Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2008
August 3-6, 2008, Brooklyn, New York, USA

DETC2008-49589

AGENT BASED VARIATION PROPAGATION FOR COLLABORATIVE TOP-DOWN ASSEMBLY DESIGN

Shuting Zhang
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, 310027, P.R.China
(Ph: 086-571-87951045)
zhangshuting@cad.zju.edu.cn

Xiang Chen
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, 310027, P.R.China
(Ph: 086-571-87951045)
xchen@cad.zju.edu.cn

Shuming Gao[*]
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, 310027, P.R.China
(Ph: 086-571-87951045)
smgao@cad.zju.edu.cn

Youdong Yang
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, 310027, P.R.China
(Ph: 086-571-87951045)
yyoudong@cad.zju.edu.cn

ABSTRACT

The design of a complex mechanical product is usually a top-down process carried out by different teams or designers that are geographically distributed. A systematical variation propagation mechanism is very important to fully support such a design process. In this paper, based on the framework for collaborative top-down assembly design previously proposed by the authors, an agent based approach is presented for addressing variation propagation for collaborative top-down assembly design. The approach achieves variation propagation during the collaborative top-down assembly design through the interaction and cooperation of the agents located at the clients and server. To make the variation propagation automated and intelligent, four kinds of variation reasoning including hierarchical variation reasoning, engineering constraint variation reasoning, feature variation reasoning, and assembly constraint variation reasoning are identified, and the corresponding algorithms are developed and utilized. Meanwhile, a distributed assembly model is put forward to effectively support the design variation propagation for the collaborative top-down assembly design. The approach is implemented and a variation propagation example is given.

Key Words: Top-down, Collaborative assembly design, Agent, Variation propagation

1 INTRODUCTION

The design of a complex mechanical product, especially the innovative design, usually takes place in a top-down fashion carried out by different teams or designers that are geographically distributed. It is widely agreed that the design process involves requirements analysis, conceptual design and detailed design. After the concept design is completed during the design process, the designers' main task is to collaboratively accomplish the assembly design based on the concept design.

One of the most important problems of the collaborative top-down assembly design is to ensure the consistency of the distributed assembly model. Since the product assembly model is distributed at different locations and there are complex correlations in the assembly model such as assembly relationships and parametric constraints, a modification can bring a series of variations to the distributed assembly model. Therefore an effective design variation propagation mechanism is of great importance. Once any modification happens, the system should automatically infer all the related variations of

[*] Corresponding author: smgao@cad.zju.edu.cn

the assembly model together with their distributions and update the distributed assembly model in a suitable way.

To effectively achieve design variation propagation during the collaborative top-down assembly design, the following three issues need to be addressed:

1 A product assembly model able to effectively support the design variation propagation.

2. A variation propagation algorithm that infers all the necessary changes of the distributed assembly model caused by a variation made by a user and achieves these changes accordingly.

3. An architecture that suits for the realization of the variation propagation mechanism.

This paper proposes an approach to variation propagation for the collaborative top-down assembly design, which includes our solutions to all the three issues mentioned above. The goal of this approach is to achieve the automated and intelligent variation propagation during the collaborative top-down assembly design. The rest of the paper is organized as follows: Section 2 introduces related work; Section 3 gives a brief overview of our collaborative top-down assembly design method; Section 4 presents the variation propagation algorithm; Section 5 describes the implementation and shows an practical example of variation propagation; The last section concludes the contribution and future work of this paper.

2 RELATED WORK

The top-down design is an iterative evolutionary design process during which the design information inheritance and consistency management should be significantly considered. Mantyla[1] proposes a multi-layer product representation for the top-down design according to the stages of the design process; inheritance and consistency of product information such as geometric model and constraints are presented. Chung[2] presents a unified framework with a complete mathematical description of the product information model which allows downstream tasks to be performed without having to rebuild analysis-specific model. Brunetti[3] proposes an approach that incorporates a feature-based representation for capturing product semantics of the conceptual design and linking early design with part and assembly modeling. Noort[4] develops a system using enhanced multiple-view feature modeling. By means of feature views and ways to keep them consistent, the system enables requirements for part design to be taken into account during assembly design, and vice versa.

While the previous discussions focus on the consistency of product information of different design stages, the distributed consistency is of great importance for the collaborative top-down assembly design. Bidarra[5] presents a collaborative framework which not only offers possibilities to simultaneously work on independent tasks in a product development process, but also synchronous facilities to collaboratively design the same component while the distributed consistency of the product information is guaranteed. Li[6] develops a client/server framework which enables dispersed teams to accomplish a feature-based design task collaboratively. Based on feature-to-feature relationships, a distributed feature manipulation mechanism is proposed which enables all the clients' feature model to be consistent. Shyamsundar[7] presents a new geometric representation called AREP providing the designers with the ability to collaboratively perform real-time geometric modification, assembly constraints specification and concurrent design of different components/sub-assemblies. The Internet-enabled real-time collaborative assembly modeling system, called e-Assembly is presented by Chen[8] which allows a group of geographically dispersed designers to jointly build an assembly model in real time over the Internet.

Software agent technology promises much for collaborative product design. Some agent systems for CAD/CAPP/CAM are implemented such as OpenADE[9], DCS[10] and RAPID[11]. It is through agent interaction that the agent based product design system completes the design task. Mori[12] describes how design agents interact with each other, exchange design information and keep track of state information to assist with collaborative design. Also a coordination algorithm that corresponds to the tracking of Pareto optimality is proposed. Shi[13] develops the broking agents called CyberAgent for collaborations among distributed applications over the Internet. Based on a flexible relationship model called CyberWorkflow the parameters can be transferred among applications through data flow between CyberAgents. Wang[14] proposes a computational model of collaborative product design management aiming to improve the efficiency and effectiveness of the cooperation and coordination among participating disciplines based on agent technology.

In this paper an agent based approach is presented to support the variation propagation for collaborative top-down assembly design. The agent at the server reasons the complex design variations involved in collaborative top-down assembly design such as hierarchical variations, engineering constraint variations, feature variations, and assembly constraint variations. Automated and intelligent variation propagation is achieved through the interaction and cooperation of the agents located at the clients and the server.

3 COLLABORATIVE TOP-DOWN ASSEMBLY DESIGN

This section gives a brief overview of the framework for collaborative top-down assembly design previously proposed by the authors [15]. In the framework, the collaborative top-down product design starts after the concept design, which is divided into three main design phases: the layout design, the skeleton design and the detailed design.

1. Layout design. It is the design stage during which the abstract specification of the product is created containing the critical elements such as the key subassemblies and parts, the main assembly relationship and functional and structural constraints of the product. Based on the abstract specification of the product, the chairman assigns the subassemblies and parts to different designers according to the human resource.

2. Skeleton design. It is the design stage during which the 3D skeleton assembly model is created collaboratively by the designers undertaking different subassemblies or parts. The 3D skeleton assembly model contains product information such as the overall shapes and container spaces of the subassemblies and key parts, the assembly relationship specifications and key assembly constraints and some key parameters and so on.

3. Detailed design. At this design stage, the final assembly model is established collaboratively by the designers consisting of the detail geometric model and assembly relationships with geometric constraints and parametric constraints.

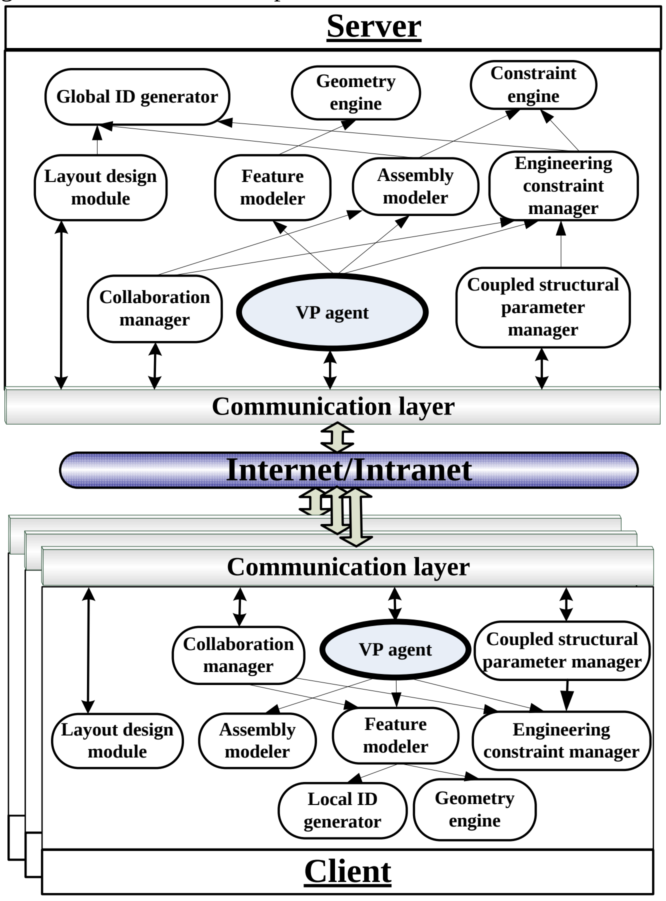

Figure 1. The system architecture for the collaborative top-down assembly design

3.1 The Architecture of Collaborative Top-down Assembly Design

To support the collaborative top-down assembly design, a replicated client-server based architecture is devised as shown in Figure 1. The client provides the ability for user interaction and is responsible for local information management. The server contains the whole product assembly model information and is responsible for the consistency of the distributed assembly model. A command based data exchange method between the server and the clients is adopted as the clients and the server have similar functional components.

Two layers can be distinguished for the system: the kernel layer and the communication layer. The communication layer is used by the kernel components to interact with each other through the network. The ground of the kernel layer is some modules that are similar to those of the traditional CAD system including the feature modeler, geometry engine and constraint engine. The feature modeler provides the ability for feature based modeling. It is constructed based on the geometry engine, ACIS. The constraint engine located at the server is responsible for constraints solving including geometric constraints and parametric constraints. Besides above components, the kernel layer has some extended modules summarized as following.

Variation propagation agent (VP agent): it is responsible for the design variation propagation which guarantees the consistency of the product assembly model over the clients and server.

Global ID generator: it is used to generate the global object ID to ensure the assembly model object IDs to be identical all over the clients and the server.

Assembly modeler: it is responsible for assembly modeling in a top-down method such as collaboratively assembly relationship definition, product structure design and assembly constraint definition.

Layout design module: it is responsible for the assembly layout design which is accomplished by the chairman.

Coupled structural parameter manager: It is responsible for the designers to collaboratively determine the coupled structural parameters.

Collaboration manager: it supports the designers to coordinate with each other. Usually it will invoke other functional component to accomplish the collaboration.

3.2 Distributed Assembly Model for Collaborative Top-Down Assembly Design

To effectively support the collaborative top-down assembly design, we put forward a distributed product assembly model, as shown in Figure 2. The product assembly model is a hierarchical object oriented model. Each level of the product assembly model corresponds to the representation for each of the three design stages of the design process.

The primary objects for the layout design include the AbstractPart, AbstractSubassembly and AbstractAssemblyRel. They are used to represent the abstract level of the part, subassembly and assembly relationship. AbstractPart and AbstractSubassembly contain such information as the name, the functional description and so on. AbstractAssemblyRel is a specification for the assembly relationship which includes the desired functional outcome such as the assembly scheme, assembly method and so on.

The objects for the skeleton design mainly include the SkeletonPart, SkeletonSubassembly, SkeletonAssemblyRel and EngineeringConstraint. SkeletonPart and SkeletonSubassembly refer to the FeatureList to represent their skeleton shapes as feature model. SkeletonAssemblyRel represents the assembly relationship information for the skeleton assembly model such as the key geometric and parametric constraints. The class EngineerConstraint that contains algebraic equations is useful for the high level parameter constraint. For example, the designers can calculate the structural parameters from function parameters through algebraic equations which involve the functional parameters (FunctionParameter) and structural parameters (StructrueParameter). The StructureParameter crosses two sections means that structural parameters are involved in both skeleton and detail assembly model.

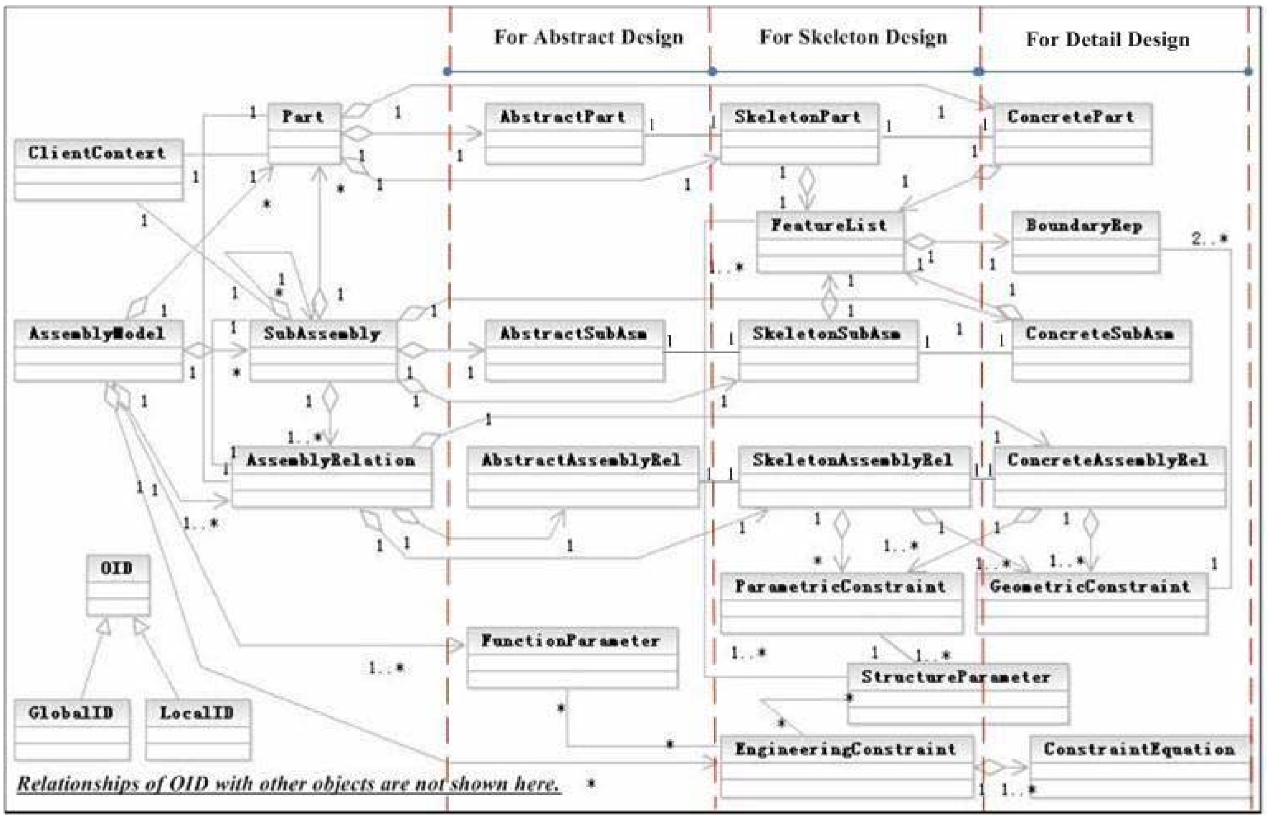

Figure 2. Class diagram of the collaborative assembly model

The basic objects for the detailed design are the ConcretePart, ConcreteSubassembly and ConcreteAssemblyRel. ConcretePart has the detail feature model which uses a boundary representation for its geometry model. ConcreteSubassembly is the detail representation of the subassembly which includes subassemblies and parts it contains and the internal assembly constraints. ConcreteAssemblyRel contains the detailed assembly constraints such as geometric and parametric constraints.

Different assembly model information is loaded by the server and clients according to the collaborative design requirements. The assembly model on the server is a complete assembly model including the whole assembly model information. To indicate how the assembly objects are distributed each subassembly or part points to a ClientContext object. It indicates the client's information associated with the designer such as the designer ID, status and address and so on. The LocalID and GlobalID are used to ensure the distributed object ID to be identical all over the distributed system. The LocalID is the ID that is generated and maintained by the client and the GlobalID is produced and managed by a central component located at the server.

A partial product assembly model is adopted to represent the assembly model distributed at the client which is dynamically loaded according to the designer's design information requirement. Since different subassemblies and parts of the assembly model are interrelated, during the design process a designer needs to be aware of others' design information that has correlation with his/her design task. Based on this, the partial product assembly model includes both the information of the subassemblies or parts undertaken by the designer and those having correlations with him/her. This is illustrated in Figure 3 in which the client assembly model for designer A contains not only all the information of subassembly A which is designer A's design task but also all the information of subassembly B because subassembly B and A are correlated through assembly relationship. It should be pointed out that every client also has the abstract assembly model of the whole product such as the product structure. So in Figure 3 the abstract assembly model of subassembly C is also included in the partial assembly model of designer A.

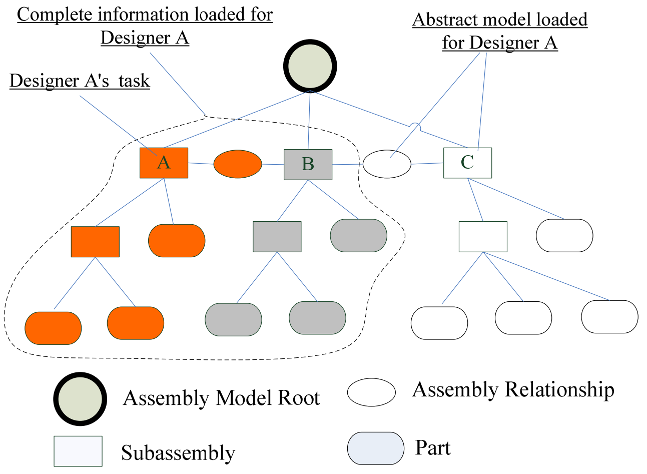
Complete information loaded for Designer A
Abstract model loaded for Designer A
Designer A's task
A
B
C
Assembly Model Root
Assembly Relationship
Subassembly
Part

Figure 3. Illustration of the partial assembly model at client

4 AGENT BASED VARIATION PROPAGATION FOR COLLABORATIVE TOP-DOWN ASSEMBLY DESIGN

4.1 Overview of the Approach

For collaborative top-down assembly design，variation propagation is the process during which the design variations initiated by the distributed designers are instantly monitored by the server which then updates the distributed assembly model on both the server and related clients accordingly. Its ultimate target is to ensure the distributed assembly model to be consistent for all the clients and the server. There are three key variation propagation situations for collaborative top-down assembly design, which mainly appear in the stages of collaborative skeleton design and detail design.

1. Hierarchical variation propagation. During the collaborative top-down assembly design process, the design information of early design stage will be inherited or shared by the later stage. If one object (e.g. a feature, a constraint, or a parameter) of the skeleton design stage is changed, its counterpart of the detail design stage should be updated accordingly.

2. The variation propagation of assembly relationships and engineering constraints. The assembly relationships and engineering constraints form a distributed constraint network. It is important to make the whole distributed constraint network satisfied when any design variation happens.

3. The propagation of feature variation. During the collaborative top-down assembly design, feature variations include feature addition, feature deletion and feature parameter modification. When the feature variation occurred at one client, it should be propagated to the server as well as the clients which are correlated with that client.

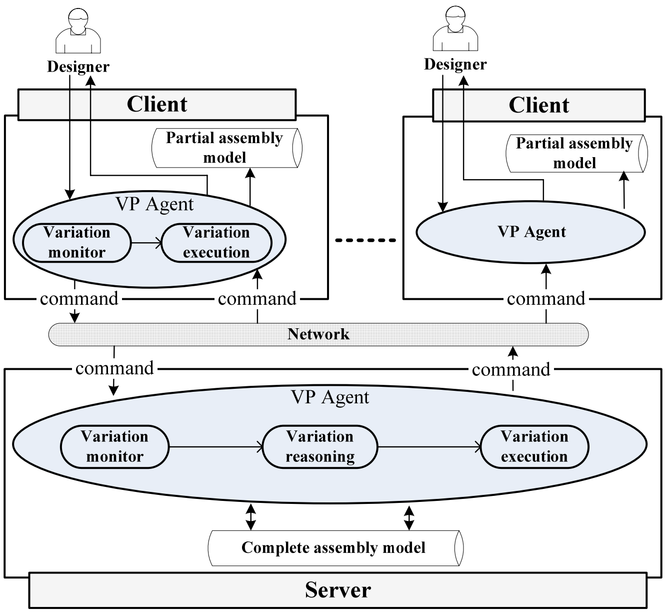

Figure 4. The agent based approach for variation propagation

To effectively achieve the variation propagation defined above, we put forward an agent based variation propagation approach which accomplishes the variation propagation through the interaction and cooperation of agents at the clients and the server, as shown in Figure 4. Here agent refers to a software entity that is autonomous, reactive, and intelligent.

In our approach, server VP (variation propagation) agent is the pivot for variation propagation. Its main task is to monitor the variation requests from the clients and infer all the assembly model objects that need to be changed accordingly for the server and related clients. To accomplish this goal, the server VP agent consists of the following functions.

1) Variation monitoring. The server VP agent concurrently manages the conversations with the distributed client VP agents, through which the server VP agent monitors any variation of the assembly model made by designers at related clients.

2) Variation reasoning. Based on the assembly model distribution and the constraints and hierarchical relations of the assembly model, variation reasoning processes the variation requests. Through variation reasoning, the VP agent at the server infers all the changed assembly model objects and the locations of the objects.

3) Variation execution. Based on the changed objects of the assembly model and their distributions, the server VP agent synthesizes an adequate task for the client VP agent to update the related assembly model on the client. A task is a set of actions described as commands that the client VP agent need to execute to update the distributed assembly model through the corresponding functional components as shown in Figure 1.

The main task of client VP agent is to monitor the variation requests from both the designer and the server, then realize the assembly model variation through invoking certain modeling operations.

In the whole approach, the variation reasoning plays a key role, which is described in detail below.

4.2 Variation Reasoning of Server VP Agent

The variation reasoning of the server VP (variation propagation) agent is to determine the objects of the assembly model that should be changed and the distributions of the objects, which is crucial to accomplish the variation propagation.

The key issues here is how to make the variation reasoning not only support traditional variation propagation but also effectively support hierarchical variation propagation between the skeleton design and detail design as well as the feature variation propagation required by the collaborative top-down assembly design. In this work, we achieve the variation reasoning for hierarchical variation propagation based on the hierarchical relationships between skeleton assembly model and detail assembly model involved in the distributed assembly model, the variation reasoning for feature variation propagation using the feature distribution and feature relationships of the distributed assembly model. Figure 5 shows the framework of the variation reasoning. As shown in Figure 5, the skeleton assembly model also invokes the three basic variation reasoning modules because they are necessary to deal with changes of the skeleton assembly model which, as shown in Figure 2, utilizes features to represent the skeleton shape and contains some engineering constraints and assembly constraints.

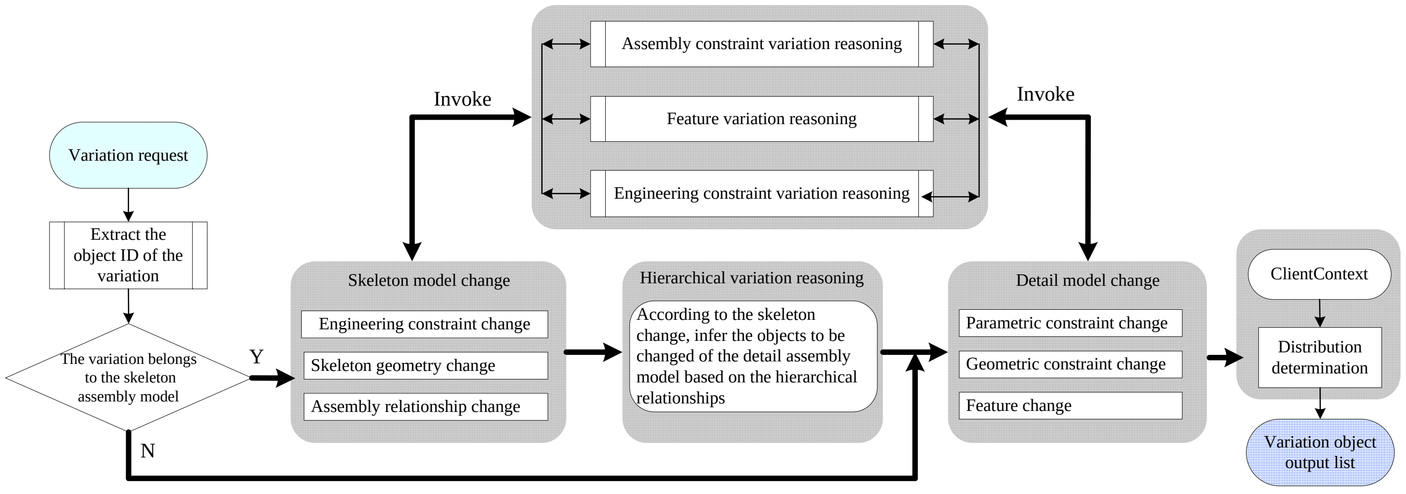
Assembly constraint variation reasoning
Feature variation reasoning
Engineering constraint variation reasoning
Invoke
Invoke
Variation request
Extract the object ID of the variation
The variation belongs to the skeleton assembly model
Y
N
Skeleton model change
Engineering constraint change
Skeleton geometry change
Assembly relationship change
Hierarchical variation reasoning
According to the skeleton change, infer the objects to be changed of the detail assembly model based on the hierarchical relationships
Detail model change
Parametric constraint change
Geometric constraint change
Feature change
ClientContext
Distribution determination
Variation object output list

Figure 5. Variation reasoning of the server VP agent

Before describing the specific variation reasoning algorithms, we first introduce some related definitions.

1． **ClientContext:** the object used to represent the assembly model distribution which contains the following information. 1) The designer information such as the ID of the designer, the physical address, the role and authority information and the status of the designer. 2) The private assembly model information which is undertaken by the designer. 3) The related assembly model undertaken by others which has assembly relationships with the private assembly model.

2. **Variation set (VaS):** the resulted objects of the variation reasoning, as shown in (1)：

VaS = {Vp，Vf，Vm} (1)

Where:

Vp: The changed parameters resulted from variation reasoning. The parameters contained in Vp may be the key parameters of the skeleton model or some feature parameters of detail assembly model. An element in Vp includes such information as the parameter ID and parameter value.

Vm: The assembly relationship changed resulted from the variation reasoning. The key information in Vm is the space transform matrix of the assembly model to position the geometric model.

Vf: The changed features resulted from the variation reasoning. An element in Vf includes such information as feature ID, feature variation type (add, delete or modify), feature parameter.

4.2.1 Hierarchical Variation Reasoning

The skeleton assembly model is used to define the key product information such as the product space claims, key parameters together with the engineering constraint and so on. When the distributed skeleton model is changed, the changes should be propagated to the detail assembly model since the key information in the assembly skeleton model is transferred to the detail assembly model. Based on the distributed and hierarchical assembly model, the hierarchical variation reasoning is realized in the following ways.

1. Parameter share based reasoning. The key parameters defined in the skeleton assembly are often reused at the detail assembly model so as to transfer the design intent from the skeleton design to the detail design. These parameters shared by different levels of assembly model are the base for the hierarchical variation reasoning. The system traces all the parameters that are shared by the skeleton and detail assembly model. When any of the parameters is changed, the skeleton and detail assembly model will both be changed accordingly.

2. Engineering constraint hierarchical based reasoning. If the engineering constraints of the skeleton model are changed, usually it will cause a hierarchical propagation which means the changed parameters in the skeleton model will influence the engineering constraints in the detail model. Based on the variation reasoning results of the engineering constraint of the skeleton model, the system infers the influenced engineering constraints of the detail assembly model.

3. Feature inheritance based reasoning. The shape of the skeleton can be reused at the detail design through feature copy. The system traces all the inheritance of the features to enable the feature change propagation of different levels of assembly model according to the inheritance relationship.

4. Assembly constraint reference based reasoning. The assembly constraints of the skeleton assembly model are some key constraints which are reused through constraint copy. If any high level assembly constraint of the skeleton assembly model is changed, its counterpart of detail assembly model is also updated according to the copy associations.

4.2.2 Reasoning of Engineering Constraint Variation

Engineering constraint expressed as algebraic equations forms a hierarchical and distributed engineering constraint network (DECN) which is built collaboratively to indicate the collaborative design intent, as shown in (2).

DECN = {X, M, R, L} (2)

Where:

X= $\{X_1, X_2 \ldots X_n\}$: A set of variables related to the product assembly model. The elements of X may be top level parameters shared by all the designers or feature parameters belonging to different designers.

M= {M_1, M_2…M_n}: A set of finite domains for the variables in X.

R= {R_1, R_2…R_n}: A set of constraints about the variables in X which are expressed as algebraic equations.

L= {L_1, L_2…L_n}: A set of locations the constraint network relates to.

The problem of variation reasoning of engineering constraint can be expressed as: when some of the variables in X or constraints in R are changed, determine the affected variables and constraints together with their distributions. The method to process the problem is shown in algorithm 1. The basic idea is to isolate the changed sub-network from the complex engineering constraints belonging to different locations. If the engineering constraints of the skeleton assembly model are changed, then the affected parameters and constraints of detail assembly model are found out according to the hierarchical associations. The related engineering constraints are divided into several sub-networks based on the method of [18] and solved through the constraint engine. Then based on the distribution of the DECN, the distribution information of the assembly constraints is figured out through ClientContext.

Algorithm 1: Variation reasoning of engineering constraint

```
Function Vpc= EC_Reasoning(Va, Ca, DECN, VaS)

Va:     input parameters changed
Ca:     input constraints changed
DECN:   input distributed engineering constraint network
Vpc:    output changed parameters
VaS:    variation set contains the reasoning output

Step 1. Extract the variables involved in Ca, define them as Pf .
        Set Pf = Pf + Va; Cf = Ca;
        Remove repeated elements in Pf.
Step 2. Get the constraints of the skeleton model related with any
        element of Pf, define them as Ct. Extract the variables
        involved in Ct, define them as Pvt.
        Set Pfp = Pf; Pf = Pf + Pvt; Cf = Cf + Ct;
        Remove repeated elements in Pf, Pfp, Cf.
        Repeat step 2 until Pfp is equal to Pf.
Step 3. Based on Pf and parameter sharing between the level of
        assembly model, search the parameters and constraints of
        the detail assembly model, which is similar to step 1 and 2.
        The results are added to Pf, Cf.
Step 4. Decompose Cf based on the method of [19].
Step 5. Decomposed Cf is calculated through constraint engine, get
        the changed parameters comparing with the old value,
        set them VaS.Vp.
Step 6. Based on the DECN, get the corresponding ClientContexts
        that the parameters in VaS.Vp relate to.
```

4.2.3 Reasoning of Feature Variation

Feature variations include feature addition, feature deletion and feature parameter modification which are the ways for designers to create or modify the distributed feature model during collaborative top-down assembly design. It is the task of the feature variation reasoning to determine the change of the distributed feature model to keep them consistent. For collaborative top-down assembly design, the feature level constraints among distributed parts are important to satisfy the product requirement and designers' design intent, which contains two aspects, the parameter relationship and assembly relationship. The former is the parameter constraints among the feature parameters of different parts or subassemblies. The later is the assembly constraints. If the feature model of one client is changed, the features at other locations that have correlations with it should be updated correspondingly. Maintaining these relationships between the distributed features is a key problem for the reasoning of feature variation.

The method to reason about the feature parameter variation is described in algorithm 2. In the algorithm, if the changed feature parameter involves in parametric constraints, the influenced parameters are figured out through algorithm 1, which results in other changed features. Then based on the association of the features between skeleton and detail assembly model, the detail feature model variation is determined according to the changed features of the skeleton assembly model. Lastly, the distribution information of the changed features is found out through the ClientContext.

The method to reason about feature addition is straightforward. The parameters used to create a feature such as the feature type, feature parameter are added to VaS.Vf to remember the change of the feature model. Feature deletion involves update of the distributed constraints between features, which means the constraints related with the deleted feature are removed. Then the feature to be deleted is added VaS.Vf.

The feature variation causes the geometric model to be reconstructed which often affects the assembly constraints. The reasoning of the assembly constraints is described next.

Algorithm 2: Reasoning of feature parameter variation

```
Procedure FT_Reasoning(Fpara, VaS)

Fpara:  input changed feature parameter
VaS:    variation set contains the reasoning output

Step 1. If the changed feature parameter is involved in any
        parametric constraint, go to step 2;
        Else, set ParaR = Fpara; go to step 3.
Step 2. Use algorithm 1, get the changed parameters:
        ParaR = EC_Reasoning(Fpara, NULL, DECN, VaS)
Step 3. For each PF∈ParaR, if PF is a feature parameter,
        Set Pc = Pc + PF. Pc is defined as feature parameter set.
Step 4. For each PF∈Pc, if the feature of PF belongs to skeleton
        assembly model and inherited by the detail model, get the
        detail feature parameter, define it as Pt, set Pc = Pc + Pt.
Step 5. For each PF∈Pc, get the related ClientContext of PF
        Set PF and its feature to Vas.Vf;
```

4.2.4 Reasoning of Assembly Constraint Variation

The product assembly constraints form a hierarchical constraint network in the sense that every subassembly has its own assembly constraints that only function on its own components, as shown in Figure 6. Besides, the subassemblies and parts involved in assembly constraints are distributed and undertaken by different designers. The distributed assembly constraint network (DACN) is defined as (3):

DACN = {E, A, L} (3)

Where:

E = {E_1, E_2…E_n}: A set of subassemblies or parts of the product distributed at different locations.

A= {A_1, A_2…A_n}: The assembly constraints of the product acting on E.

L= {L_1, L_2…L_n}: A set of ClientContexts indicate the distributions of the entities in E.

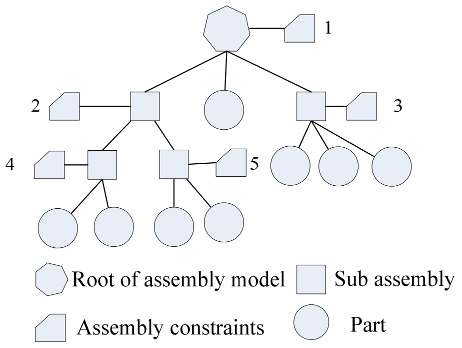

Figure 6. Illustration of the hierarchical assembly constraints

```
Algorithm 3: Reasoning of assembly constraint variation
Procedure   AC_Reasoning(Ac, DACN, VaS)
Ac:      input changed assembly constraints
DACN:    input distributed assembly constraint network
VaS:     variation set contains the reasoning output

Step 1: Locate the node of Ac in DACN, define it as Ap and put it
        to a queue defined as Qap.
Step 2: If Ap is not root of DACN, get the parent node of Ap,
        define it as App. Set Ap = App; put App to Qap.
        Repeat step 2 until App is the root of DACN.
Step 3: If any element of Qap is associated with the assembly
        constraint of the detail assembly model, get the changed
        constraints of the detail assembly model using the
        method similar to that of step 1 and 2.
Step 4: Solve the constraints of Qap in sequence, put the resulted
        transforms to VaS.Vm.
Step 5: Get the related ClientContext of the related entities of the
        assembly constraints in Qap.
```

The variation of assembly constraint is the addition or deletion or modification of the assembly constraint. Based on DACN the variation reasoning of assembly constraint is shown in algorithm 3 which is summarized as follows: First a queue of assembly constraints is searched based on the changed assembly constraints and DACN. For example, if the assembly constraints 4 in Figure 6 are changed, then the searched queue contains 4, 2 and 1. Then based on the hierarchical relationship of the assembly constraints, the detail assembly constraints changed are found out. Lastly, constraint solver figures out the transform matrix of every subassembly and part that are affected by the changed assembly constraints and the distributions of the entities are figured out through ClientContext.

4.3 Transmission Mechanism of Assembly Model

Based on the replicated client-server based architecture as described in section 3.1, a command based communication between the client and sever is adopted to accomplish the variation propagation. It directly transmits operation commands between the client and server and greatly reduces the data transferred through the network. The command based communication method makes it possible for the designers to collaborate with each other at real time because the design change can be transferred with few data to the related clients very quickly which makes the clients be aware of the design change immediately.

A command is a 6-tuple entity:

OP = <S，O，P，I，F，T> (4)

Where:

S: The designer ID who initiates the design variation.

O: A predefined operation type that acts on the object of the assembly model.

P: The parameters of the command.

I: A serial number which denotes the time and sequence of the command.

F: The source agent ID of this command.

T: The destination agent ID of the command.

Take the typical situation of the feature variation during the collaborative design process as an example. Because the clients and the server have uniform feature modelers, the command together with its parameters can be processed by the server and clients to avoid transmitting the complex boundary representation. For example, the command <S1, VFP, (13, 7, 2, 45.0), 0x73, 0x04, 0x01> is to change a feature parameter value. In this command, the command parameter (13, 7, 2, 45.0) is used to uniquely indicate the feature parameter and its value, the meaning of the numbers is explained as follows: part ID is 13, feature ID is 7, parameter ID is 2 and the new parameter value is 45.0.

Because the clients have no constraint engine, they can not deal with the constraint-specific commands. Therefore, the solved results of the constraints are transmitted between the clients and the server. For example, the space transforms of the part or subassembly are transferred between the clients and server for assembly constraint variations.

As shown in the example of feature modification, for the command based data transmission between clients and server to be effective, it is crucial that the object ID to be identical all over the clients and the server because it is used to refer to an object by the server and the clients to exchange data. There are two kinds of ID in the system, the global ID and local ID. For the object such as the ID of a part or subassembly, a global ID is used which is produced and managed by a central component located at the server. For the object such as the ID of a feature or a geometric element, a local ID is used which is generated and maintained by the client. For example, when the feature model is to be updated, the system can find out the feature through the feature’s local ID and parent ID (a global ID of the part or subassembly the feature belongs to).

5 THE SYSTEM IMPLEMENTATION AND A DESIGN VARIATION PROPAGATION EXAMPLE

An agent is a software entity that is autonomous, initiative, automatic and target-oriented. The agent based system supporting the variation propagation for the collaborative top-down assembly design is implemented based on the variation propagation method discussed above. It is a subsystem of our collaborative top-down assembly design system [15]. As shown in Figure 1, the system consists of variation propagation (VP) agents residing at the server and the clients. Both the server and client agents are coded in Visual C++ 6.0 on a network PCs with Windows 2003/XP operating systems. For the promising features of IEEE FIPA[16] and XML[17], an ACL (agent communication language) [16] represented in XML based on FIPA is adopted for the agents' communication.

For the server VP agent to concurrently interact with the agents of the clients, a multi-thread environment is adopted where a work thread dedicating to the specified interaction is created for each client agent. The ground of each work thread is a message pump built on the Transmission Control Protocol (TCP), which provides the reliable data transmission. All messages from the work thread are processed by the command parser, which is a key C++ class for the agent to interpret the variations request.

To update the assembly model，the VP agent depends on some system kernel components as shown in Figure 1 such as the feature modeler, constraint engine and so on. The difference for the server and client is that there are no constraint engines on the clients. The feature modeler is constructed based on solid modeling engine that utilizes the ACIS geometry kernel. To make the system flexible and extensible, a common middleware is constructed which separates the system with the actual constraint engine. The middleware enables the new constraint engine to be integrated into the system easily. Currently, as one of the most popular math software, MATLAB is being encapsulated into the system as the constraint engine through the COM API. The geometric constraint engine in the system is implemented utilizing the knowledge-based method [19].

Next, based on the developed prototype system, the process of a number of designers collaboratively accomplish a manipulator assembly design in a top-down way is given as an example to illustrate the variation propagation. The chairman initially creates the layout assembly model as shown in Figure 7 and assigns the subassemblies to different designers. As shown in the figure, there are six key subassemblies: Hand, Hand driver, Hand deliver, Swing motor, Elevator, Walk device (Stepper motor and the Slide pole). The designers Designer_2 and Designer_3 are responsible for Hand driver and the Hand deliver respectively. We demonstrate the variation propagation through two collaborative activities of Designer_2 and Designer_3: the collaborative design of assembly relationship and the variation of structural parameter, as shown in Figure 8 and Figure 9. For convenience the following abbreviations are used: D_2(Designer_2), D_3(Designer_3), HDE(Hand deliver), HDR(Hand driver).

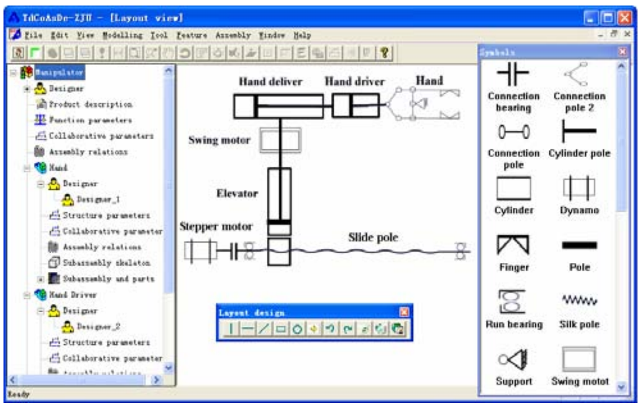

Figure 7. Layout design of the manipulator

Figure 8 is the variation propagation processes during which D_2 and D_3 collaboratively design the assembly relationship from skeleton design to detail design.

Figure 8a and 8b are the snapshots of the skeleton design of D_2 and D_3. The system provides the designer with two views, one is the personal work view that is for the designer to design his own subassembly or parts, and the other is the collaborative assembly design view which contains all the subassemblies that have assembly relationship with the designer's design task. As Figure 8 shows, the collaborative assembly design view of D_2 and D_3 each has the other's subassembly model because they are correlated through assembly relationships. The Figure 8g and 8h show the detail assembly model shared by D_3 and D_2 after they collaboratively accomplish the assembly relationship. This final assembly model is generated through the following three main steps.

1) According to the assembly relationship specification that is collaboratively determined at the skeleton design stage, designers D_2 and D_3 modify their own assembly model concurrently. In Figure 8c and 8f, D_3 and D_2 concurrently construct the features needed to define the assembly constraints. Figure 8d and 8e are the propagation results after D_3 and D_2 create their feature model. It can be seen from the figures that the D_2 and D_3 both have the feature model of the other's so that they can collaboratively define the assembly constraints.

2) When the feature model of D_2 and D_3 are ready, the designers start to collaboratively establish the geometric and parametric constraints to realize the assembly relationship. Figure 8e shows the parametric constraint built by D_3, named as "D1=D2", which means the two cylinders have equal diameters. Figure 8f depicts the geometric constraints constructed by D_2, in which the faces of the HDE and HDR are mated and the centerlines are collinear.

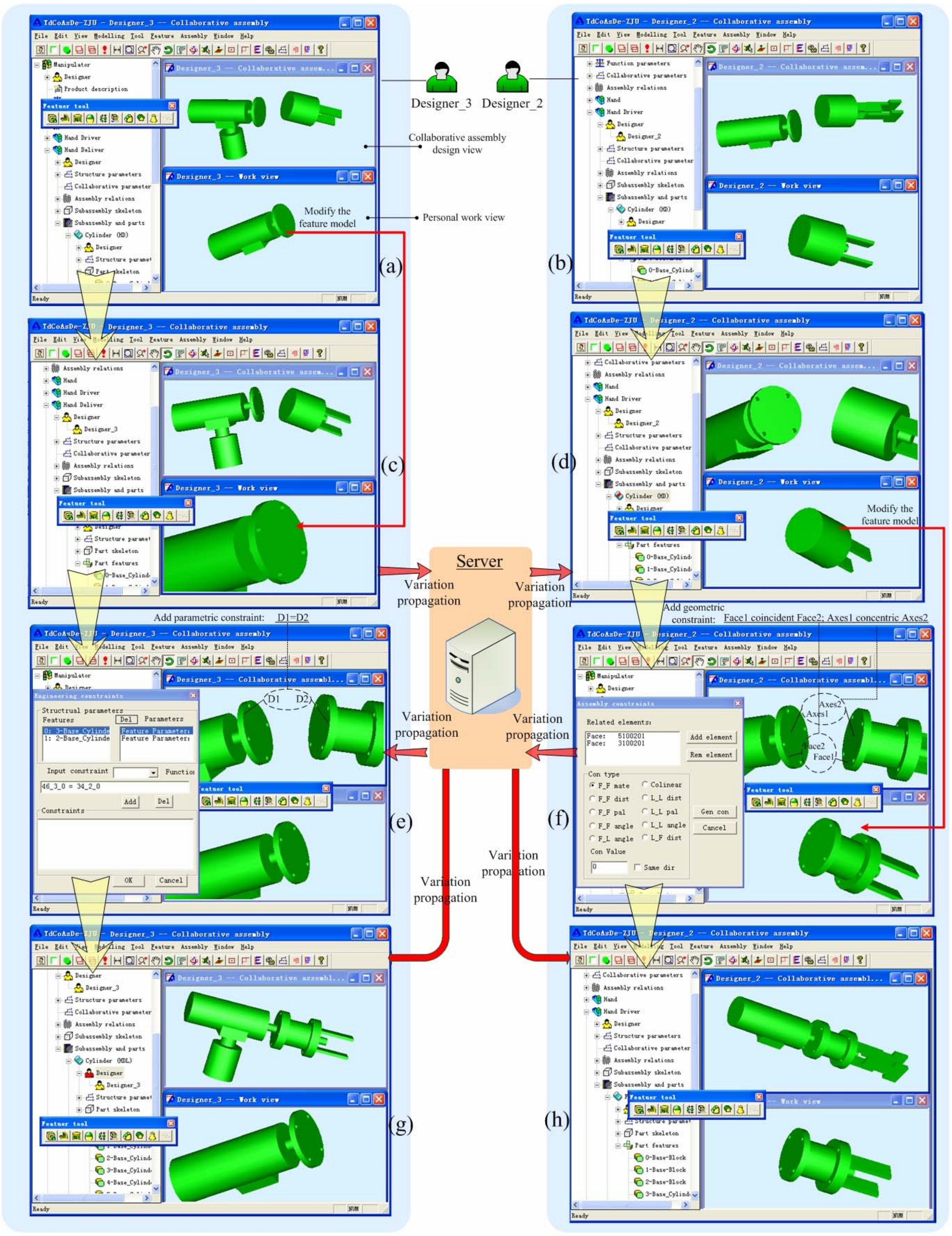

Figure 8. Variation propagation of collaborative assembly relationship design

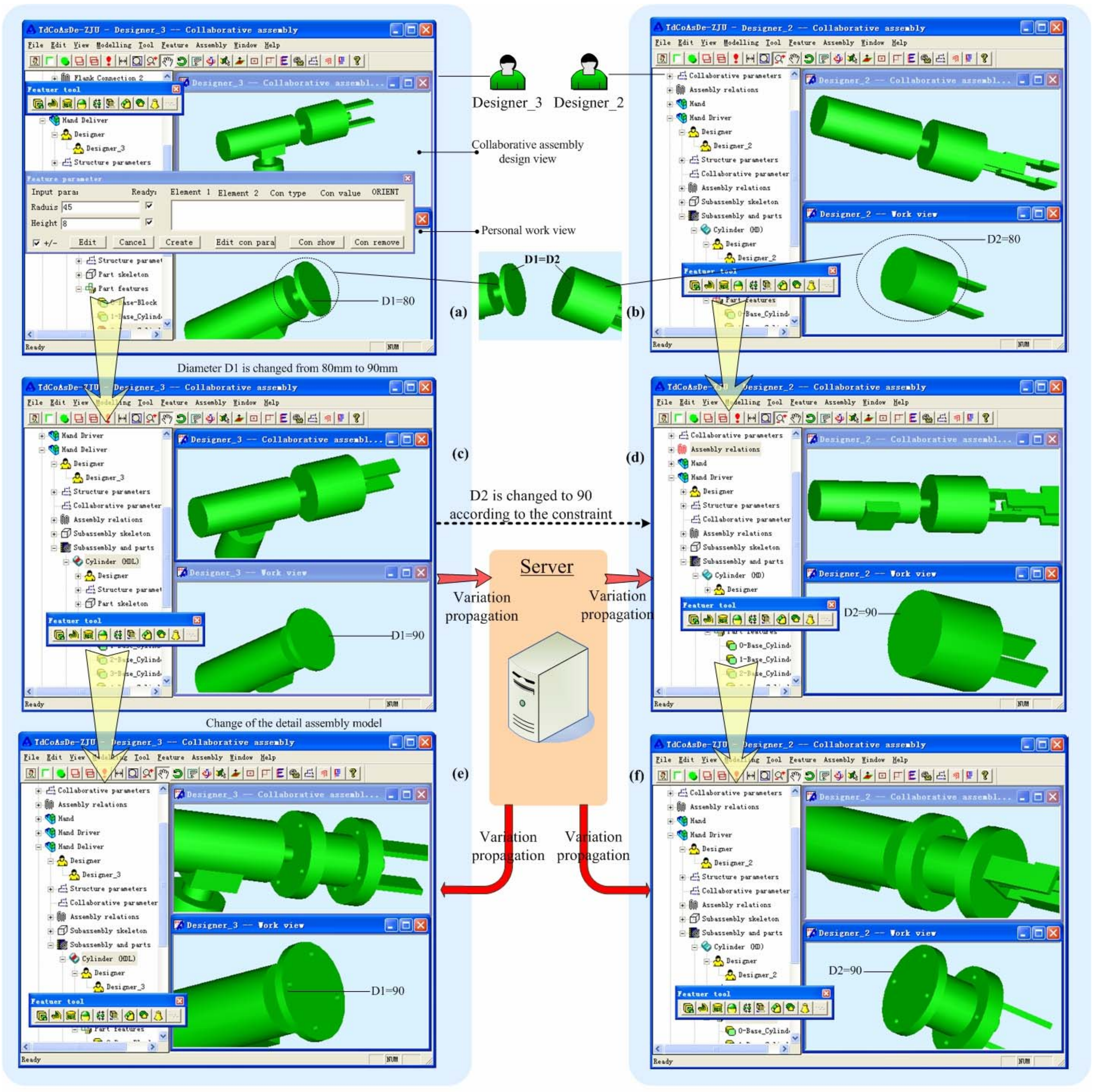

Figure 9. The variation propagation of the structural parameter

3) The assembly constraints are propagated to the server which infers all the affected subassemblies and their distributions, i.e., the space positions of HDR and HDE, the diameters of HDR and HDE. Then HDE and HDR are updated for clients of D_2 and D_3. The final assembly model of them are shown in Figure 8g and 8h.

Figure 9 is the variation propagation of the parameter which updates the distributed skeleton assembly model and detail assembly model. In Figure 9, designer D_3 modifies the parameter of HDR from 80mm to 90mm, as shown in Figure 9a. Because the parametric constraint of the diameters of HDR and HDE defined early as shown in Figure 8, the parameter of the D_2 will change to 90mm after HDR is changed, as show in Figure 9d. Because the changed parameter is shared by the skeleton and detail assembly model, the variation of the feature parameter in the skeleton drives that of the detail model to change too. So the diameters of detail assembly model of HDR and HDE are changed to 90mm, as shown in Figure 9e and 9f.

CONCLUSIONS AND FUTURE WORK

In the paper an approach to the variation propagation for collaborative top-down assembly design is presented. The contributions of this paper are summarized as follows:

1. An agent based approach is adopted to support the variation propagation for collaborative top-down assembly design. Through the interaction and cooperation of the agents located at the clients and the server, automated and intelligent variation propagation is achieved.

2 According to the requirements of the variation propagation for collaborative top-down assembly design, four kinds of variation reasoning including hierarchical variation reasoning, engineering constraint variation reasoning, feature variation reasoning, and assembly constraint variation reasoning are identified and the corresponding algorithms are developed. The variation reasoning algorithms can not only effectively support traditional variation propagation but also support hierarchical variation propagation between the skeleton design and detail design as well as the feature variation propagation.

3 A distributed assembly model that can effectively support the design variation propagation for the collaborative top-down assembly design is given.

To fully support the variation propagation, the future work will focus on the following areas:

1 Automatic validation of variation propagation results.

2 Information security during variation propagation.

ACKNOWLEDGMENTS

The authors are very grateful to the financial support from NSF of China (No. 60574061, No.60736019) and 863 High Tech Plan of China (2007AA04Z149).

REFERENCES

[1] Mantyla M, A modeling system for top-down by design of assembled products, IBM Journal of Research and Development, 1990, 34(5): 636—659

[2] Chung JCH, Hwang TS, Wu CT, et al., Framework for integrated mechanical design automation, Computer-Aided Design, 2000, 32(5): 355-365

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

[4] Noort A, Hoek GFM, Bronsvoort WF, Ingegrating part and assembly modeling, Computer-Aided Design 2002, 34(13): 899-912

[5] Bidarra R, Kranendonk N, Noort A, W. F. Bronsvoort, A collaborative framework for integrated part and assembly modeling, Proceedings of Solid Modeling, Seventh ACM Symposium on Solid Modeling and Applications, Saarbrucken Germany, June 17-21, 2002

[6] Li WD, Ong SK, Fuh JYH, et al., Feature-based design in a distributed and collaborative environment, Computer-Aided Design 2004, 36(9): 775–797

[7] Shyamsundar N, Gadh R, Internet-based collaborative product design with assembly features and virtual design spaces, Computer-Aided Design, 2001, 33(9): 637-651

[8] Chen L, Song Z, Feng L, Internet-enabled real-time collaborative assembly modeling via an e-Assembly system: status and promise, Computer-Aided Design, 2004, 36(9):835-847

[9] Lyons K, Shooter S, The open assembly design environment: an architecture for design agent interoperability, Proceedings of the ASME Design Engineering Technical Conferences, Las Vegas, Nevada, USA, September 12-15, 1999

[10] Dornfeld D, Wright PK, Roundy S, A. Rangarajan, S. H. Ahn, Agent interaction in CAD/CAM, Transactions of NAMRI/SME, Volume XXIX, 2001

[11] Zha XF, A knowledge intensive multi-agent framework for cooperative/collaborative design modeling and decision support of assemblies, Knowledge-Based Systems, 2002, 15(8): 493–506

[12] Mori T, Cutkosky MR, Agent-based collaborative design of parts in assembly, Proceedings of ASME Design Engineering Technical Conferences, Atlanta, Georgia, USA, September 13-16, 1998

[13] Shi J, Huang GQ, Mak KL, CYBERAGENT: collaborative agents for distributed applications over the internet, Proceedings of ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Pittsburgh, Pennsylvania, USA, September 9-12, 2001

[14] Wang JX, Tang MX, An agent-based approach to collaborative product design, Proceedings of ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Philadelphia, Pennsylvania, USA, September 10-13, 2006

[15] Zhang ST, Chen X, Gao SM, Yang YD, A framework for collaborative top-down assembly design, Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Las Vegas, Nevada, USA, September 4-7, 2007

[16] http://www.fipa.org/repository/aclspecs.html

[17] http://www.w3.org/XML/

[18] Sridhar N., Agrawal R., Kinzel G.L., Active occurrence-matrix-based approach to design decomposition, Computer-Aided Design, 1993, 25(8): 500-512

[19] Aldefeld B., Variation of geometric based on a geometric-reasoning method. Computer-Aided Design, 1988, 20(3):117-126.