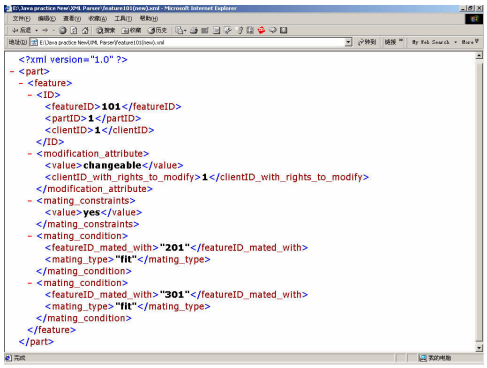


Fig. 10. XML file defining feature and assembly information in Part 1

6. CASE STUDY

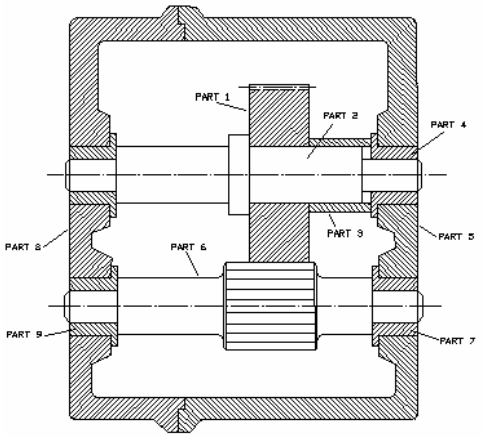
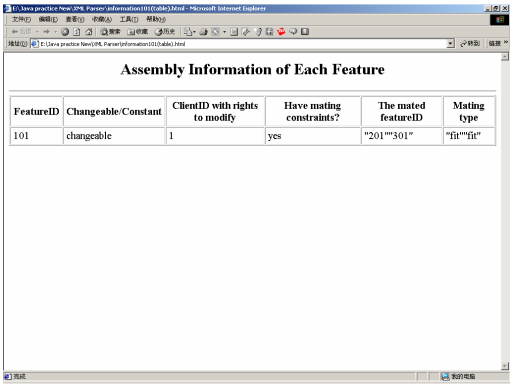


Figure 12. Simplified gearbox assembly diagram

Through the above parsing process, it builds a design modification propagation mechanism in co-assembly design.



Assembly Information of Each Feature

FeatureID	Changeable/Constant	ClientID with rights to modify	Have mating constraints?	The mated featureID	Mating type
101	changeable	1	yes	"201""301"	"fit""fit"

Figure 11. Parsing result of XML file in Figur10

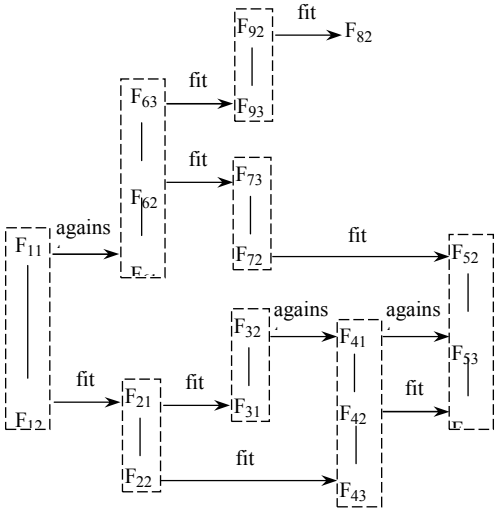


Fig. 13. The design modification propagation triggered by modification of F11 & F12

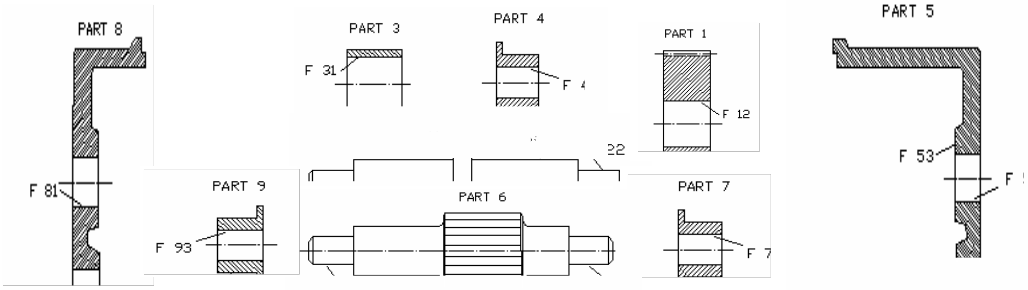


Figure 14. Some features of each part

In this section, we use a simplified gearbox assembly (Figure.12) to demonstrate the real-time design modification in co-assembly design. Figure 14 shows some features (only some features with mating constraints are marked) of each part designed by different designers geographically dispersed. Figure.13 shows the design modification propagation routes controlled by the design modification propagation control mechanism in section 5 due to the modification of F11 and F12. In this case, if the designer increased the diameter of the gear (PART 1) and its internal diameter, then F11 and F12 are modified, these modifications are propagated through the different routes. For instance, F21 is affected by the modification of F12, and the modification of F21 is propagated to F31, the designer of PART 3 should increase the diameter of F32 in order to keep a certain wall thickness, then this modification should be propagated to F41. The designer of PART 4 increase the diameter of F41 to ensure the contact with F32 and this modification is further propagated to F53.

The above design modification propagation conditions are based on the assumption that modification attribute of each feature is changeable, otherwise, the propagation will be stopped at any feature with the "constant" modification attribute.

7. SYSTEM IMPLEMENTATION

According to the problem stated in this paper, a prototype system has been developed and the system framework is shown in Figure 15. It is a three-tier client-server structure that includes modeling server, design client and Apache Web server.

The modeling server executes the modeling function to realize the part and assembly modeling. It communicates with the design clients through Java RMI (Remote Method Invocation). When the design client sends the designing order to the modeling server through Java RMI, the modeling server finishes the modeling by calling OpenCascade modeling kernel [15] through JNI (Java Native Interface). The modeling result is exported to the VRML file and feed back to the client for visualization in the Java 3D environment.

In the design client, when the designer input the design parameters of the feature in a feature-based design model, these parameters are written into an XML file as in List 1 through the XML writer in the client. In addition, the XML files defining the feature and the assembly information as in List 2 are also written through the XML writer in the client when the assembly modeling is finished. When a feature is

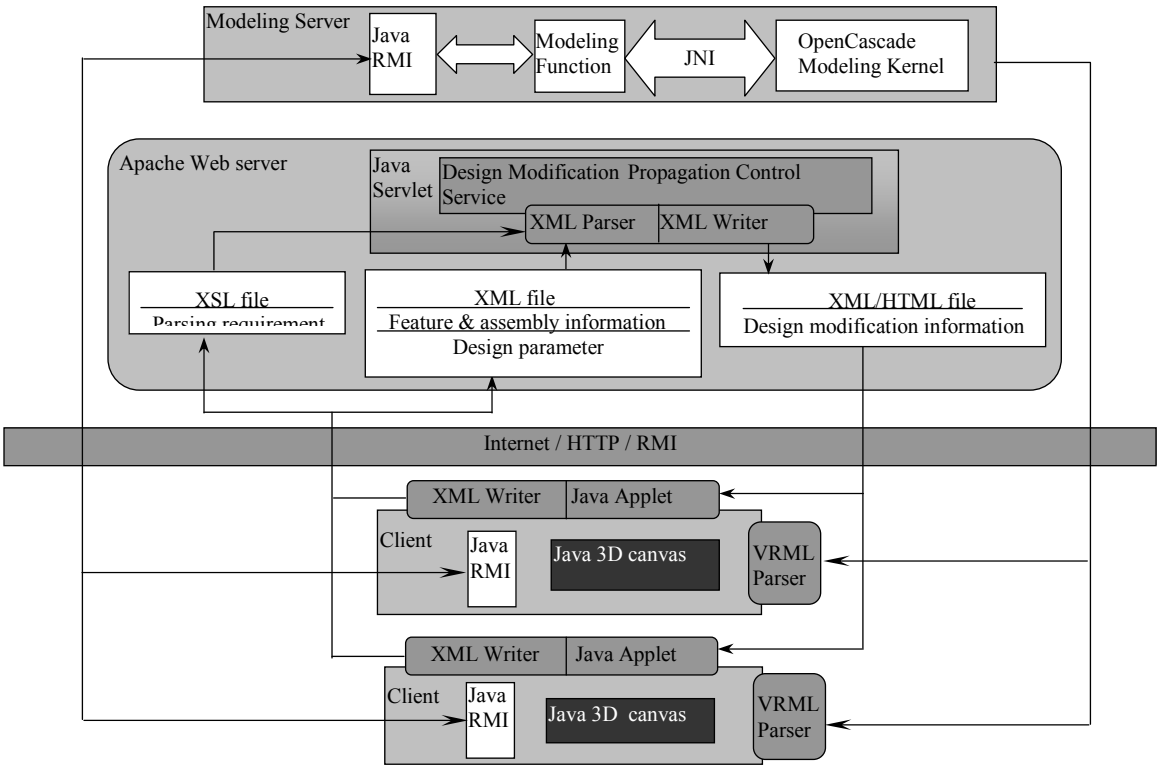


Figure 15. System framework

modified in one client, two types of XML files and the XSL file defining the parsing requirement by the designer are transferred to the Apache Web server for the design modification propagation control service. The result, an XML/HTML file embedded with the design modification information will transfer to the client due to the features designed by it are affected by the modification of other features. The detailed design modification propagation control mechanism has been illustrated previously in section 5.

8. CONCLUSIONS

This paper discusses the feature-based design modification in co-assembly design. A set of methodologies have been developed to resolve the problem and to maintain the consistency of the whole assembly.

The main contributions of this work can be summarized as follows:

(1) Through a feature-based co-assembly representation model and a new assembly feature scheme, the assembly relationship between different parts and the working relationship among different designers geographically dispersed can be built up. In addition, the assembly feature can help decide the design modification propagation routes in the co-assembly design process.

(2) In order to transfer the assembly design information, an XML schema has been adopted based on the proposed co-assembly representation model, and an XML parsing mechanism was developed to realize the design modification propagation control.

(3) The system framework suitable for realizing the real-time design modification in a co-assembly design environment has also been developed and demonstrated.

9. REFERENCES

[1] M. Rezayat, The enterprise-web portal for life-cycle support, Computer-Aided Design, Vol.32, No. 1, 2000, pp 85-96.

[2] J.J. Shah, M.T. Rogers, Assembly modeling as an extension of feature-based design, Research in Engineering Design, Vol.5, 1993, pp 218-237.

[3] X.G. Ye, J.Y.H. Fuh and K.S.Lee, Automated assembly modeling for plastic injection moulds, The International Journal of Advanced Manufacturing Technology, Vol.16, 2000, pp 739-747.

[4] W. van Holland, W.F. Bronsvoort, Assembly features in modeling and planning, Robotics and Computer Integrated Manufacturing, Vol.16, 2000, pp 277-294.

[5] T.L. De Fazio, A prototype of feature-based design for assembly, In: Ravani B, editor, ASME Advances in Design Automation, Chicago, IL, USA, 1990, pp 9-16.

[6] K. Lee, G. Andrews, Inference of positions of components in an assembly: part 2, Computer-aided Design, Vol.17, No. 1, 1985, pp 20-24.

[7] R. Sodhi, J.U. Turner, Representing tolerance and assembly information in a feature-based design environment, In: Gabriele GA, editor, Proceedings ofthe ASME Design Automation Conference, vol. DE-vol. 32-1, Miami, Florida, USA, 1991, pp 101-108.

[8] J.J Shah, R. Tadepalli, Feature based assembly modeling, In:Gabriele GA, editor, Proceedings of the ASME International Computers in Engineering Conference, vol. 1, San Francisco, California, USA, 1992, pp 253-60.

[9] N. Shyamsundar, R. Gadh, Internet-enabled collaborative product design with assembly features and virtual design space, Computer-Aided Design, Vol.33, 2001, pp 637-651.

[10] L. Chen, Z. Song, L. Feng, Internet-enabled real-time collaborative assembly modeling via an e-assembly system: status and promise, Computer-aided Design (in press).

[11] R. Bidarra, N. Kranendonk, A. Noort and W.F. Bronsvoort, A collaborative framework for integrated part and assembly modeling, In: Proceedings of Solid Modeling '02 - Seventh Symposium on Solid Modeling and Applications, 17-21 June, ACM Press, NY, 2002, pp 389-400.

[12] A. Noort, GFM Hoek and W.F, Bronsvoort, Integrating part and assembly modeling, Computer-aided Design, Vol.34, No. 12, 2002, pp 899-912.

[13] M. Toshiki and R.M. Cutkosky, Agent-based collaborative design of parts in assembly, Proceedings of the DETC'98, ASME Design Engineering Technical Conference, September 13-16, 1998, Atlanta, Georgia, USA.

[14] http://www.w3.org/xml

[15] http://www.opencascade.com/products/