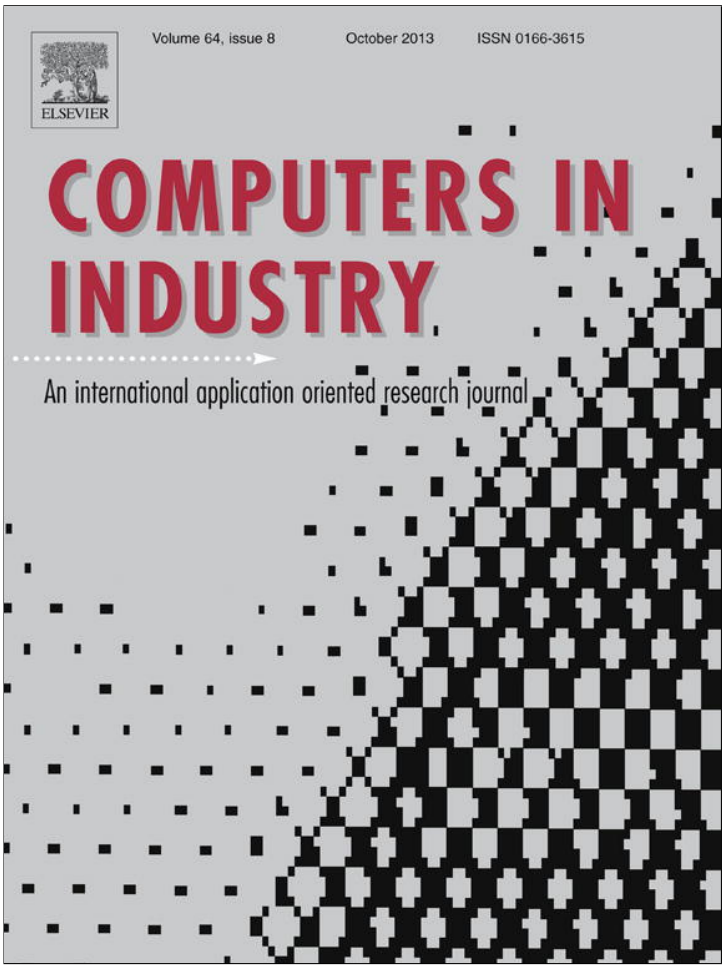


Computers in Industry 64 (2013) 967–983

Contents lists available at SciVerse ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

A framework for collaborative top-down assembly design

Shuming Gao [a,*], Shuting Zhang [b], Xiang Chen [a], Youdong Yang [c]

[a] State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou 310027, PR China
[b] School of Mechanical Engineering & Automation, Zhejiang Sci-Tech University, Hangzhou 310018, PR China
[c] Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, PR China

ARTICLE INFO

Article history:
Received 30 October 2012
Received in revised form 19 March 2013
Accepted 31 May 2013
Available online 18 July 2013

Keywords:
Collaborative design
Top-down design
Assembly design
Computer-aided design

ABSTRACT

This paper presents a new system framework for collaborative top-down assembly design. Different from current computer-aided design (CAD) systems, the framework allows a group of designers to collaboratively conduct product design in a top-down manner. In our framework, a multi-level and distributed assembly model is adopted to effectively support collaborative top-down assembly design. Meanwhile, fine-granularity collaborative design functionalities are provided. First, the coupled structural parameters involved in the distributed skeleton models of the product can be collaboratively determined by the correlative designers based on fuzzy and utility theory. Second, agent based design variation propagation is achieved to ensure the consistency of the multi-level and distributed assembly model during the whole design process. Third, collaborative design of assembly interfaces between the components assigned to different designers is supported. The prototype implementation shows that our framework works well for supporting practical collaborative top-down assembly design.

1. Introduction

It is well recognized that the design of a complex mechanical product has two characteristics. One is that the design is generally carried out in a top-down manner, consisting of several different phases. Top-down design manner takes product design as a problem solving method which involves step-by-step analysis and synthesis. A top-down product process proceeds from the qualitative to the quantitative, each new step being more concrete than the last [1]. Normally this design process can be divided into four phases, i.e., functional design, conceptual design, embodiment design, and detail design. Another characteristic of the complex product design is that it is usually conducted by a group of designers in a collaborative way. That is, a group of designers conduct product design in parallel with each of them being in charge of a component and interacting with others in real time to resolve design conflicts in the early stages. In this way, the design time of a complex product can be greatly shortened [2].

The current computer aided design (CAD) systems are able to model parts and assemblies very well but not to design [3]. They provide only limited support for top-down design and collaborative design although they are good at supporting geometric modeling at the detailed design stage. Before using the CAD systems to create the part or assembly digital model, the key problems of product design are already worked out, e.g., the solution scheme is determined, the structural parameters of the parts are figured out, the layout is planned, the assembly relationships and assembly constraints are determined. Similarly, the research on supporting users to collaboratively accomplish the product design also focuses on detail digital modeling. In order to enable CAD systems to support real design of complex products, the researches on both top-down design and collaborative design have been conducted since early 1990s. Historically, top-down design and collaborative design were studied independently. The research on top-down design mainly focused on achieving the automated and intelligent mapping from the product model for the early design phase to that for the late design phase. Nevertheless, limited by the computer's capability of representing and handling design knowledge and experience, the progress made in this aspect is not much and still far from the practical requirements. On the other hand, as the rapid development of network and computer supported cooperative work (CSCW) technologies in the past two decades, research on collaborative design has been paid increasing attentions. The emphasis of the research is at enabling the collaboration in design process by renovating conventional CAD systems to become distributed and collaborative using network, distributed computing, CSCW, and Web technologies. The developed collaborative CAD systems allow a group of designers to review design models and conduct part or assembly modeling collaboratively. However, since the current collaborative CAD systems are based on the traditional CAD technologies, they are limited to supporting the collaborative design activities in detail design phase. In view that the most important collaboration

* Corresponding author. Tel.: +86 571 87951045; fax: +86 571 87951780.
E-mail addresses: smgao@cad.zju.edu.cn (S. Gao), zhangshuting@zstu.edu.cn (S. Zhang), xchen@cad.zju.edu.cn (X. Chen), yyoudong@cad.zju.edu.cn (Y. Yang).

0166-3615/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.compind.2013.05.007

activities in practical product design often occur in early design phase rather than in detail design phase, the existing collaborative design functions mentioned above are still far from the practical requirements.

To effectively support both top-down design and collaborative design, a new system framework is proposed in this paper for developing a new generation of CAD systems, which allows a group of designers geographically dispersed to collaboratively conduct top-down assembly design of complex products with their design knowledge and experience in a computer supported cooperative work environment. In the new system framework, a new distributed and multi-level assembly model is adopted as a meta-level representation. Meanwhile various collaborative design activities, including collaborative determination of coupled parameters, collaborative defining of assembly interfaces, design variation propagation, are supported.

The rest of the paper is organized as follows. In the second section we review the related work. In the third section we give a brief overview of the proposed framework for the collaborative top-down assembly design. A multi-level and distributed assembly model is presented in the fourth section. In the following three sections the main functions of the framework are introduced including collaborative determination of coupled structural parameters, agent based variation propagation and collaborative design of the assembly interfaces. In the eighth section we present the prototype system developed based on the proposed framework and a practical design example based on the system is presented in the ninth section. Finally, the conclusions and future work are given.

2. Related work

The related work of this paper can be divided into two categories: top-down product design and collaborative CAD. In general, these two categories of work were conducted independently.

2.1. Top-down product design

Although the traditional CAD systems are geometric modeling centered and are mainly suitable for supporting detail design, driven by the requirement to support complex product design, CAD technologies that can support top-down product design has been investigated since early 1990s. One focus of the research is on how to support early design stage and enable information inheritance and consistency of different design stages. As one of pioneers, Mäntylä discussed the main requirements for CAD systems to support top-down product design and described a prototype modeling system for top-down design of assembled products in [4]. Although this work is preliminary and lacked of details, it points out an important direction for CAD research. To support early design stage, much effort is put on developing methods for automated conceptual design. These methods include cased-based reasoning [5], qualitative reasoning [6], agent-based approach [7], genetic algorithm [8], neural network method [9], and so on. While these formal methods depend on computers to generate the concept design, other researches fall into constructive approaches which focus on assisting human to express the design intent at early design stage. For example, computer supported sketching method is adopted to support concept design [10,11]. This method enables designers to input sketches as they do on a notebook, and then the computer can reconstruct the 3D object from the sketches. As a predominant way of geometry modeling [12], feature technology was extended to support early design stage and enable information inheritance and consistency of different design stages by Csabai et al. [13]. They used design spaces and functional features in their 3D Layout Module to support top-down layout design, which can construct a bridge between the abstract nature of the conceptual design phase and the geometric nature of the detail design phase. Similarly, Aleixos et al. [14] presented a fundamental structure that behaves as a three-dimensional scheme containing the whole project criteria and the basic elements, which helps to realize the integration of conceptual design and final geometric modeling. There are also some semi-automated methods to promote the product design such as design rationale capturing method [15], ontology mapping method [16], knowledge search and reuse method [17], constraint based method [18].

Another kind of research work is on extending the traditional CAD model to make it support top-down product design. Besides Mäntylä's work afore mentioned, Brunetti et al. [19] presented an approach to enabling a feature-based representation to capture the product semantics generated in conceptual design so as to support the evolution of product semantics along the product development process. To make the approach effective, the feature definition should be complete and an effective feature mapping method between different feature levels should be provided, but they are difficult to achieve. In recent years, the design and process group of NIST has been investigating interoperability framework for PLM and has proposed Core Product Model (CPM) and Open Assembly Model (OAM) [20]. In view that CPM incorporates the function and behavior of products and OAM is based on CPM, both of them have potentials to be used to support top-down design. Nevertheless, since both CPM and OAM do not provide a relatively independent representation for each design stage, they need to be extended according to the requirements of top-down design.

In addition, some commercial CAD systems such as Pro/E [21] have started to provide certain functions to support top-down design scenario for a single user. Pro/E has introduced skeleton model to capture central design information for an assembly, such as key dimensions of each part and geometric constraints between parts, which enables the user to conduct design at skeleton level first. However, since the skeleton model of an assembly in Pro/E does not have explicit hierarchy and contains all the parts and geometric constraints involved in the final assembly model, it is not clear if the real top-down design of complex products can be effectively supported based on such skeleton model.

2.2. Collaborative CAD

As classified by Li et al. [22], the work on collaborative CAD can be divided into two types: (1) visualization-based collaborative CAD; and (2) co-design oriented collaborative CAD. The former focuses on making CAD systems have the functions supporting visualization, annotation and inspections of CAD models under design, whereas the latter aims at providing users the functions of modeling and modifying CAD models interactively and collaboratively online, including collaborative assembly modeling. In the following, we just review the work on collaborative assembly modeling in more detail. For a comprehensive review on collaborative CAD, please refer to [22].

Mori and Cutkosky [23] investigated agent-based collaborative assembly design. In their framework, each design agent is responsible for the design of a part or a component in assembly and coordinates with others based on the theory of Pareto optimality. Since the communication between design agents is simple and limited, the framework seems hard to support the complex collaborative activities.

Shyamsunder and Gadh [24] presented an approach to internet-based collaborative assembly design and developed a prototyping system cPAD. In the approach, a compact assembly representation called AREP is proposed, which adaptively simplifies the model of

the product thereby preserving the details of a particular component required during product assembly design. They adopted a three-tier client-server architecture to support internet-based collaborative assembly design between original equipment manufacturer (OEM), in which the virtual design space and interface assembly feature are used to ensure the related sub-assemblies or components to be compatible. AREP has the potential to support collaborative top-down assembly design. However, how to reuse the AREP to collaboratively accomplish the product detail design is not addressed.

In order to achieve internet-enabled real time collaborative modeling, Chen et al. [25] developed a web-based collaborative assembly modeling system called e-Assembly. In e-Assembly, a distributed assembly model consisting of a master assembly model (MAM) and a slave assembly model (SAM) is adopted. The MAM is a complete representation stored in the server while the SAM is just the mesh representation of the assembly unit associated with a client and is stored in the client for supporting visualization-based manipulation in the client. The aim of e-Assembly is also at supporting the collaborative assembly modeling between OEM and suppliers. Although e-Assembly allows multi-user to collaboratively set up an assembly model by defining the assembly constraints between the components of the assembly through Web, it does not address real-time design modification and top-down assembly design.

Bidarra et al. [26] presented a collaborative framework for integrated part and assembly modeling. Based on the multi-view feature modeling technique, the association and interaction between part modeling and assembly modeling during collaborative design are addressed. With the framework, although the collaborative design tasks can be assigned in a top-down manner, the assembly modeling itself is still conducted in a bottom-up way.

According to the requirement of information protection in collaborative modeling, Cera et al. [27] proposed role-based viewing envelopes of an assembly. They combined the multi-resolution geometry of a part with role-based access control to generate the role-based views of the part, limiting the user to the design detail he/she is permitted to access so as to protect the information proprietary during collaborative design.

Lu et al. [28] investigated design modification in a collaborative assembly design environment and proposed a control mechanism for design modification propagation. Their work focused on the design modification propagation after the assembly model had been set up, without addressing the design variation propagation during top-down assembly design where there exist more complex design variation propagations.

Mun et al. [29] presented a neutral skeleton model to exchange design information between collaborative OEM and suppliers, through which the intellectual property of companies could be protected and engineering changes during development could be propagated. Their work mainly put effort on the definition and translation of geometric elements involved in the skeleton model, rather than a thorough collaborative top-down assembly design.

Chen et al. [30] presented a top-down assembly design process which is refined from the traditional product design process to better exhibit the recursive execution and structure-evolvement characteristics of product design. Moreover, various inheritance mechanisms for transferring and converting information pertaining to different design phases are classified and described including function inheritance, geometry inheritance, feature inheritance and interface inheritance. However, how to support designers geographically distributed to collaboratively conduct assembly design in a top-down manner is not considered.

Different from the previous work mentioned above, this paper is intended to propose a system framework for product design that crosses two concepts: top-down design and collaborative design. In a word, the proposed system framework should be able to effectively support the collaborative top-down assembly design of complex products among a group of designers inside a single company.

3. Overview of the framework

Due to the great complexity of the collaborative top-down product design, this work focuses on the product design process which comes after the conceptual design. Since most of collaborative activities during product design occur in this design process, we call this design process as collaborative top-down assembly design which takes the product structure created during conceptual design as input and the complete assembly model of the product as output. While the design process is generally a part of top-down product design process, it is refined according to designers' collaborations which includes not only the basic design activities such as collaborative assembly modeling and part design, but also activities such as the collaborative parameter determination and collaborative assembly interface definition. Generally, collaborative top-down assembly design can be divided into the following three phases: the layout design, the skeleton design and the detail design. Compared with traditional design method, collaborative top-down assembly design raises more requirements for a CAD tool as follows:

(1) Allow designers to do assembly design in a top-down manner from layout design, skeleton design to detail design and support the design information inheritance and evolution from the early design phase to the detail design phase.

(2) Permit designers geographically distributed to collaboratively conduct assembly design. In other word, the CAD tool should be distributed, and with such a distributed CAD tool, every designer should be able to collaborate with those designers in charge of the components having assembly interfaces with the component being designed by the designer. Specifically, the CAD tool should be able to support the relevant designers to collaboratively determine the coupled structural parameters and assembly interfaces between the components they are responsible for.

(3) The assembly model adopted by the CAD tool should encompass the design information for every design phase so that all the three design phases of the collaborative top-down assembly design can be effectively supported. Moreover, the assembly model should be distributed and the consistency of the distributed assembly model should be automatically guaranteed through a design variation propagation mechanism during the whole design process.

Based on the above requirement analysis, in order to enable CAD systems to effectively support the collaborative top-down assembly design, we propose a new CAD system framework, as shown in Fig. 1. The framework adopts a replicated client-server architecture where the server is used to store and manage the complete assembly model of the product as well as the information about designers and is responsible for achieving the communication and design variation propagation between all the clients, while the client mainly provides the functions for a designer to do component design in their familiar way and to communicate with the server in real-time. In the architecture, the client itself is a design system with the facility of communicating with the server, so the architecture is a replicated one. In order to keep the partial assembly models located in different clients as well as the global assembly model in the server synchronized and consistent, we let all the clients and the server have the same design and modeling functions and adopt command based design information transfer strategy instead of direct transmission of the model between client and server. Since the data amount of transferring commands is

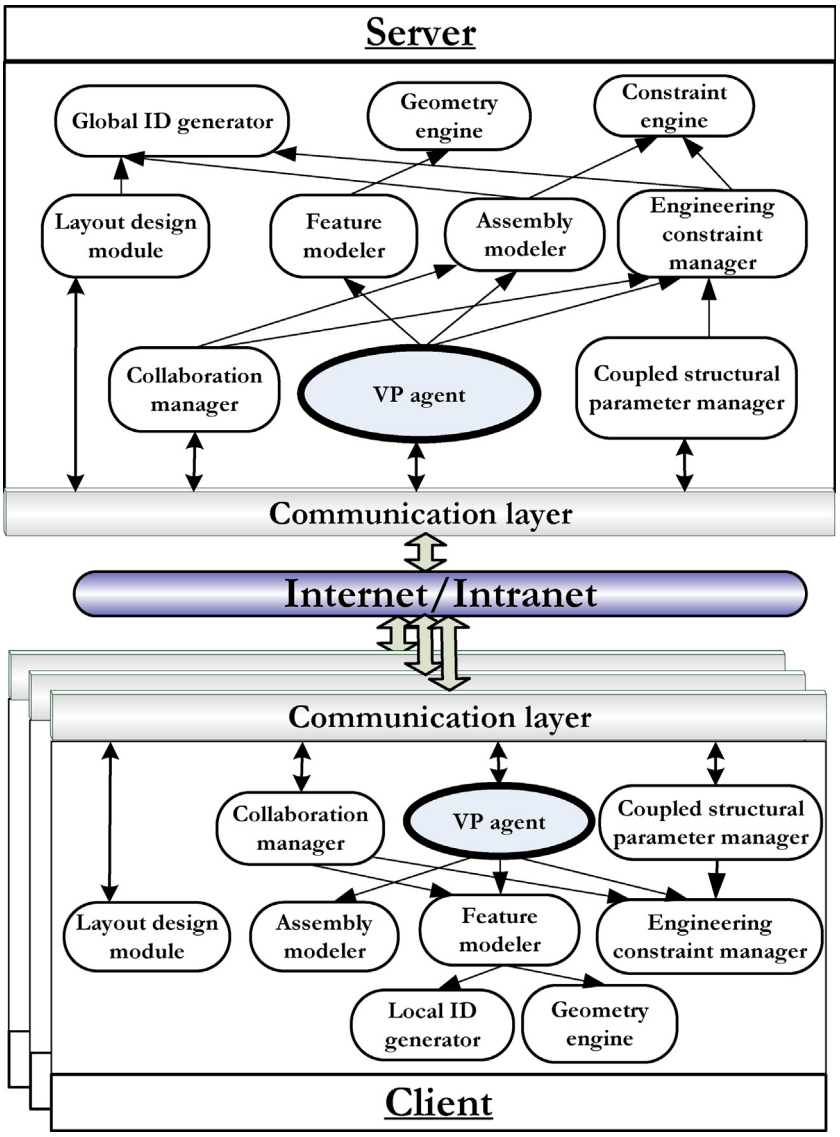


Fig. 1. The system framework for the collaborative top-down assembly design.

very small and the same command always generates the same result in either clients or the server, the synchronization and consistency of the models in the server and clients can be guaranteed. Besides the major function modules of traditional CAD systems such as geometry engine and feature modeler, the clients and the server have the same following function modules.

Layout design module: It provides the functions for the principal designer to set up the layout model of the product based on the conceptual design result. Using these functions, the principal designer uses graphic symbols in a symbol library to construct the layout model. A graphic symbol stands for a component or an assembly scheme, which has the standard engineering semantics and can be understood by engineers. Besides, the graphic symbol includes descriptive information such as component name, component functional requirement, assembly scheme name, assembly method description. It does not have any datum or profile curves that are usually used to define the assembly interface. Instead, we defer those information to be constructed at the skeleton and detail design stage to let designers focus on the representation of conceptual design.

Assembly modeler: It provides functionalities for designers to conduct assembly design in a top-down and collaborative manner, including skeleton design and detail design of components, collaboratively defining assembly interfaces and constraints between components, creating assembly features.

Coupled structural parameter manager: It is responsible for managing and solving the coupled structural parameters of components that are collaboratively defined by different designers.

Collaboration manager: It provides the basic collaborative facilities such as collaborative annotation, video conference. Usually it will invoke other functional components to accomplish the collaboration.

Variation propagation agent (VP agent): It is in charge of monitoring the design variation and propagating the design variations between the clients and server so as to guarantee the consistency of the distributed assembly model.

It is noticed that the assembly constraints usually need to be solved globally [31,32], the constraint engine is thus only deployed on the server. In addition, the global ID generator on the server and the local ID generator on the client are used to generate the global object ID and local object ID respectively to ensure that the object IDs are identical all over the clients and server.

In the following, we first present our assembly model for collaborative top-down assembly design, then describe the three special functions in our framework, i.e., collaborative determination of coupled structural parameters, agent based design variation

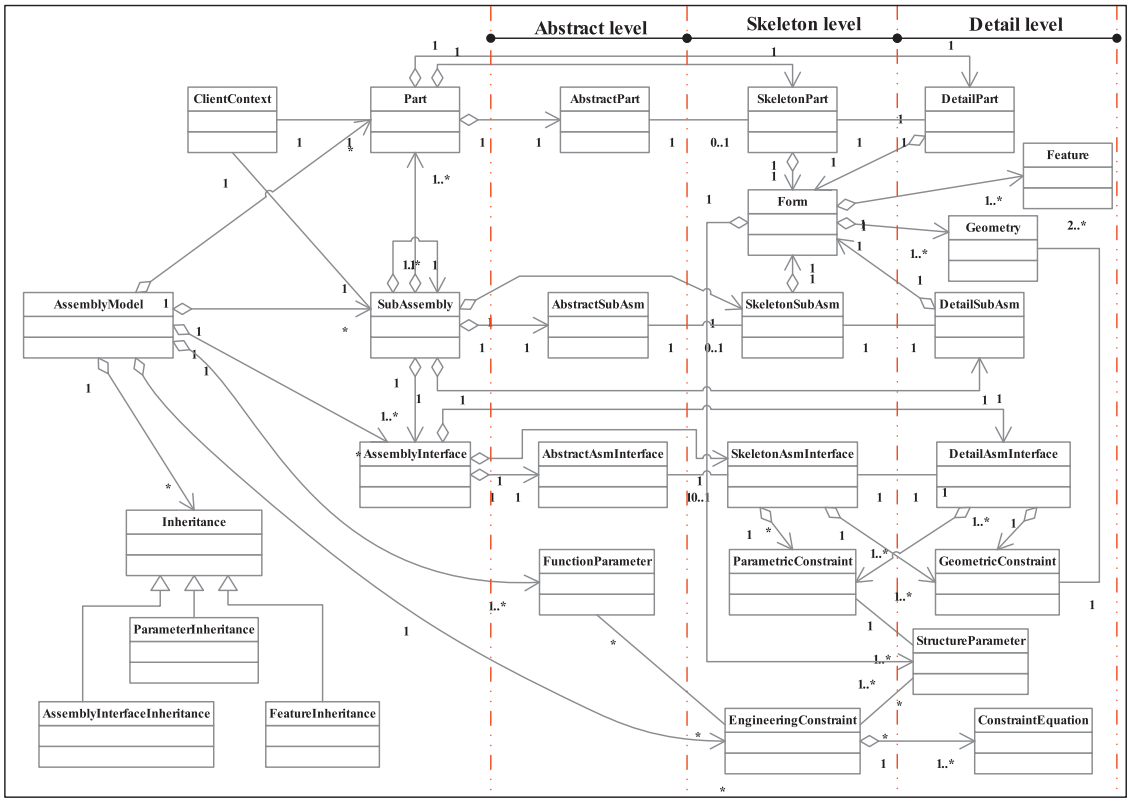


Fig. 2. The assembly model for collaborative top-down assembly design.

propagation and collaborative design of assembly interfaces in more detail.

4. Assembly model for collaborative top-down assembly design

In order to effectively support the collaborative top-down assembly design, we put forward a new assembly model as shown in Fig. 2. Different from the assembly model in [30], which is a multi-level assembly model capturing the abstract information, skeleton information and detailed information to support information transferring and conversion between different design phases in the top-down assembly design process, this assembly model is multi-level and distributed to support designers to collaboratively accomplish assembly design in a top-down manner.

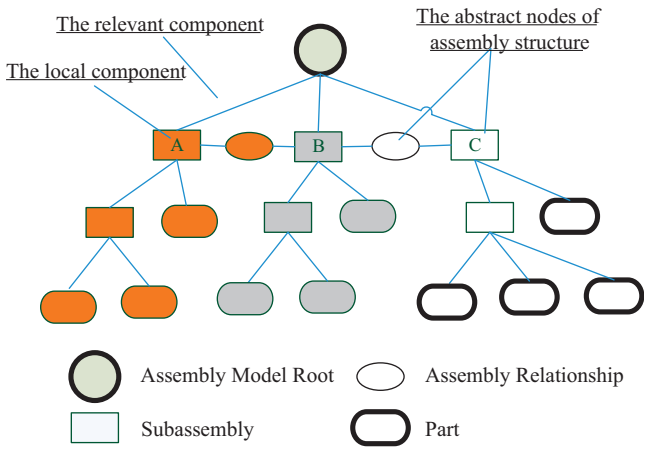


Fig. 3. Illustration of the partial assembly model on client.

(1) Three levels of the assembly model

Our assembly model consists of three levels, each of which is used to represent the design information for one of the three design phases of the collaborative top-down assembly design. The specific three levels are as follows:

Abstract design information level: It consists of AbstractPart, AbstractSubAsm, and AbstractAsmInterface. AbstractPart and AbstractSubAsm mainly contain the function description of the component and AbstractAsmInterface is a specification for the high level assembly scheme.

Skeleton design information level: It is composed of SkeletonPart, SkeletonSubAsm and SkeletonAsmInterface. SkeletonPart refers to the skeleton of the part which is a preliminary 3D geometry of the part with key form parameters and thus can be used as the space and form restriction for the detail design. Similarly, SkeletonSubAsm represents the skeleton for a subassembly. SkeletonAsmInterface contains the assembly interfaces between the skeletons of the two components.

Detail design information level: It includes DetailPart, DetailSubassembly and DetailAsmInterface. DetailPart and DetailSubAsm refer to the final 3D geometry models of the components, and DetailAsmInterface contains all the geometry constraints and parametric constraints between the two components.

Compared with the traditional assembly models, the key point here is that each component and assembly interface has three levels in responding to the three design stages. Each level of the assembly model is constructed by the user during each design stage, i.e., the low level assembly model is not automatically generated by the computer according to the high level information. For example, the abstract design information for the layout design is not translated into the skeleton design information through intelligent algorithm; instead, the objects contained in the skeleton level are built by the users collaboratively according to the abstract design information. To enable the design intent of different levels to be inherited and consistent, the assembly model

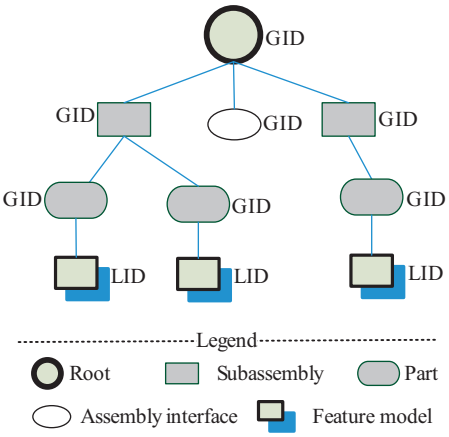


Fig. 4. Illustration of GlobalID and LocalID.

has an inheritance manager object (Inheritance) [30] that manages all the inheritances among different levels of the objects, including the inheritance of features (FeatureInheritance), parameters (ParameterInheritance) and assembly interfaces (AssemblyInterfaceInheritance).

(2) Distribution of the assembly model

Our assembly model is distributed on both the clients and server so as to effectively support collaborative assembly design.

On each client, a partial assembly model is stored, consisting of the following three contents: (1) the model of the component undertaken by the designer on this client (called the local component hereafter), used to support the design of the local component on the client; (2) the models of the components having assembly interfaces with the local components (called the relevant components hereafter), used to help the designer on the client be aware of the relevant design information and collaborate with the related designers to define assembly interfaces and determine coupled structural parameters; (3) the assembly structure of the whole assembly model, used to let the designer know the overall structure of the product. Each node in the assembly structure here stands for a component without geometry information. Fig. 3 illustrates the partial assembly model on the client, where A is the local component, B refers to the relevant component of A, and C as well as others (in white) are the normal nodes in the assembly structure of the whole assembly model.

On the server, a complete and collaborative assembly model including the model distribution information and designers' information is stored. Its main function is to record the whole design results and support the design variation propagation that is imperative for guaranteeing the consistency of the distributed assembly model. In the assembly model on the server, each component has a reference to a ClientContext object which contains the corresponding client information including its IP address, the designer on the client. The designer's information consists of the designer's ID, the designer's role, the designer's authority and state, and so on.

To achieve the association between the corresponding components on the clients and the server, a new ID mechanism is adopted in our distributed assembly model. In the new ID mechanism, the IDs of objects are divided into two types as shown in Fig. 4: GlobalID (GID) and LocalID (LID). GIDs refer to the IDs of part, subassembly and assembly interface and are generated and maintained on the server, whereas the LIDs are the IDs of the objects in the feature model of a part and are generated by the local feature modeler. Since the feature modelers on the clients and the server are the same in our framework, the generated LIDs of the same objects in the feature models on the server and different clients must be the same. Similarly, the GIDs of the same objects on the server and different clients are definitely the same as they are uniformly generated and maintained on the server. Therefore, with the help of this ID mechanism, the association and consistency of the distributed assembly model can be achieved.

5. Collaborative determination of coupled structural parameters

One of the key tasks of skeleton design is to figure out the key structural parameters (KSPs) of every component. It is recognized that some KSPs of different components may be coupled because they must satisfy the same function requirement and thus have to be determined by the related designers in a collaborative way. These KSPs are called coupled structural parameters (CSPs), and they have the following characteristics: distributed, fuzzy, and being determined based on designer's knowledge and experience.

According to the characteristics of CSPs, we propose a collaborative and optimal approach to determining CSPs. In the approach, triangular fuzzy numbers are adopted to represent imprecise parameters and utility functions are utilized to express the designers' preferences which reflect their knowledge and experience. As shown in Fig. 5, the approach consists of the following four steps. Due to the space limitation, the detail of each step is omitted.

Step 1: Collaborative defining of CSPs

Based on the functional requirements, the related designers collaboratively define the coupled structural parameters. In view that the parameters are usually fuzzy instead of precise in early design stage, we adopt triangular fuzzy numbers to represent the structural parameters defined by designers. Meanwhile, to reflect knowledge and experience of designers, we use utility functions and parameter weights to indicate the preference of designers.

With triangular fuzzy number, a parameter can be denoted as:

$$x = (l, m, t) \tag{1}$$

Its membership function is:

$$\mu(x) = \begin{cases} \dfrac{x-l}{m-l} & l \le x \le m \\ \dfrac{x-t}{m-t} & m \le x \le t \end{cases} \tag{2}$$

Suppose the number of structural parameters is p, then they can be defined as a vector:

$$X = [x_1, x_2 \cdots x_p] \in R^p \tag{3}$$

Then the corresponding membership function for X is:

$$\mu(X) = (\mu_1(x_1), \mu_2(x_2) \cdots \mu_p(x_p)) \tag{4}$$

To represent the designers' preference which reflects their knowledge and experiences, utility functions are utilized for structural parameters of X which is denoted as:

$$U(X) = (u_1(x_1), u_2(x_2) \cdots u_p(x_p)) \tag{5}$$

Each element of $U(X)$ in (5) is a function of analytic representation such as quadric curve, exponential function or logarithmic function and so on.

The membership functions and utility functions of fuzzy numbers are synthesized to get the final utility:

$$U^F(X) = U(X) \cdot \mu(X) = \sum_{i=1}^{p} u_i(x_i)\mu_i(x_i) \tag{6}$$

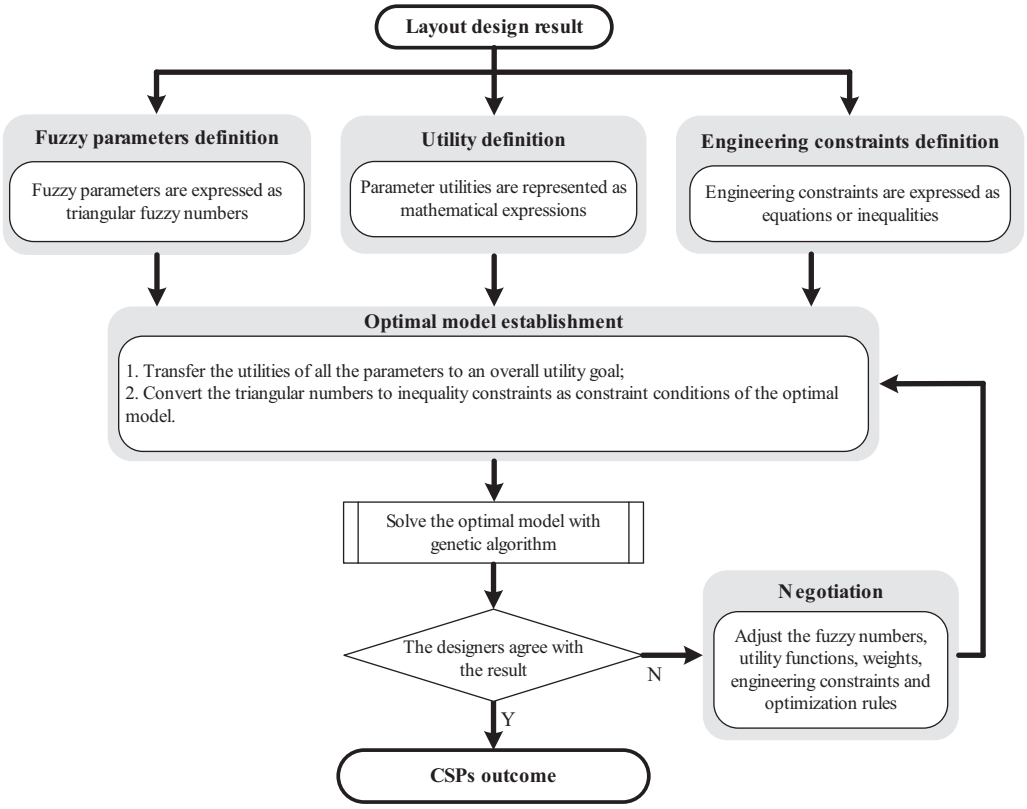


Fig. 5. The flowchart of collaborative determination of CSPs.

Step 2: Collaborative setup of parameter constraint equations

Based on physical principles, the constraint equations on the CSPs defined are established and expressed as equations and inequalities:

$$\begin{aligned} &g_i(X) \le 0 \quad i = 1, 2, \ldots, n \\ &h_j(X) = 0 \quad j = 1, 2, \ldots, m \end{aligned} \tag{7}$$

Step 3: Establishing and solving of optimal model

The overall utility in (6) is taken as the optimization objective and the triangular fuzzy parameters in (1) are converted to inequality constraints:

$$l \le x \le t \tag{8}$$

Considering constraints of (7) and (8), the final optimization model is defined as:

$$\begin{aligned} &X = [x_1, x_2, \ldots, x_p] \in R^p \\ &\max \quad U^F(X) \\ &s.t. \quad g_i(X) \le 0 \quad i = 1, 2, \ldots, n \\ &\quad h_j(X) = 0 \quad j = 1, 2, \ldots, m \\ &\quad l_k \le x_k \le t_k \quad k = 1, 2, \ldots, p \end{aligned} \tag{9}$$

The optimization model is solved by genetic algorithm with Matlab which is integrated into our system.

Step 4: Determination of final CSPs

If the results generated in step 3 cannot satisfy designers, designers can negotiate with each other to adjust the CSPs by collaboratively modifying the triangular fuzzy numbers, utility functions and so on.

Based on the above approach, in our framework, the coupled structural parameter managers (CSPM) for the server and the clients are developed and responsible for collaborative determination of CSPs. Specifically the CSPM on the server is in charge of storing and managing of CSPs, and solving of parameter constraints, whereas the CSPM on the client supports the designer to interactively define CSPs and constraints on the client. The CSPMs on the server and clients keep connected during the collaborative determination process of CSPs. As shown in Fig. 5, the approach takes into account the engineering constraints, i.e., the physical principles to ensure the computation validation, which are the traditional method used by designers to compute the parameters. Furthermore, our approach facilitates the collaborative determination of CSPs in the following ways:

(1) The triangular fuzzy number helps the users to select an interval value instead of a single value for a structural parameter. This is more flexible than the trial-and-error method in which the user repeatedly selects a single value and checks its validity through engineering computation.

(2) Utility functions are utilized to formalize the negotiations between related designers, which are based on the optimization of preference.

6. Agent based design variation propagation

6.1. Overview of variation propagation

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 models 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. In general, the variation propagation during the collaborative top-down assembly design can be classified into the following three types.

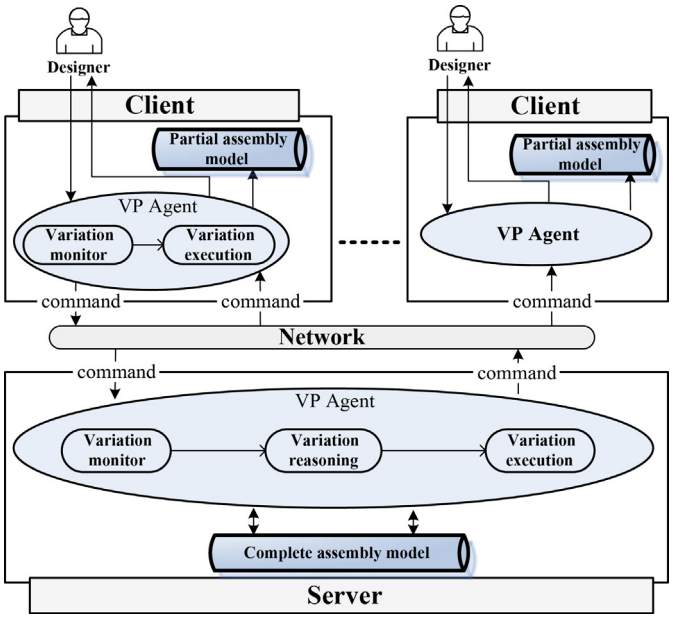


Fig. 6. Schematic overview of agent based design variation propagation.

(1) Hierarchical variation propagation. During the collaborative top-down assembly design, the design information of the previous design stage should be inherited or shared in the next design stage. If an object (e.g. a feature, a constraint, or a parameter) generated in the skeleton design stage is changed, its counterpart generated in the detail design stage should be updated accordingly.

(2) Propagation of constraint variation. The assembly constraints and engineering constraints in our assembly model form a distributed constraint network. During the collaborative top-down assembly design, when any constraint variation happens, the variation should be propagated to keep the whole distributed constraint network satisfied.

(3) Propagation of feature variation. When the feature variation including feature addition, feature deletion and feature parameter modification occurs at any client, it should be propagated to the server as well as the relevant clients to guarantee the feature models consistency of the corresponding parts distributed on the clients and the server.

Agent technology promises much for collaborative product design. Much work in this area focuses on integrating the legacy systems to form an agent based distributed system [23,33]. Furthermore, while there are some work on agent based collaborative product design [34–36], few literature mentions to support variation propagation for collaborative design. So 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 Fig. 6. The agent based approach is suitable for variation propagation because of agents' special characteristics described below:

(1) Autonomy. Agents have the capabilities to accomplish a given task or make decision without human intervention. This is important because the server should itself determine the assembly change without human intervention. Also, the agents on the clients are able to update the local assembly model by itself, which avoids interrupting the work of the users.

(2) Social ability. Agents are able to interact with each other through communication and coordination mechanism. This makes it possible for the agents in the framework to collaborate and negotiate with each other to determine how to update the distributed assembly model.

(3) Reactivity. Agents reside in the system and react according to the environment appropriately. This means the agent can keeps idle only if a design variation happens, when the agent will take actions to guarantee the assembly model to be consistent.

In the following, we will present the main aspects of the approach including task allocation and negotiation algorithms and reasoning schemes of the agent.

6.2. Task allocation and collaboration mechanism of agents

Based on the multi-agent system as shown in Fig. 6, the task allocation algorithm is straight-forward. The agent on the server is the pivot for variation propagation. It is responsible for monitoring the variation requests from the clients and inferring all the assembly model objects that need to be changed accordingly on the server and the related clients. The main task of the VP agent on each client is to monitor the variation requests from both the local designer and the server and realize the corresponding variation on the partial assembly model through invoking certain modeling operations on the client.

For multi-agent system, agent communication language (ACL) is popular and effective for the agents' communication. In our approach, ACL represented in XML (eXtensible Markup Language) is used to support the agents to communicate with each other. On the other hand, token based technology is adopted for the agents to modify the assembly mode which avoids the conflicts happened between agents while reduces concurrency. This makes the negotiation of agents much easier which includes consecutive actions as follows:

(1) Variation monitoring. The VP agent on the server concurrently detects the variation requests from all the VP agents on the clients, through which any design variation made by any designer is monitored.

(2) Variation reasoning. According to the design variation monitored, the VP agent on the server first infers all the objects of the assembly model on the server that need to be changed accordingly, and then infers all the objects in the partial assembly models on the clients that need to be modified accordingly based on the multi-level and distributed assembly model.

(3) Variation execution. For each variation request monitored, the modeling operations in the variation request are first executed on the server to update the assembly model there, and then the VP agent on the server synthesizes a suitable task for the VP agents on the related clients. The task is comprised of a set of actions described as commands that the VP agents on the related clients need to execute to update the partial assembly models on the clients.

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

6.3. Variation reasoning of server VP agent

VP agent on the server is the pivot for variation propagation and variation reasoning is its most important function whose flowchart is shown in Fig. 7.

The key issue 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. 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. Furthermore, there are three basic variation reasoning modules as

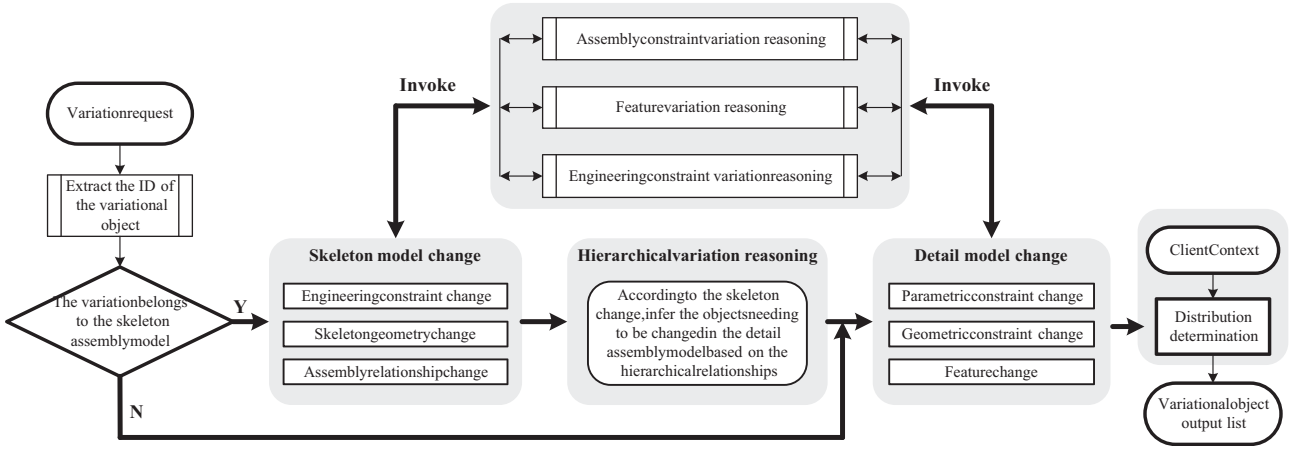


Fig. 7. The flowchart of variation reasoning of the VP agent on the server.

shown in Fig. 7 which are used for variation reasoning of assembly constraints, features and engineering constraints.

(1) Hierarchical variation reasoning.

Based on the distributed and hierarchical assembly model, the hierarchical variation reasoning is realized in the following ways.

(a) Parameter sharing based reasoning. All the parameters that are shared by the skeleton and detail assembly model are traced. When any of the parameters is changed, the skeleton and detail assembly model will both be changed accordingly.

(b) 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.

(c) 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.

(d) Assembly constraint reference based reasoning. 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.

(2) Reasoning of engineering constraint variation.

Engineering constraint expressed as algebraic equations forms a hierarchical and distributed engineering constraint network which is built collaboratively to indicate the collaborative design intent. The changed sub-networks are isolated from the complex engineering constraints belonging to different locations and solved through the constraint engine.

(3) Reasoning of feature variation.

Feature variations include feature addition, feature deletion and feature parameter modification. For feature addition, the parameters used to create a feature such as the feature type, feature parameter are recorded and notified to the related agents. Feature deletion will update the distributed constraints between features, which means the constraints related to the deleted feature are removed. Feature parameter changes are usually related to the parameter relationships among the feature parameters of different parts or subassemblies. The influenced parameters are figured out through engineering constraint engine. As a result, the related changed features are obtained.

(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. The variation of assembly constraint is the addition or deletion or modification of the assembly constraint. The agent will invoke the constraint solver to figure out the transform matrices of every subassembly and part which are affected by the changed assembly constraints.

7. Collaborative design of assembly interfaces

Assembly interfaces are the important content of product design which captures the intent of the design engineer with regard to the relative position of assembly components [37]. In collaborative top-down assembly design, assembly interface design has its own characteristics: (1) for the assembly interfaces between the components that are designed by different designers on different clients, they need to be designed in a collaborative way so as to make the designed assembly interfaces satisfy the requirements of both sides; (2) since the product is designed in a top-down manner, the assembly interfaces should be designed in a top-down manner too, being compatible with the top-down design process. In this work, the specific design process of assembly interfaces consists of the following three steps:

(1) Collaboratively determine assembly scheme: In layout design phase, the designers responsible for the relevant components collaboratively determine the desired assembly scheme between the components according to the function requirement of the product. In this phase, the designers do not need to consider the specific form and geometric constraints of the assembly interface.

(2) Collaboratively create assembly features: According to the assembly scheme determined, in the next design phase, the related designers collaboratively determine the overall form of the assembly interface first, and then each one create the corresponding assembly features on the components she/he is responsible for, guided by the overall form of the assembly interface. In order to guarantee the assembly features created on the relevant components compatible, in our framework, the assembly features being created on the client are transferred to the related clients for reference in real-time using design variation propagation function.

(3) Collaboratively define assembly constraints: Based on the assembly scheme determined and the assembly features created, the related designers collaboratively determine the allowed degree-of-freedoms between the relevant components further and collaboratively define the corresponding geometric constraints and

parameter constraints on the related geometric entities and parameters of the created assembly features to set up the complete assembly constraints.

It is noted that, in the detail design phase, all the assembly interfaces established in the skeleton design phase are automatically inherited, and some of them are refined according to the detail geometry generated in this phase.

8. Implementation

We have developed a prototype system named CTDAD based on the proposed framework. CTDAD is implemented with Visual C++6.0 under windows XP operating system. The architecture of CTDAD is shown in Fig. 1 and is described in Section 2. Some other implementation details include: (1) TCP/IP [38] is adopted as the low level communication protocol between the server and clients; (2) ACL (agent communication language) [39] represented in XML is employed for supporting the communication between VP agents on the server and clients; (3) MATLAB [40] is used as the solver for calculating the coupled structural parameters; (4) ACIS 5.0 [41] is utilized as the geometric kernel of CTDAD.

For these modules to be integrated into the system while ensuring its reliability and stability, we adopt following steps. Firstly, multi-thread method is employed to make the modules to work independently and concurrently. For example, a fundamental thread runs in a blocked way which invokes TCP/IP APIs to receive network stream. The raw stream is then translated into a basic data structure and pumped into a queue which will be processed by modules of upper levels. Secondly, Matlab is integrated into the system as an equation solver. Generally there are two ways to call Matlab from the users' programs. One is called 'MATLAB Compiler' which builds Matlab code into the end users' applications. The other is called 'COM interface' and is what we used in our system. Based on this method, Matlab acts as a computation engine and provides a library of functions to start and end a MATLAB process, send data to and received data from MATLAB, and send commands to be executed by MATLAB. The typical APIs of 'COM interface' usually have a prefix of 'eng' such as engOpen, engClose and so on. Lastly, object oriented programming technology is the guiding principle to implement the prototype system. Following this way, ACIS, the basic geometric kernel can be easily put to use because it wraps its basic framework with the MFC (Microsoft Foundation Classes) as model-view architecture.

Although it takes about ten seconds the first time to start the process of Matlab engine, it is acceptable because Matlab engine runs on the server side and only needs to start with the initialization of the sever. After that, Matlab usually has high efficiency on solving equations. For those situations where real-time is of great importance, a new mathematical kernel is needed to improve the overall performance, which will be our future work.

9. A case study

Using the prototype system developed, we have conducted several experiments to validate the proposed framework. As an example, the collaborative top-down assembly design of a manipulator using CTDAD is described below with emphasis on the key functions of the framework including collaborative determination of coupled structural parameters, collaborative assembly interface establishment and design variation propagation.

Fig. 8 shows the layout assembly model of the manipulator that set up by the chairman according the result of the conceptual design. The layout assembly model consists of six key subassemblies, i.e. Hand, Hand driver, Hand deliver, Swing equipment, Elevator and Base, and each of them is assigned to a designer, say, Hand to Designer_1(DS1), Hand driver to Designer_2(DS2), Hand deliver to Designer_3(DS3), . . ., Base to Designer_6(DS6).

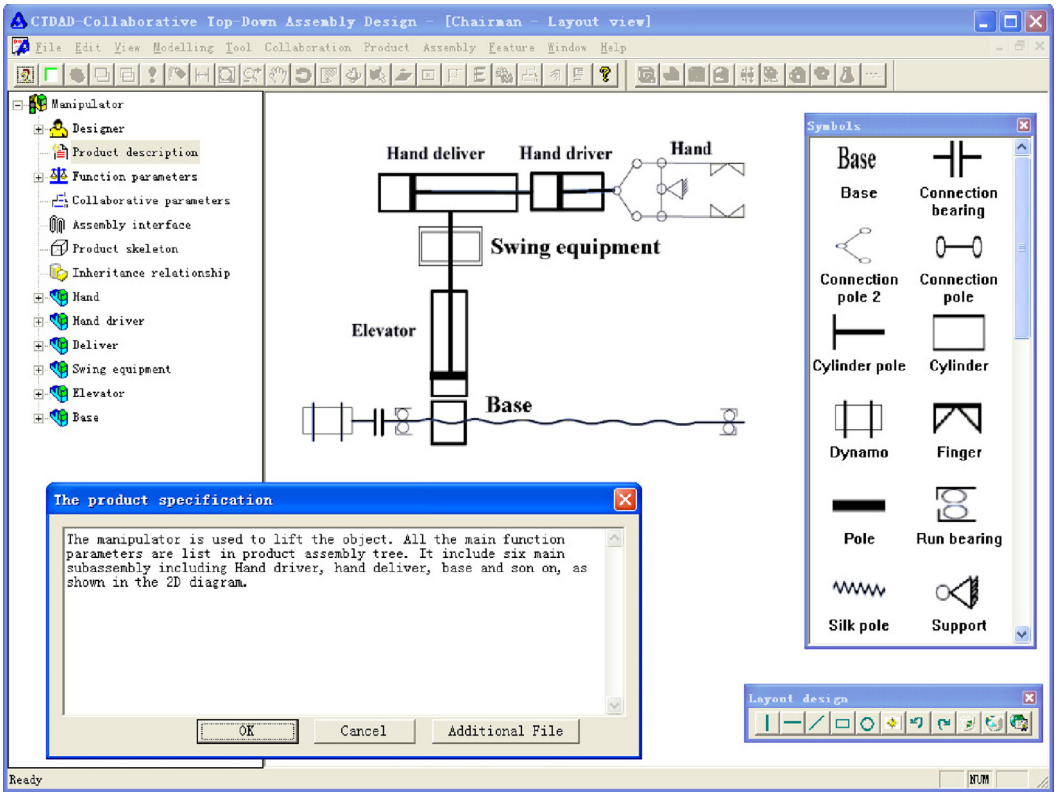


Fig. 8. Layout model of the manipulator.

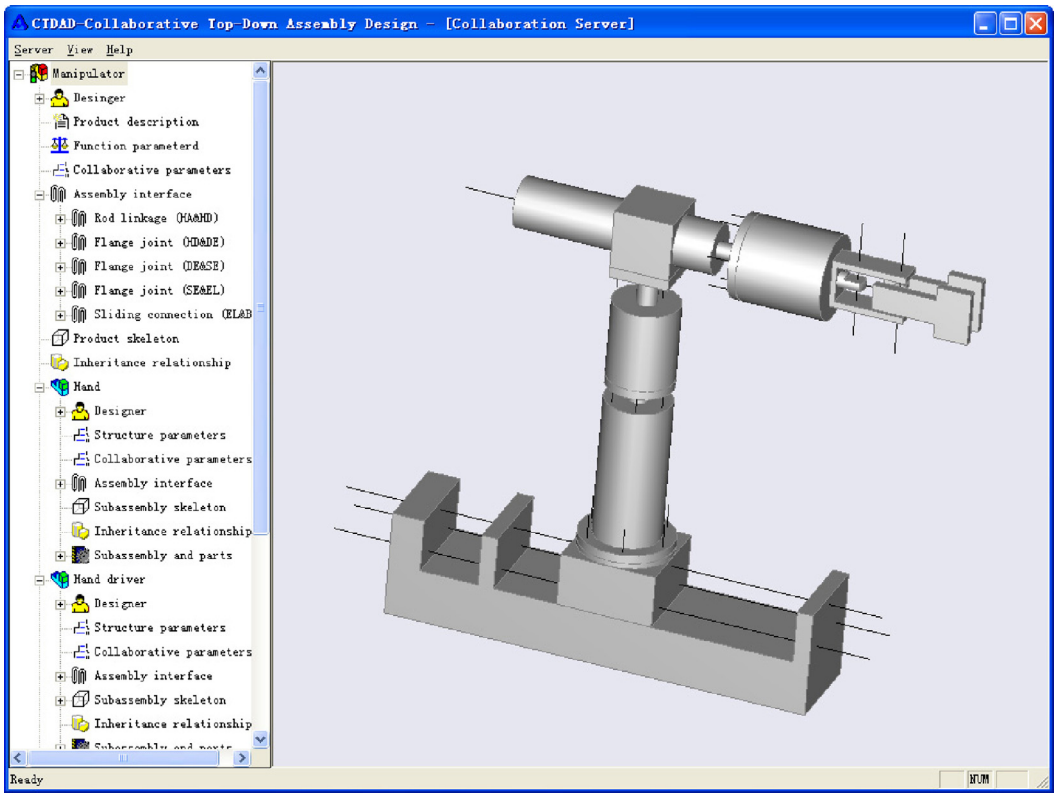


Fig. 9. Skeleton model of the manipulator.

Table 1
Coupled parameters information of manipulator.

Symbol	Triangular fuzzy number	Utility	Weight	Coupled relationship				Result
				HA	HD	HE	BA	
Hl	(500,550,600) mm	F2	0.48	R	R	R	R	550 mm
Hd	100 mm	n/a	n/a	R	R	R	R	100 mm
wd	(60,75,85) mm	F3	0.06	divonx;	□	□	□	75 mm
b	(35,45,60) mm	F1	0.05	divonx;	□	□	□	48 mm
c	(90,110,130) mm	F4	0.05	divonx;	□	□	□	118 mm
al	(15,20,30)°	F2	0.05	divonx;	□	□	□	18°
Dist	(20,40,50) mm	F2	0.05	divonx;	R	□	□	40 mm
Lp	(80,100,120) mm	F4	0.09	R	divonx;	R	R	100 mm
Lc	(50,60,70) mm	F3	0.08	R	R	divonx;	R	64 mm
Ls	(180,200,220) mm	F4	0.09	R	R	R	divonx;	220 mm

HA: Hand; HD: Hand driver; HE: Hand deliver; BA: Base; divonx;: Belong to; □: Unrelated; R: Related; Utility: See Table 2.

Some CSPs existing in the skeleton model of the manipulator are shown in Fig. 9, among which Lp and Lc need to be collaboratively determined by DS2 and DS3. Fig. 10 shows the user interface for collaborative defining of CSPs, by which the related designers can interactively input the fuzzy parameters and determine the utility functions in a collaborative manner. The detailed information about the CSPs of the manipulator is listed in Table 1, including the triangular fuzzy numbers, the utility functions, weights, and so on. Besides, four kinds of utility functions used are also given in Table 2.

Figs. 11–18 shows how DS1 and DS2 collaboratively design the assembly interface between two subassemblies Hand and Hand driver. Also they are used to show the design variation propagation function supported by CTDAD. As shown in these figures, to effectively support the collaborative top-down assembly design, four types of design views are provided by CTDAD: the standalone design view, the layout assembly model view, the design view for collaborative skeleton assembly design, and the design view for

Table 2
Utility functions.

Expression
$F_1(x) = \begin{cases} \frac{t-x}{t-m} & x > m \\ \frac{x-l}{m-l} & x \le m \end{cases}$
$F_3(x) = \begin{cases} e^{\frac{-(x-m)}{b_r}} & x > m \\ e^{\frac{(x-m)}{b_l}} & x \le m \end{cases}$
$F_2(x) = \begin{cases} z_r \cdot x^2 + \frac{(z_r \cdot m^2 - z_r \cdot t^2 - 1)}{t-m} \cdot x - \frac{t(z_r \cdot m^2 - z_r \cdot m \cdot t - 1)}{t-m} & x > m \\ z_l \cdot x^2 + \frac{(z_l \cdot m^2 - z_l \cdot l^2 - 1)}{l-m} \cdot x - \frac{l(z_l \cdot m^2 - z_l \cdot m \cdot l - 1)}{l-m} & x \le m \end{cases}$
$F_4(x) = \begin{cases} c_r \cdot \ln\frac{x}{t} & x > m \\ c_l \cdot \ln\frac{x}{l} & x \le m \end{cases}$

x: fuzzy parameter; l, m, t: triangular fuzzy numbers, i.e. (l, m, t); the other symbols are constants of different function expression.

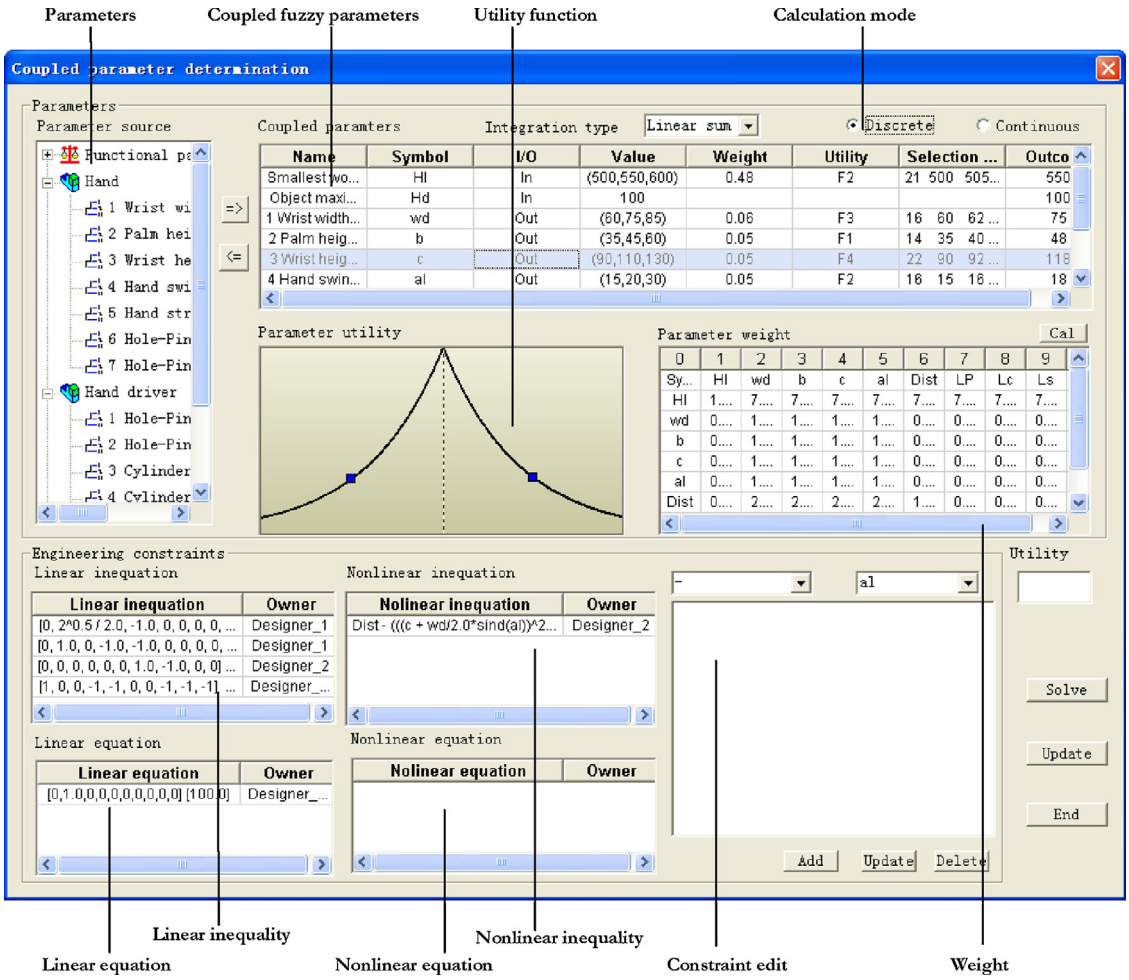


Fig. 10. User interface for collaborative determination of CSPs.

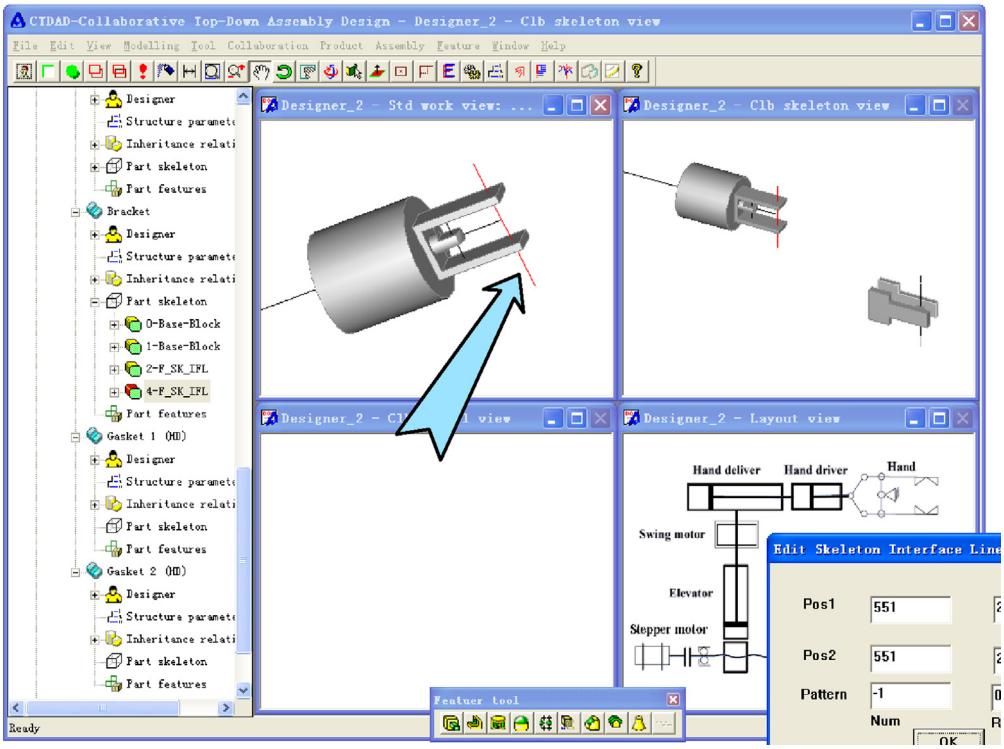


Fig. 11. An skeleton assembly feature created by DS2.

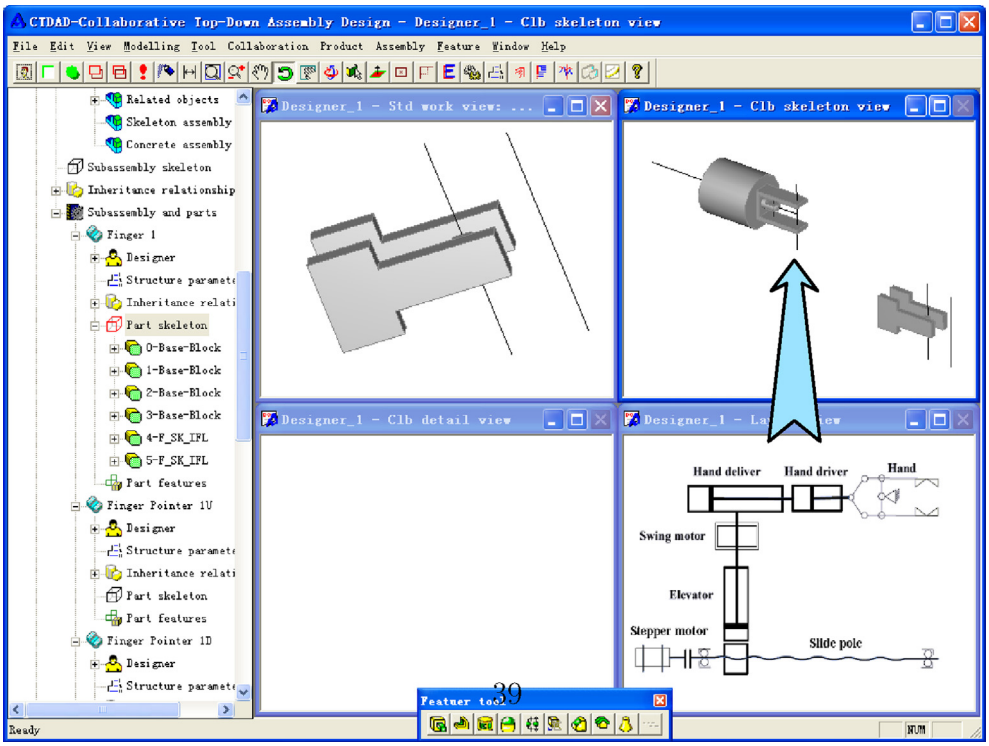


Fig. 12. The skeleton assembly feature created by DS2 is propagated to the Client1(DS1).

collaborative detail assembly design. The specific process of the collaborative assembly interface design shown in Figs. 11–18 consists of the following two steps:

(1) Collaborative design of skeleton assembly interface. As an example, DS2 first creates the skeleton assembly feature of a skeleton assembly interface as shown in Fig. 11 and the skeleton assembly feature created is propagated to the Client1(DS1) in real time as shown in Fig. 12. Then DS1 sets up an assembly constraint collinear between two skeleton assembly features as shown in Fig. 13, and the assembly constraints established are solved on the server and the results are propagated to both Client1(DS1) and Client2(DS2). The final skeleton assembly interface designed is shown in the top-right view of Figs. 14 and 15.

(2) Collaborative design of detail assembly interface. Based on the skeleton assembly interface created, the detail assembly features are created in detail design phase. Figs. 14 and 15 show the

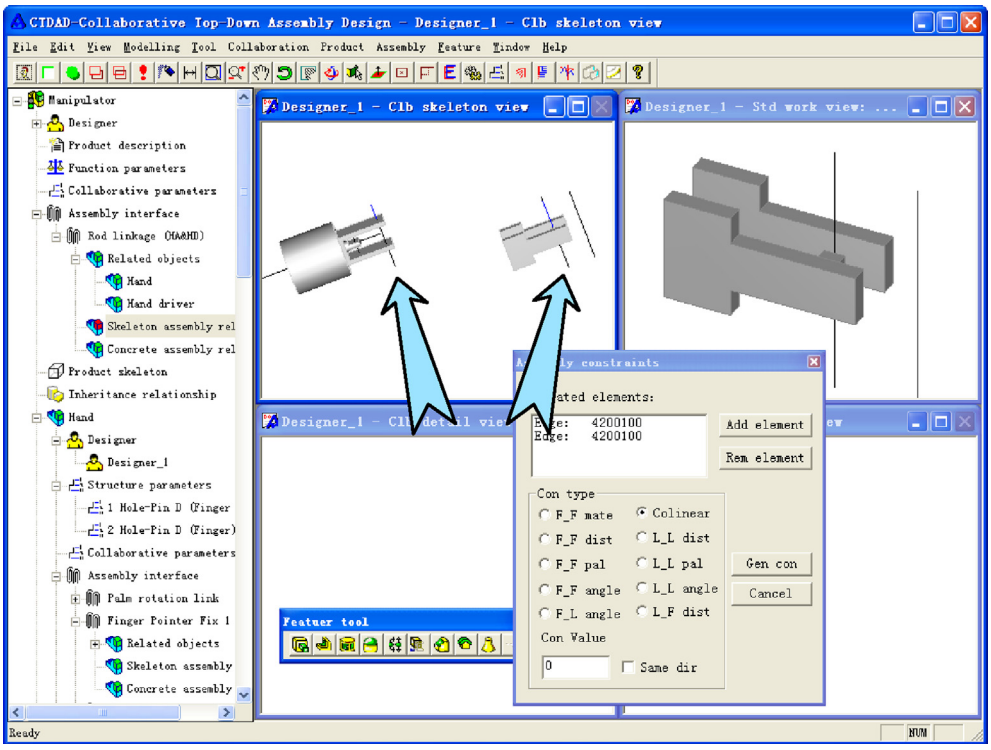


Fig. 13. DS1 defined an "collinear" constraint between two skeleton assembly features.

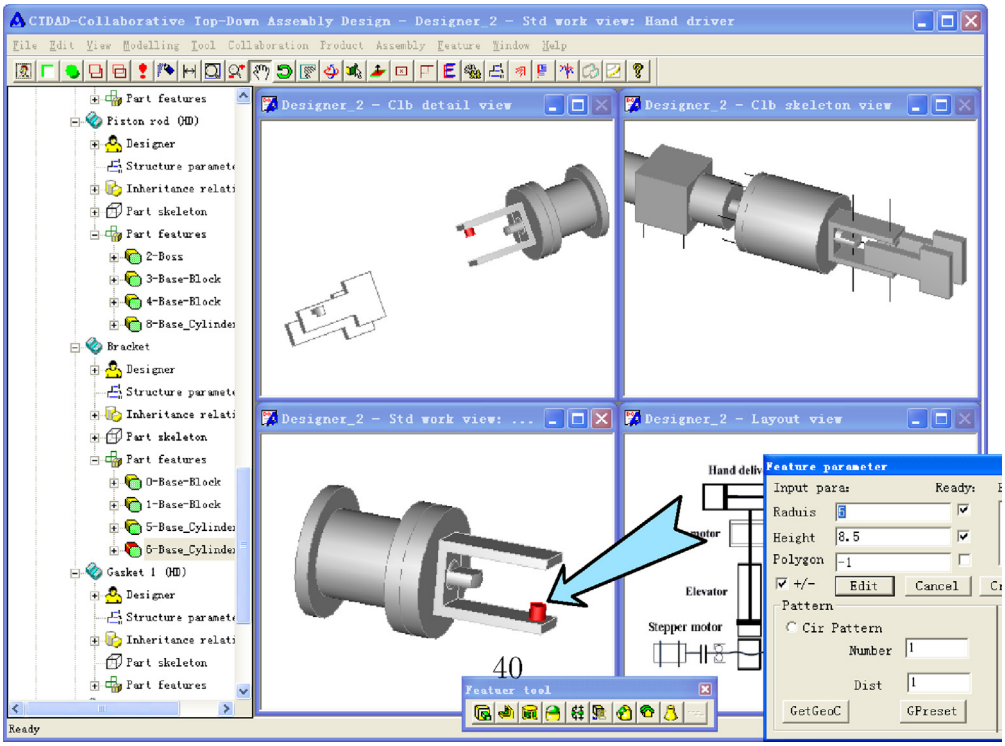


Fig. 14. An detail assembly feature created by DS2.

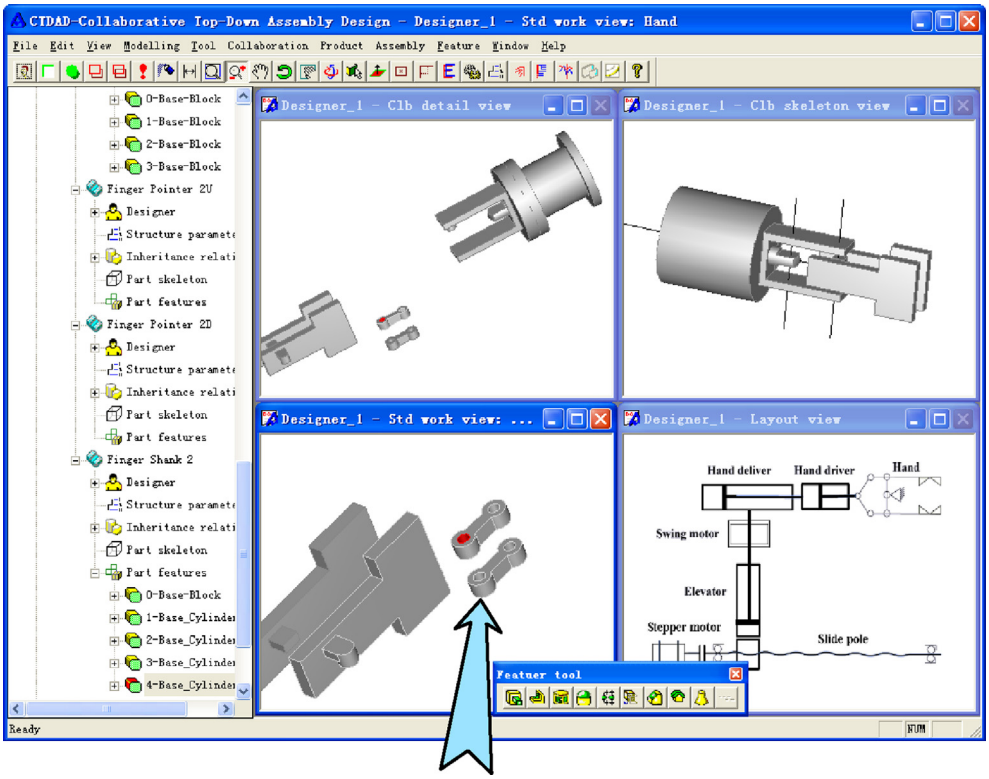


Fig. 15. Parts created by DS1 for defining the detail assembly interface.

detail assembly features further created by DS1 and DS2 and the whole results are shown in Fig. 16. In this step, the related information of skeleton assembly interface is inherited by the corresponding detail assembly interface and the association between them is established. As shown in Fig. 17, the axis in the skeleton assembly interface is inherited by the corresponding detail assembly interface, which makes it possible that the design variation in the skeleton assembly interface will be automatically propagated to the detail assembly interface. Consequently, the assembly constraints involved in the detail assembly interface are solved on the server and propagated to the Client1(DS1) and Client2(DS2). The final detail assembly interface designed is shown in Fig. 18.

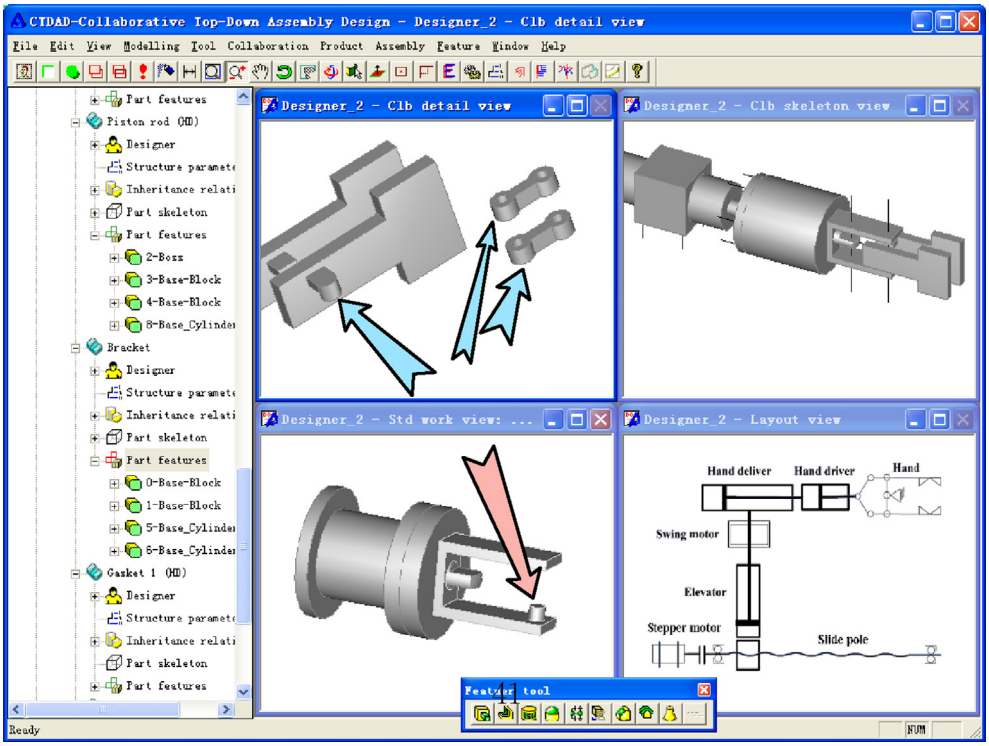


Fig. 16. The assembly features created by DS2 and the assembly features propagated from the Client1.

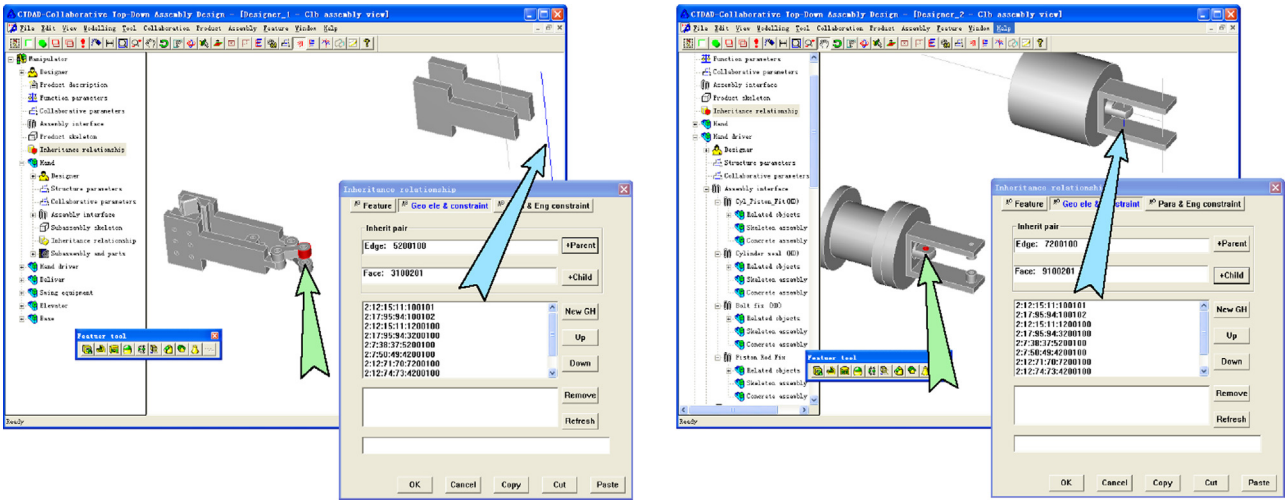


Fig. 17. The axis in the skeleton assembly interface is inherited by the corresponding detail assembly interface (DS1 and DS2).

10. Conclusions and future work

With a comparison to one typical study of Hwang et al. [42,43] on top-down product design, we conclude our work. Hwang et al. presented an approach using neutral reference model (NRM) for representation and propagation of engineering change information in collaborative product development, which can support top-down product design. The difference between our approach and NRM is highlighted in Table 3. It shows that, to support the early design stage, both NRM and our approach use abstract geometry such as datum axis, datum plane to specify interface between components, geometric constraints and so on. Our approach also gives a feature based method for users to construct a simplified component shape or design spaces at early design stage. NRM is CAD system independent and can support collaborators with different CAD systems. Our approach is based on CAD system developed by us (using the ACIS geometric kernel). Also parameter inheritance and conceptual representation are considered in our approach.

Table 3
Comparisons of our approach with NRM [42,43].

Items for comparison	Our approach	NRM
Feature support	✓	
Abstract geometry	✓	✓
CAD system independent		✓
Design space	✓	
Version control		✓
Concept design representation	✓	
Parameter inheritance	✓	

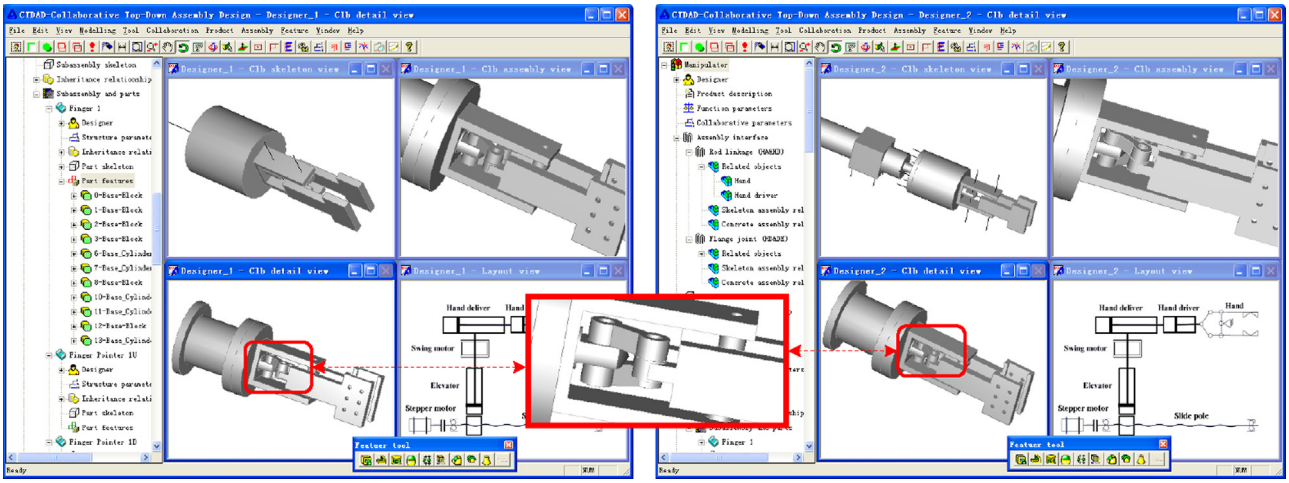


Fig. 18. The final assembly interface shown on DS1 and DS2.

To summarize, the main contribution of our approach lies in that it allows a group of designers to collaboratively conduct complex product design in a top-down manner. Specifically, it has the following characteristics:

(1) A multi-level and distributed assembly model is adopted. The assembly model consists of three levels, each of which represents the design information for one of the three design phases of the collaborative top-down assembly design. Moreover, the assembly model comprises the partial assembly model on the clients and the complete assembly model on the server. Therefore it can effectively support collaborative top-down assembly design.

(2) Collaborative determination of coupled structural parameters is supported. In the approach, triangular fuzzy numbers are adopted to represent imprecise parameters and utility functions are utilized to express the designers' preferences so that the determined CSPs are optimal.

(3) Agent based design variation propagation is achieved. The variation propagation agents on the server and clients can effectively realize the design variation propagation between the clients and the server and ensure the consistency of the multi-level and distributed assembly model during the whole design process.

(4) Collaborative design of assembly interfaces is supported. The assembly interfaces between the components that are designed by different designers on different clients can be collaboratively designed by the related designers in a top-down way, which can solve conflicts as early as possible and make the designed assembly interfaces satisfy the requirements of all the related designers.

Our future work includes: solving of the design conflicts that may occur in collaborative top-down assembly design; and extension to supporting the whole product design process including conceptual design; more validation experiments needs to be conducted.

Acknowledgments

The authors are very grateful to the financial support from NSF of China (No. 61173125). This research was also carried out as a part of the CASES project which is supported by a Marie Curie International Research Staff Exchange Scheme Fellowship within the 7th European Community Framework Programme under the grant agreement No. 294931. The work reflects only the authors views and the Union is not liable for any use that may be made of the information contained therein.

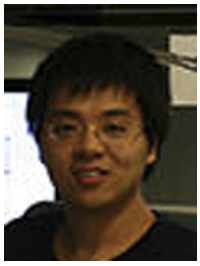
References

[1] G. Pahl, W. Beitz, K. Wallace, Engineering Design: A Systematic Approach, Springer Verlag, 1996.
[2] Z. Li, X. Jin, Y. Cao, X. Zhang, Y. Li, Architecture of collaborative design grid and its application based on lan, Advances in Engineering Software 38 (2007) 121–132.
[3] J. Chung, T. Hwang, C. Wu, Y. Jiang, J. Wang, Y. Bai, H. Zou, Framework for integrated mechanical design automation, Computer-Aided Design 32 (2000) 355–365.
[4] M. Mantyla, A modeling system for top-down design of assembled products, IBM Journal of Research and Development 34 (1990) 636–659.
[5] Y. Han, K. Lee, A case-based framework for reuse of previous design concepts in conceptual synthesis of mechanisms, Computers in Industry 57 (2006) 305–318.
[6] C. Li, S. Tan, K. Chan, A qualitative and heuristic approach to the conceptual design of mechanisms, Engineering Applications of Artificial Intelligence 9 (1996) 17–32.
[7] M. Campbell, J. Cagan, K. Kotovsky, Agent-based synthesis of electromechanical design configurations, Journal of Mechanical Design 122 (2000) 61–69.
[8] Y. Jin, W. Li, Design concept generation: a hierarchical coevolutionary approach, Journal of Mechanical Design 129 (2007) 1012–1021.
[9] G. Galán-Marí n, D. López-Rodrí guez, E. Mérida-Casermeiro, A new multivalued neural network for isomorphism identification of kinematic chains, Journal of Computing and Information Science in Engineering 10 (2010) 011009.
[10] Y. Zeng, A. Pardasani, J. Dickinson, Z. Li, H. Antunes, V. Gupta, D. Baulier, Mathematical foundation for modeling conceptual design sketches, Journal of Computing and Information Science in Engineering 4 (2004) 150–159.
[11] C. Tian, M. Masry, H. Lipson, Physical sketching: reconstruction and analysis of 3d objects from freehand sketches, Computer-Aided Design 41 (2009) 147–158.
[12] E. van den Berg, W. Bronsvoort, Validity maintenance for freeform feature modeling, Journal of Computing and Information Science in Engineering 10 (2010) 011006.
[13] A. Csabai, I. Stroud, P. Xirouchakis, Container spaces and functional features for top-down 3d layout design, Computer-Aided Design 34 (2002) 1011–1035.
[14] N. Aleixos, M. Contero, et al., Integrated modeling with top-down approach in subsidiary industries, Computers in Industry 53 (2004) 97–116.
[15] R. Bracewell, K. Wallace, M. Moss, D. Knott, Capturing design rationale, Computer-Aided Design 41 (2009) 173–186.
[16] P. Zhan, U. Jayaram, O. Kim, L. Zhu, Knowledge representation and ontology mapping methods for product data in engineering applications, Journal of Computing and Information Science in Engineering 10 (2010) 021004.
[17] F. Alizon, S. Shooter, T. Simpson, Reuse of manufacturing knowledge to facilitate platform-based product realization, Journal of Computing and Information Science in Engineering 6 (2006) 170–178.
[18] C. Hoffmann, Constraint-based computer-aided design, Journal of Computing and Information Science in Engineering 5 (2005) 182–187.
[19] G. Brunetti, B. Golob, A feature-based approach towards an integrated product model including conceptual design information, Computer-Aided Design 32 (2000) 877–887.
[20] S. Rachuri, Y. Han, S. Foufou, S. Feng, U. Roy, F. Wang, R. Sriram, K. Lyons, A model for capturing product assembly information, Journal of Computing and Information Science in Engineering 6 (2006) 11–21.
[21] Pro/ENGINEER Advanced Top-Down Design, PTC Global Services, 2002.
[22] W. Li, W. Lu, J. Fuh, Y. Wong, Collaborative computer-aided design research and development status, Computer-Aided Design 37 (2005) 931–940.
[23] T. Mori, M. Cutkosky, Agent-based collaborative design of parts in assembly, in: Proceeding of ASME Design Engineering Technical Conference, 13–16, September, 1998.
[24] N. Shyamsundar, R. Gadh, Internet-based collaborative product design with assembly features and virtual design spaces, Computer-Aided Design 33 (2001) 637–651.

[25] L. Chen, Z. Song, L. Feng, Internet-enabled real-time collaborative assembly modeling via an e-assembly system: status and promise, Computer-Aided Design 36 (2004) 835–847.
[26] R. Bidarra, N. Kranendonk, A. Noort, W. Bronsvoort, A collaborative framework for integrated part and assembly modeling, Journal of Computing and Information Science in Engineering 2 (2002) 256–264.
[27] C. Cera, T. Kim, J. Han, W. Regli, Role-based viewing envelopes for information protection in collaborative modeling, Computer-Aided Design 36 (2004) 873–886.
[28] C. Lu, J. Fuh, Y. Wong, Z. Qiu, W. Li, Y. Lu, Design modification in a collaborative assembly design environment, Journal of Computing and Information Science in Engineering 6 (2006) 200–208.
[29] D. Mun, J. Hwang, S. Han, Protection of intellectual property based on a skeleton model in product design collaboration, Computer-Aided Design 41 (2009) 641–648.
[30] X. Chen, S. Gao, Y. Yang, S. Zhang, Multi-level assembly model for top-down design of mechanical products, Computer-Aided Design 44 (2012) 1033–1048.
[31] J. Pu, K. Ramani, Priority-based geometric constraint satisfaction, Journal of Computing and Information Science in Engineering 7 (2007) 322–329.
[32] K. Sambhoos, B. Koc, R. Nagi, Extracting assembly mating graphs for assembly variant design, Journal of Computing and Information Science in Engineering 9 (2009) 034501.
[33] D. Dornfeld, P. Wright, S. Roundy, A. Rangarajan, S. Ahn, Agent interaction in CAD/CAM, Transactions-North American Manufacturing Research Institution of SME (2001) 569–576.
[34] G. Huang, J. Huang, K. Mak, Agent-based workflow management in collaborative product development on the internet, Computer-Aided Design 32 (2000) 133–144.
[35] M. Mahdjoub, D. Monticolo, S. Gomes, J. Sagot, A collaborative design for usability approach supported by virtual reality and a multi-agent system embedded in a PLM environment, Computer-Aided Design 42 (2010) 402–413.
[36] S. Wu, H. Ghenniwa, Y. Zhang, W. Shen, Personal assistant agents for collaborative design environments, Computers in Industry 57 (2006) 732–739.
[37] R. Pierce, D. Rosen, Simulation of mating between nonanalytic surfaces using a mathematical programming formulation, Journal of Computing and Information Science in Engineering 7 (2007) 314–321.
[38] W. Stevens, TCP/IP Illustrated: The Protocols, vol. 1, Addison-Wesley Professional, 1994.
[39] Fipa, Fipa ACL Message Structure Specification, Foundation for Intelligent Physical Agents, 2005 http://www.fipa.org/specs.
[40] S. Attaway, MATLAB: A Practical Introduction to Programming and Problem Solving, A Butterworth-Heinemann Title, 2011.
[41] J. Corney, 3D Modeling with the ACIS Kernel and Toolkit, John Wiley & Sons, Inc., 1997.
[42] J. Hwang, D. Mun, S. Han, Neutral reference model for engineering change propagation in global top-down modeling approach, International Journal of CAD/CAM 7 (2009) 81–89.
[43] J. Hwang, D. Mun, S. Han, Representation and propagation of engineering change information in collaborative product development using a neutral reference model, Concurrent Engineering 17 (2009) 147–157.

Shuming Gao is a Professor of the State Key Lab of CAD&CG, Zhejiang University. He received his PhD from the Applied Mathematics Department of Zhejiang University in 1990, and was a Visiting Scholar and a Visiting Professor in the Design Automation Lab of Arizona State University, respectively, in 1996 and 2001. His research interests include CAD, virtual prototyping, collaborative engineering, engineering informatics and so on.

Shuting Zhang is a lecturer in the School of Mechanical Engineering & Automation, Zhejiang Sci-Tech University. He received his Ph.D. degree in Computer Science from Zhejiang University in 2009. His research interests include collaborative product design and computer-aided design.



Xiang Chen is an assistant researcher in the State Key Lab of CAD&CG, Zhejiang University. He received his Ph.D. degree in Computer Science from Zhejiang University in 2012. His research interests mainly include image analysis/editing, shape modeling/retrieval and computer-aided design.

Youdong Yang is an associate Professor of Zhijing College, Zhejiang University of Technology. He received his PhD from the Computer Science and Technology Department of Zhejiang University in 2008. His research interests include CAD, CSCW, mechanical engineering and so on.