ARTICLE IN PRESS

Computer-Aided Design ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Computer-Aided Design

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

Multi-level assembly model for top-down design of mechanical products

Xiang Chen, Shuming Gao *, Youdong Yang, Shuting Zhang

State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, PR China

ARTICLE INFO

Keywords:
Top-down assembly design
Top-down component design
Multi-level assembly model
Shape skeleton
Layout skeleton
Skeleton interface
Skeleton feature
Inheritance mechanism

ABSTRACT

To enable next generation CAD tools to effectively support top-down design of products, a top-down assembly design process is refined from the traditional product design process to better exhibit the recursive-execution and structure-evolvement characteristics of product design. Based on the top-down assembly design process, a multi-level assembly model is put forward to capture the abstract information, skeleton information and detailed information involved. The multi-level assembly model is a meta-level implementation and is easy to be extended. Moreover, the inheritance mechanisms are explored to ensure the feasibility of information transferring and conversion between different design phases in the top-down assembly design process. A top-down assembly design sample is analyzed at length to show the application effects of the multi-level assembly model and the relevant inheritance mechanisms. In addition, a practical topic about the model adaptation of existing CAD systems is also discussed for a broader application of the top-down assembly design. Finally, the conclusion of the work and the future directions for further exploration are given.

1. Introduction

Due to the rapid development of global economics environment, many new products possess the characteristics of great complexity and scale, and need knowledge from multiple disciplines. Therefore, how to design these products effectively and efficiently is of great significance. Among various strategies for product design, the top-down approach is a quite prominent and natural way. In a top-down approach an overview of the product is first formulated, and each component (could be a part or a sub-assembly) is then refined in greater detail, sometimes in many additional subcomponent levels, until the base components are defined exactly. In this way the complex design work of a product is subdivided into several simpler design works of sub-modules gradually and recursively, hence to reduce the difficulty and complexity of the design. Meanwhile, these subdivided works could be executed in parallel once most of the interdependence among them has been predetermined. This parallelizability makes design cooperation between different groups possible.

Considering the importance of the top-down approach in product design, computer-based tools and packages should be provided to help designers carry out the top-down product design more easily and conveniently. Unfortunately, with the limited support of most commercial CAD software to the top-down product design nowadays, there are still many design works that cannot be powered up by computers. This will waste too much time in the product design and eventually delay the time the new product enters the market. It is obviously a loss to both companies and consumers.

To make next generation CAD tools supporting the top-down product design better, the following fundamental issues should be considered:

1. A reasonable top-down assembly design process underlaid which is suitable for computerization.
2. An integrated multi-level assembly model for capturing information in different levels of abstraction.
3. Various flexible mechanisms which ensure the transition and association of design information between different design phases.

The work here is exactly meant to explore the novel assembly design process, assembly model and inheritance mechanisms that are required by next generation CAD tools in order to support top-down product design effectively. Specifically, in this paper, a more accurately and compactly depicted top-down assembly design process is refined from traditional product design process. Based on the top-down assembly design process, we present a multi-level assembly model which has the ability to capture the important data and knowledge in design and thus can support different stages of the top-down assembly design. This model is a meta-level implementation and can support mainstream CAD systems through adaptation and extension. Meanwhile, the relevant inheritance mechanisms are explored to ensure the effective transmission and evolvement of design information

* Corresponding author. Tel.: +86 571 88206681x514; fax: +86 571 88206680.
E-mail addresses: xchen@cad.zju.edu.cn (X. Chen), smgao@cad.zju.edu.cn (S. Gao), yyoudong@cad.zju.edu.cn (Y. Yang), zhangshuting@cad.zju.edu.cn (S. Zhang).

0010-4485/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2010.12.008

between different design phases in the whole product design process.

The rest of the paper is organized as follows. Section 2 reviews some previous studies related to this work. Section 3 introduces the top-down assembly design process and analyzes the requirements for the corresponding computer-based supporting tool. In Section 4 we give the details of a multi-level top-down assembly model and Section 5 describes various inheritance mechanisms for top-down assembly design. Section 6 shows a top-down assembly design sample and some relevant applications of the multi-level assembly model. Then in Section 7, the adaptation and extension method for the existing CAD systems is discussed for practical top-down assembly design. Finally, conclusion and future work are provided.

2. Related works

Traditional mechanical design is a top-down process which starts with overall sketch and rough requirements to detailed and refined components gradually. It is well recognized that, in the long history of evolvement in mechanical design, top-down product design is always an important issue and the relevant computer-based tools supporting top-down design are absolutely necessary.

Some works analyze the characteristics of traditional top-down product design and discover the interesting issues about it. Libardi et al. [1] give an overview of the literature before 1988 about the development of computer environments for mechanical assembly design. In the review, support for top-down design and multiple viewpoints is one of the key points. Wen Jian et al. [2] overview the state of the art in the research of top-down product design and point out some problems which need to be overcome in top-down design systems, such as assembly model representation for top-down product design and the reasoning method from conceptual model to parametric model. The research conducted by Mäntylä [3] is a pioneer work which addresses the top-down product design system seriously. In the work, the author points out that the design process could be decomposed into functional design, conceptual design and detail design, while a top-down product design system should support multiple abstraction models for all the three design phases. Many important concepts and issues about top-down design approach in mechanical engineering are also discussed, such as abstract geometry, focus change, geometry inheritance and redesign problem.

The whole top-down product design consists of several design phases dealing with different levels of design information. Many relevant works are presented for specific design phases in the top-down product design.

There are a number of methods and techniques for establishing function structure in conceptual design. Sturges et al. [4] present functional flow charts and functional logic diagrams for function representation. Umeda et al. [5] propose the Function-Behavior-State (FBS) model which associates the function symbols, behaviors and states together, the first one the subjective part and the latter two the objective parts. Karnopp et al. [6] discuss the use of bond graphs in modeling of electrical, mechanical and hydraulic systems. Gui et al. [7] developed a set of behavioral specifications to capture the inter-relationships among components. More details and discussion about these techniques can be found in [8].

Layout design is a very important part in the embodiment design phase which follows the conceptual design. There are also some works concentrating on the development of computer-based tools for layout design. Lashin et al. [9] analyze six levels of abstraction from the coarsest convex hull to the finest geometric model and conclude that the abstraction level 2 model is suited for design of large layouts, in which all the geometry necessary to check function, spatial compatibility, etc., are described. Csabai et al. [10,11] use design spaces and interface features in their 3D Layout Module to determine the kinematic constraints between functional components in layout design. Based on their representation, kinematic analysis could be executed in an early stage during the whole design process. Mantripragada et al. [12] present the concept of DFC (datum flow chain) to capture the fundamental structure of assembly. The logical layout design could be carried out to establish directed chains of dimensional datums to control how parts are located with respect to each other. Besides the works mentioned, Clement et al. [13] present a model called TTRS (technologically and topologically related surfaces) to associate elementary surfaces. Along with TTRS, the MGDE (minimum geometric datum elements) is used to define the reference frames of various surfaces associations. Although the TTRS and MGDE are mainly presented for dimensioning and tolerancing, the idea of the "abstraction of real surfaces" behind MGDE could be potentially used in layout design to help design kinematic relationships.

In the last two decades, feature-based assembly modeling has attracted many researchers' attention. Shah et al. [14] describe the assembly modeling as an extension of feature-based modeling for parts. In the work an assembly feature is used to bind two components together, which is substantially an association between two form features on different parts. Constraints on mating features' shapes and relative positions are defined in assembly features. Holland et al. [15,16] use "Related" and "Relation" as the base classes for both part and assembly modeling. Assembly features are used in both assembly modeling and assembly planning (assembly sequence planning, assembly motion planning, fixture planning, etc.), which include handling features for handling components and connection features for connecting components together. Shyamsundar et al. [17] introduce the concept of virtual space and present a geometric representation AREP for collaborative assembly design. Assembly features in the work are classified into relational assembly features indicating the relation between geometric features, and assembly form features as the result of joining certain shape features of two components together. Singh et al. [18] present assembly ports to group together the interface information between parts. Based on the port representation, analysis for label matching, dimension evaluation, mating constraint solvability, etc. could be carried on to automate the mating definition and reduce the designers' effort. Kim et al. [19] describe their ARM (assembly relation model) and develop an AsD ontology based on ARM which captures the semantics of assembly/joining concepts and relations. The AsD ontology is applied in collaborative product development and shows its capability in maintaining the design intent of assembly relations. However, the presented ontology is not aimed at capturing the design knowledge and information involved in the dynamic top-down assembly design process.

Besides the work that focuses on specific design phases, many researchers have also explored integration methods of the different information representation involved in conceptual design and downstream product development. Kusiak et al. [20] use diagraphs to help the transformation from conceptual design and embodiment design. Brunetti et al. [21] present a feature-based representation to establish the relationships among requirements, functions, working principles and geometric models. Roy et al. [22] give an object-oriented approach to help the product design passing through the complete product's life cycle from functional requirements to artifacts. Bronsvoort et al. [23] describe a multiple-view feature modeling approach for integral product design which includes conceptual design view, assembly design view, part detail design view and part manufacturing planning view. The consistency maintenance mechanism is also discussed in the work.

Some commercial CAD systems like UG, Pro/E and CATIA notice the significance of top-down product design and extend

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

their systems to support it more or less. As a representative one, Pro/E [24] develops some top-down design functions such as 2D layout, 3D skeleton model, parameter declaration and geometry feature publishing. These functions are built based on the existing parametric modeling system and strengthen the association between design knowledge to some extent. After analyzing the differences between top-down product design modules of commercial CAD systems, Aleixos et al. [25] propose a hierarchical control framework above commercial CAD systems and some design rules for integration of conceptual and detail design. Mun et al. [26,27] present 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. Lee et al. [28] present an interesting cellular-based approach to generating progressive solid models which are in various levels of detail. However, these models are generated from detailed feature-based models and are mainly used for visualization, analysis and collaboration.

Besides the works mentioned above, there are several researches focusing on the representation of the whole product in a single integrated model. Fenves et al. [29–32] present a well-defined Core Product Model (CPM) which is a base-level model not tied to any specific application or software, and is aimed at capturing product information shared throughout the whole product's lifecycle. The Open Assembly Model (OAM) [33,34] extended from CPM provides an object-oriented definition of an assembly model which incorporates representations for tolerance, kinematics, assembly relationships and assembly features. Manbub Murshed et al. [35] present Open Assembly Model Plus (OAM+) to support legacy systems engineering. The assembly feature is the key part of OAM+, and screw representation is adopted to express the relations between parts which can support kinematic analysis and force analysis well.

In general, the research on top-down product design is still preliminary. There are two problems with the works described above. The first one is the absence of an assembly model dedicated to top-down product design. Some works discover and establish significant concepts about the top-down product design, while others mainly focus on the representation of design information in some specific design phases. Although the OAM and OAM+ are quite useful assembly models for representing conventional assembly information, they are not designed to be dedicated to the top-down product design. Commercial CAD systems develop some functions for top-down design indeed, but the models and methods used are generally different and strongly dependent on their own conventions. The second problem is the insufficiency of data transferring and association mechanisms for top-down product design. The aim of the multi-level assembly model and various inheritance mechanisms presented in this paper is to solve these two problems, which account for the very obstacles of implementing general computer-based tools supporting top-down product design.

3. Top-down product design process

The product design process can be divided to five phases according to Shah et al. [36] (the division in Pahl et al. [37] is only a little different): functional design, conceptual design, embodiment design, detail design and engineering analysis as shown in Fig. 1.

However, these design phases are, in general, not sequential. Indeed, they are mostly iterative, recursive and mixed together, with no clear borderlines between different phases. Based on this observation, we refine a **top-down assembly design** process which involves the evolution of assembly structure (tree-based hierarchical model [38]) during product design as shown in Fig. 2.

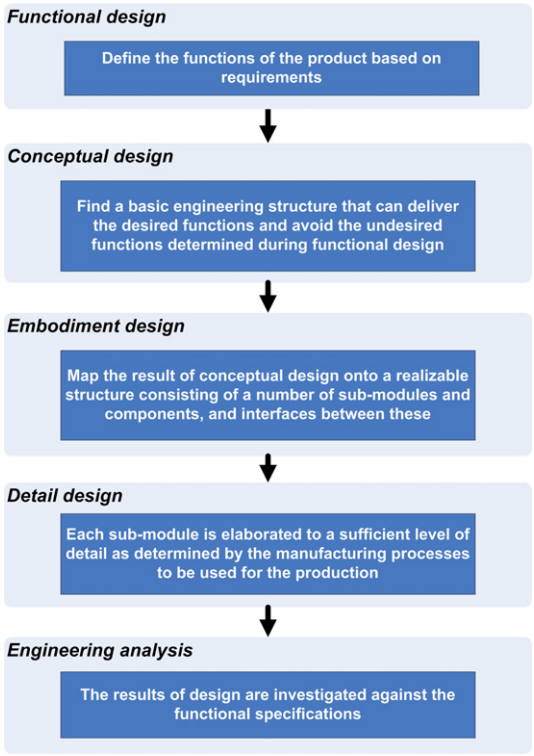

Fig. 1. Traditional product design process.

The refined top-down assembly design process is more simple and compact at fine granularity and hence more suitable for the computerization of top-down product design.

In the top-down assembly design process, an elemental subprocess **top-down component design** (denoted by **TDCD** below) is executed on each component along with the expansion of the assembly tree. When top-down assembly design starts, the TDCD is executed at the product root and the product is decomposed into several sub-components. Then loose-coupled and parallel execution of TDCD is carried out on each of the sub-components after handling most of the coupled data among them. The last step repeats recursively when some sub-components still have sub-sub-components. Finally the product design finishes when TDCD has been successfully executed on every part (leaves in the assembly tree) of the product.

Fig. 3 gives the detailed flow chart of TDCD on an arbitrary component. The key points of TDCD are explained below.

Precondition: The execution of TDCD on a component requires the predefinition of overall function and shape skeleton of the component. In other words, overall function and shape skeleton are the prerequisite of TDCD. For the product root, the overall function and shape skeleton are defined based on the requirements of the product; for components other than the product's root, the required information is generated during the TDCD of its parent assembly. Here shape skeleton is a vague and incomplete shape of a component, which is similar to the concept "design space" or "base shape" mentioned in other works [10,15,20]. It is often used like an envelope which constrains the spatial dimension and rough shape of a component.

Content: The inner behavior of TDCD is quite different according to the type of the component.

TDCD on assembly: When the component under TDCD is an assembly, abstract design and skeleton design are executed first, and then TDCD of subcomponents are launched recursively. These three steps are further described below:

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

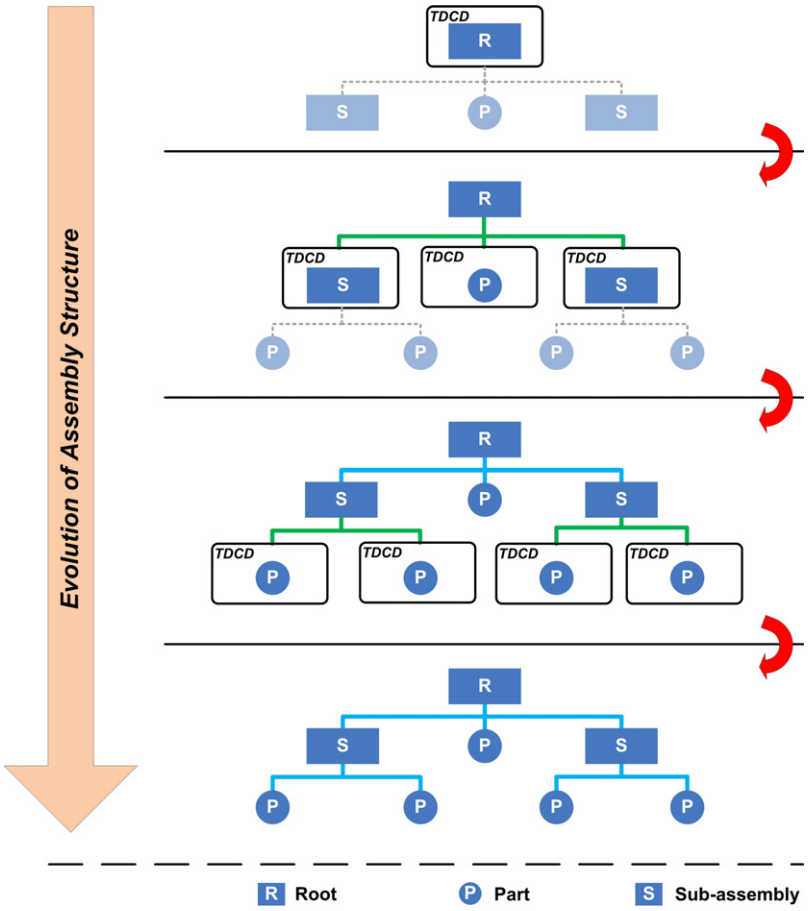

Fig. 2. Top-down assembly design along with evolution of assembly structure.

(a) The abstract design deals with abstract information such as functions, ideas and concepts which are the main responsibilities of functional design and conceptual design in traditional design process. The overall function is decomposed to establish a function structure and an engineering structure with specific physical behavior is found to deliver the desired functions. The result of abstract design is actually a "concept" (or principle solution) of the product at a specific abstract level.

(b) Based on the result of abstract design, skeleton design is carried out which mainly considers information about shape and spatial arrangement. Assembly is first decomposed into several sub-components and mapped to the design concept as a realization structure. Then the shape skeletons of these sub-components are generated within the shape skeleton of the assembly. After that, skeleton interfaces are defined between sub-components and link the corresponding shape skeletons together. Here the skeleton interface is an abstract form of assembly interface which describes the mating rules and relative motions between two or more components. Finally the spatial arrangement (relative positions of shape skeletons) is determined by specific kinematic constraints embedded in the skeleton interfaces. The result of skeleton design is a "3d-layout" which describes the spatial configuration of the assembly elements and kinematic behaviors between them.

(c) When abstract design and skeleton design finish, the TDCD should be executed on each sub-component of the assembly recursively. However, the precondition of TDCD described before needs to be prepared for the successful execution of TDCD. Fortunately, it can be seen from (a) & (b) that the functions and shape skeletons of sub-components have already been generated in abstract design and skeleton design, hence the TDCD is able to be carried out on each sub-component of the assembly recursively. In other words, the invariant needed for recursion is perfectly maintained during TDCD.

TDCD on part: On the other hand, when a component under TDCD is a part, the shape skeleton of the part is refined to the detailed shape while various design aspects such as manufacturing cost and ergonomics are considered.

It can be seen that, although it is a bit ideal as to practical situations, the top-down assembly design process can describe the product design without vague borderlines and overlapping phases. As a result, computational models and relevant algorithms which need exact data-flow information during product design can be brought in based on the top-down assembly design process.

Based on the top-down assembly design process, we list below the requirements on computer-based tools supporting top-down assembly design:

1. An integrated multi-level assembly model is needed for capturing information in different levels of abstraction during top-down assembly design. It should represent not only the detailed geometric information but also the more abstract information such as function and layout. CAD system independent property should be kept on the integrated multi-level model so that commercial CAD models can be extended and adapted to it.
2. Various inheritance mechanisms should be provided to ensure the transition and association of design information between different design phases.
3. Effective approaches for combinations of top-down design and bottom-up design should be explored to sufficiently reuse the abundant existing models.

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

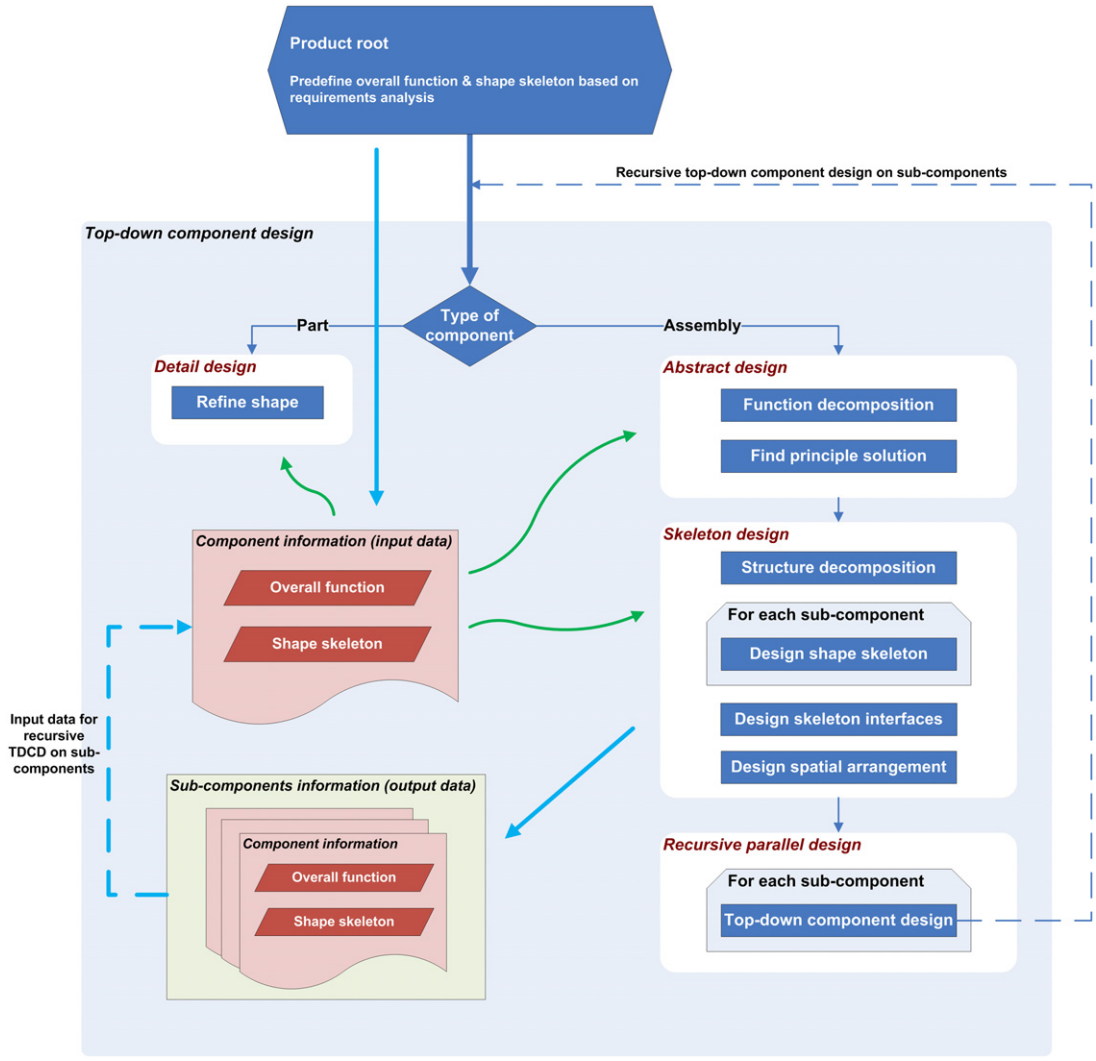

Fig. 3. Flow chart of TDCD on an arbitrary component.

4. Multi-level assembly model for top-down assembly design

Inspired by CPM [32] and OAM [34], we extend and revise the previous work [39,40] in order to support top-down assembly design better. A new multi-level assembly model is put forward here to capture multiple levels of design information, which includes the information for abstract design, skeleton design and detail design in top-down assembly design process. The abstract design mainly processes abstract information such as function and behavior, while the skeleton design processes information about the spatial arrangement of components (a.k.a. layout). Those design information is all represented in the multi-level assembly model. Meanwhile, the information for detail design such as geometry and material is also covered certainly. The whole class diagram of the multi-level assembly model is shown in Fig. 4. More details of the model are described below:

Main assembly structure

TDComponent is the abstract base class that represents a component which could be either a part or a sub-assembly in the product, and the prefix TD means top-down here. TDComponent contains the common data of an arbitrary component in a product regardless of the component type, i.e. the data shared by part and sub-assembly. *TDPart* and *TDAssembly* derived from TDComponent are the main concrete classes for controlling the product's structure and managing product data in the whole design process. TDPart represents an elementary part used in the product, which is a leaf in the assembly tree. Meanwhile, TDAssembly represents a sub-assembly (including the assembly root of the product), which is a non-leaf node and contains some child nodes under it as the components of the assembly. *AssemblyInterface* is used to represent the assembly interfaces between components of an assembly, in which kinematic relationships, connection forms and corresponding geometric-mating relationships are all managed. In addition, *CoupledRelationManager* maintains the coupled relations among components, such as algebraic equations for constraining relevant design parameters.

Abstract information

Function and *Behavior* represent information processed in abstract design. They are the common data stored in class TDComponent, which contain functional and behavioral information of a component respectively.

Function describes "what to do", i.e. the intended behavior of a component, and a reasonable function-taxonomy (e.g. [41]) could be adopted to help the description. In addition, specific function parameters and constraints are stored to give quantitative information about a component. Feature could also have functions because of the fine-granularity relationships between functional and geometric information.

Compared with function, behavior is a more objective concept which describes the way a component achieves its function, hence various behaviors could realize the same function, and

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

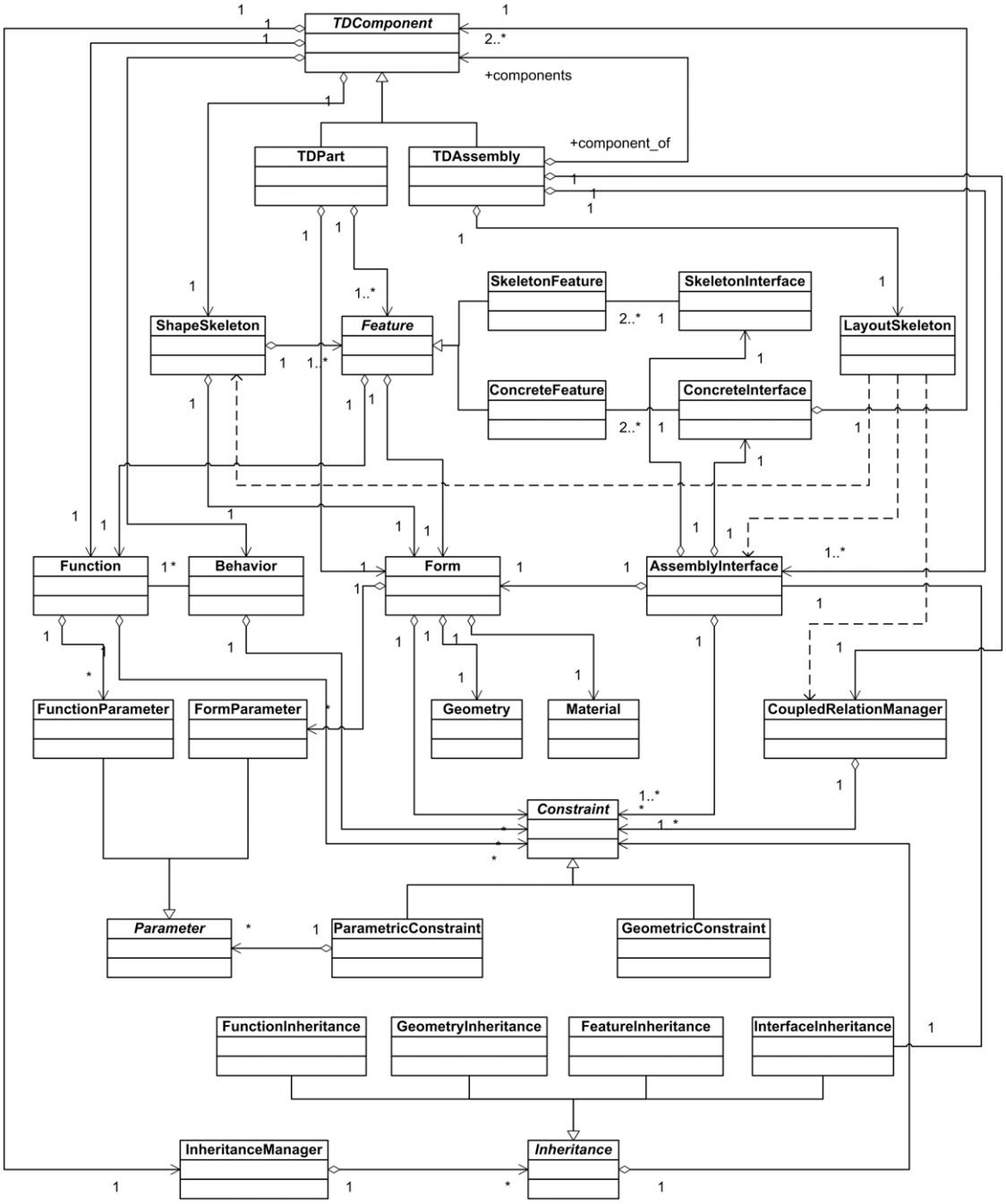

Fig. 4. The multi-level assembly model for top-down design.

the association between them is not one-to-one. On the other hand, behaviors of components can be seen as sequential state transitions along time which is determined by some specific physical phenomena and principles.

Currently, the support for abstract design is still preliminary in our model. Details about function and behavior information used in abstract design are not further explored here. Meanwhile, the connection between the abstract information and other information in the multi-level assembly model needs to be complemented.

Skeleton information

ShapeSkeleton represents the rough shape of a component which somewhat likes an envelope (see Fig. 5). It is the carrier and presenter of significant geometries and parameters derived from earlier design. ShapeSkeleton is stored in TDComponent so that each component in a product can have one. Here we formally define a shape skeleton as:

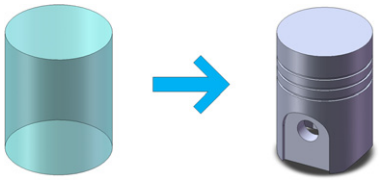

Fig. 5. The shape skeleton (left) and the final shape (right) of a piston model.

A preliminary but sufficient 3D geometric model with significant form knowledge and parameters embedded in, and the base of the following design as both the space restriction and form restriction.

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

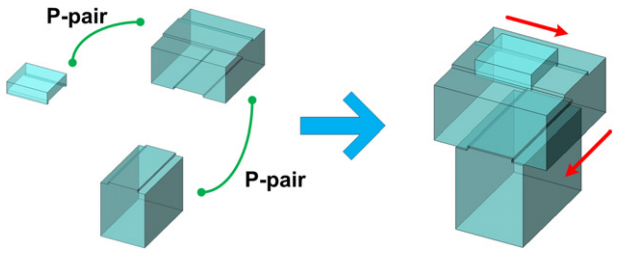

Fig. 6. A layout skeleton with three shape skeletons and two prismatic pairs.

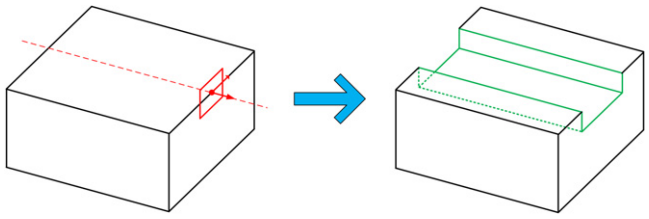

Fig. 7. The skeleton feature (left) and corresponding slot feature (right) involved in an assembly interface with prismatic pair.

LayoutSkeleton represents the fundamental layout of an assembly. Generally, the layout of an assembly consists of a number of functional components linked together through kinematic relationships. Here we define a layout skeleton as:

> *A skeleton-level assembly consisting of the child components' shape skeletons, which are arranged in three-dimensional space and connected together through the assembly interfaces defined between them.*

Fig. 6 shows an example of layout skeleton. As the layout skeleton contains the three-dimensional layout information of an assembly, preliminary kinematic analysis can be carried out in an early design phase; hence the flaws of existing design could be revealed and recovered in time, and later designs based on the layout skeleton could be executed more robustly.

The design of skeleton related information adheres to the least commitment principle. Designers do not need to consider unrelated or unimportant information of the product in design.

Detailed information

Feature is a generalized abstract class for all types of features. Each feature contains a *Form* representation which includes geometric information and material information in *Geometry* and *Material* respectively. Besides that, form also possesses a set of constraints abstracted by class *Constraint*. There are various concrete constraints such as *ParametricConstraint* and *GeometricConstraint* derived from Constraint. An extension of the constraint category could easily be integrated into the assembly model.

Multi-level assembly interface information

AssemblyInterface stores multi-level information of an assembly interface which connects two or more components together. The information in an assembly interface includes the connection type, the connection forms and the various relevant constraints.

The *SkeletonInterface* and *ConcreteInterface* represent the two implementation levels of an assembly interface. SkeletonInterface stores information about the kinematic aspect of an assembly interface, while ConcreteInterface stores information about the physical form of an assembly interface.

The *SkeletonFeature* and *ConcreteFeature* are all derived classes of abstract class Feature. A concrete feature is just the form feature used in traditional feature-based design. On the other hand, the skeleton feature contains a group of abstract geometric elements used as anchor for defining kinematic constraints (the idea comes from the interface feature in [10], and the MGDE concept in [13] shows similar thoughts behind its abstraction mechanism). The skeleton feature is positioned parametrically in the shape skeleton and can be seen as the abstraction of the concrete feature dedicated for connection. An illustration example of the skeleton feature and concrete feature in an assembly interface is shown in Fig. 7.

Normally, a skeleton interface contains kinematic constraints between two or more skeleton features, while a concrete interface contains form constraints and mating constraints between two or more concrete features. Furthermore, the skeleton features should have inner consistencies with the corresponding concrete features in an assembly interface, while the involved kinematic constraints, form constraints and mating constraints should be also consistent.

It should be pointed out that sometimes ConcreteInterface could have extra component which has no main functions but aims at the implementation of the assembly interface (like a bearing between shaft and housing). This specific relationship is also maintained in the model.

Inheritance information

Due to the existence of multiple levels of knowledge and information in the product model, it is required that the information transition and association are correctly established and maintained. *InheritanceManager* is such a manager that exists in each component to manage different kinds of information inheritance instances. The abstract class *Inheritance* is the super class of various concrete inheritance types which have specific purposes such as function inheritance and geometry inheritance. The relevant inheritance mechanisms are described in next section.

Component instancing

During the top-down assembly design, designers may frequently want to instance new components from existing ones and locate them in different positions in the assembly. Due to different design requirements, there could be various instancing strategies, and designers should be able to select the one suitable for them. The following mechanisms are provided along with the multi-level assembly model for instancing new components:

(a) Each component has a transform field which is essentially a matrix. Every time a component is instanced, a new transform is created to represent its unique position in the assembly.
(b) When designers want to instance a new component identical to an existing one throughout the design process, the *shallow copy* mode is activated. In this mode, the same contents of the original component are referenced by the newly created one concurrently. Fig. 8(a) shows how a top-down part is instanced under the shallow copy mode. All the references to the contents of the original part such as function and form are copied to the new one. Therefore, any time one of the two parts is changed, the other part is also changed accordingly since they share the same copy of contents. The shallow copy mode is suitable for the symmetrical components in the assembly.
(c) When designers want to instance a new component from an existing one and make them independent after the instancing, the *deep copy* mode is activated. In this mode, the same contents of the original component are copied first and the new component references the copied contents. Fig. 8(b) shows how a top-down part is instanced under the shallow copy mode. All the contents of the original part such as function and form are copied, and then the new part references the new copies of the contents. Therefore, the two parts could be changed independently since they do not share the same copy of contents. The deep copy mode is often used for the components with the same configuration, e.g. the feature structures, located in different assembly levels.

In all, the multi-level assembly model integrates information of different design stages together in a single extensible framework. Currently, the model puts emphasis upon the information required by the embodiment design and detail design phases of

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

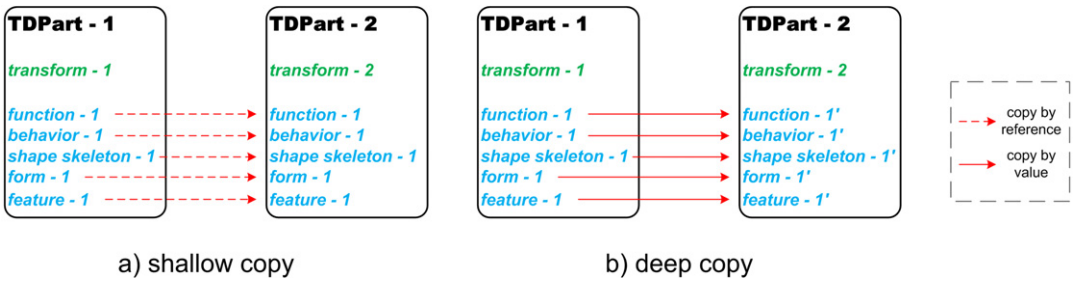

Fig. 8. The two different part-instancing strategies.

traditional top-down product design process. Information involved in these design activities such as skeleton information and detailed shape information is represented suitably. Meanwhile, the connections between the information like assembly interface, various constraints and inheritances are contained. However, to include the information involved in the abstract design better, the representation of function, behavior and the relevant connections (besides the constraints of physical laws) to other information in the multi-level assembly model should be explored further.

5. Inheritance mechanisms

In the top-down assembly design, designers need to switch among abstract design, skeleton design and detail design continuously until the detailed model of all parts in the whole product are generated. During this process, there are multiple levels of model representations involved as described before, and the design knowledge and information need to be transferred fluently and accurately between the various model representations. As a result, how to support the information flow in computer-based tool supporting top-down assembly design is a serious problem.

Inheritance definition

To transfer between various design phases during the top-down assembly design smoothly, the existing high-level design information should be utilized as both the design base and the design constraints in low-level designs. Considering that, we need here a new concept:

> *Inheritance, which normally means the process of transmission of characteristics from parents to offspring, is used here to refer to the process of transferring design information from high-level design to low-level design and establishing associations between them.*

We believe that an intelligent tool for design information inheritance during top-down assembly design is significant for designers. Using this tool, designers should be able to choose the data they are interested in or need, and the choice on the way of how these data are used should also be provided. Then the tool transfers the design information and establishes suitable associations automatically for designers. As a result, designers could use the existing design information without recreating them from scratch. Meanwhile, the inherited information could also play the role of design constraints for following design. This eases the burden of designers without loss of semantic accuracy of existing design information.

Inheritance classification

Due to the diversity of design information involved in top-down assembly design, various design information inheritances could happen. We list here four different inheritances according to the objects and data processed:

Function inheritance: Functional information in high-level design should be able to be inherited to constrain the low-level design information. For example, volume of a part could be decided by function parameters defined in abstract design, and the length, width and height of the part could be further constrained through the volume.

Geometry inheritance: Geometric information in the shape skeleton of a component should be able to be used to decide the positions or dimensions of geometric information in a detailed model of the component or shape skeletons of the sub-components.

Feature inheritance: Feature information is at a higher level than geometric information. Therefore, the feature information in a shape skeleton should be able to be inherited in whole to determine the preliminary shapes of detailed models or child shape skeletons.

Interface inheritance: The assembly interface is multi-level itself. Therefore, the skeleton interface should be able to be inherited to generate the corresponding concrete interface. Moreover, skeleton features and concrete features for assembly interfaces should be inherited either from shape skeleton to detailed model or from shape skeleton to suitable child shape skeletons.

The inheritances listed above are representative ones in top-down assembly design, and the relevant mechanisms for their implementation are described below.

Inheritance mechanisms

To support the various information inheritances in top-down assembly design, a set of robust inheritance mechanisms with diversity of intention should be explored. Intrinsically, inheritance could be regarded as the composition of specific "copy" and "association" operations (Fig. 9), in which "copy" means making a duplicate object from the original one and "association" means linking the duplicated object and the original one for further change propagation. Below we explain the details of the inheritance mechanisms for the multiple information inheritances described above.

(a) *Function inheritance*

Usually the quantitative data in function related information are some function parameters constrained by some algebraic equations, and designers choose several parameters to be inherited to restrict the following design activities. The inheritance of algebraic parameters is straightforward in which the copies of parameters are created with their values and corresponding association equations are established.

Of course, designers could constrain the design parameters in low level to parameters in high level directly without inheritance, but in this way the low level model is tightly-coupled to the high level model. In other words, when the high level model is absent, the low level model becomes invalid as the needed parameters cannot be found. This confuses and disturbs designers when they want to use the low level model alone (situation like this happens quite often). On the other hand, with the help of parameters inheritance, the low level model is loosely-coupled to the high level model because there are extra copies of parameters in the low level model which ensure the integrality (Fig. 10).

In order to thoroughly support the abstract design, more powerful function inheritance mechanism is needed which could transmit non-quantitative functional knowledge from a high-level

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

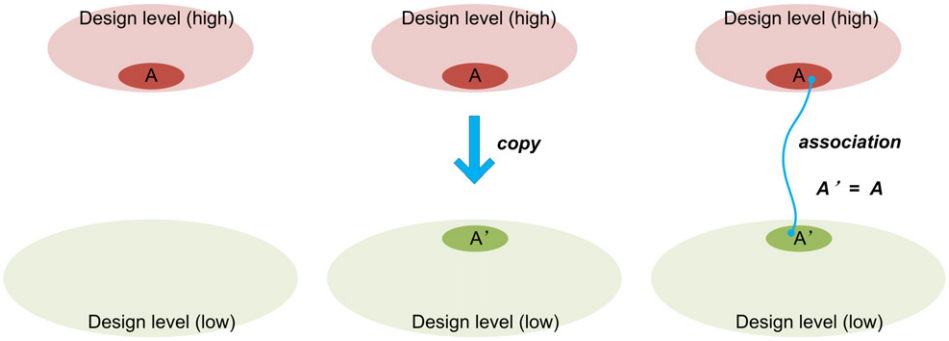

Fig. 9. The intension of inheritance.

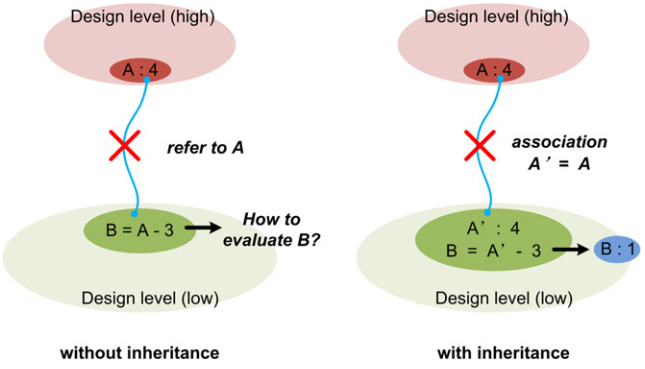

Fig. 10. The comparison of evaluation without inheritance and with inheritance.

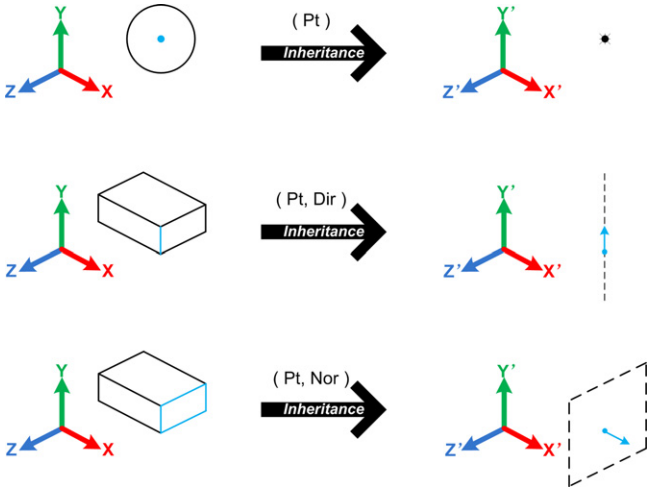

Fig. 11. The inheritance of geometric elements.

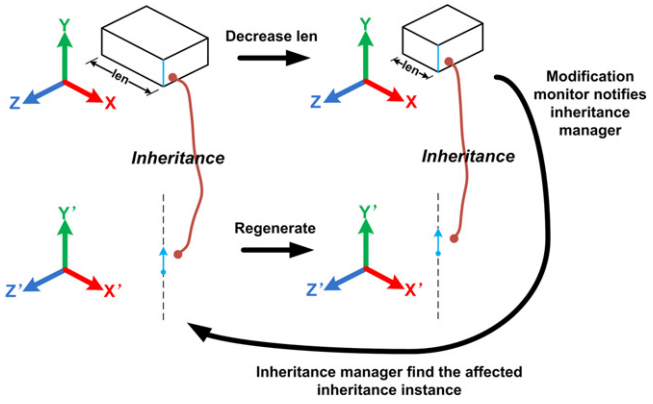

Fig. 12. The effect of inheritance under shape modification.

design to a low-level design. However, as the abstract information is not fully represented in the current multi-level assembly model, the relevant inheritance mechanism cannot be explored here and will be developed in the future.

(b) *Geometry inheritance*

The geometric elements in the shape skeleton are always needed to be referred to in defining the shapes of sub-skeletons or detailed models, while geometry inheritance mechanism works in this situation. Again, two operations "copy" and "association" are executed in sequence to inherit a geometric element.

The geometric elements to be inherited are often the ones which could be used as a datum for reference. Hence datum point, datum axis, datum plane, vertex, edge and plane are candidates for geometric inheritance. Moreover, the inherited geometric elements are all converted to datum elements, e.g. the inheritance of an edge is a datum axis. The reason is that designers would not use a single geometric element to construct the body of the shape. In other words, they just use it to position the main body of the shape. The copy of geometric elements uses the internal geometric data of source element to generate the target element (Fig. 11).

After a copy operation of a geometric element, an association relation (GEsrc, GEtar) between source and target geometric element is established. Such relations are maintained in the inheritance manager of components (see Fig. 4). Shape modification is supervised so that inheritances will be checked when any shape modification happens, and the affected inheritance will notify the target model to regenerate the inherited element according to the new source element (Fig. 12).

The geometric element inheritance mechanism makes the source model and the target model loosely coupled again, as the target model could still be valid when the association between source model and target model is cut off.

(c) *Feature inheritance*

As shape feature is used so often in design nowadays, feature inheritance mechanism must be provided with diversity of use. For a feature inheritance, we define three modes with different meanings and intentions.

Non-parametric mode: Only the shape of the source feature is copied to the target model (Fig. 13). Mostly, a designer inherits a feature purely for its shape information, and the parameters in the feature are not important to him. Actually, designer of the source feature may not want the parameters in it to be modified in following design. Therefore, the body of the source feature is copied to the target model without any parameters, and designer of the target model can refer to the geometric elements in the shape for the following design. Association between the body of source feature and the body of the target feature is established for consistency maintenance. Once the body of the source feature is

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

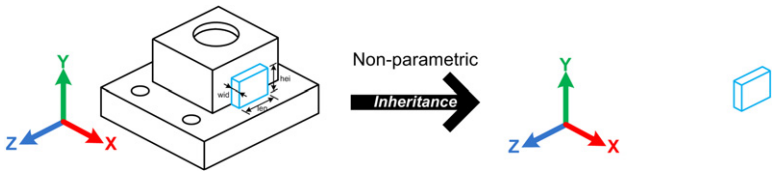

Fig. 13. Non-parametric feature inheritance.

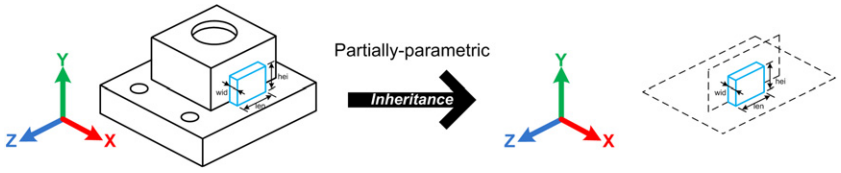

Fig. 14. Partially-parametric feature inheritance.

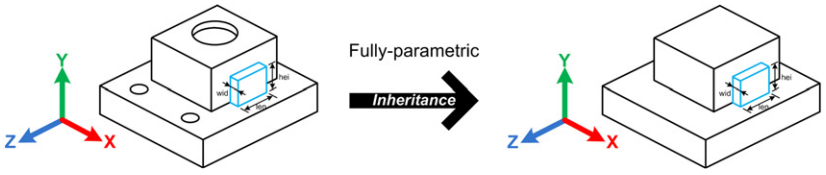

Fig. 15. Fully-parametric feature inheritance.

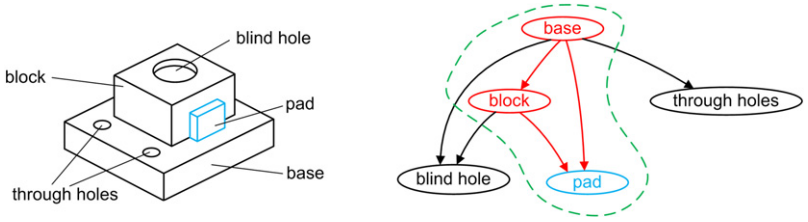

Fig. 16. The finding of the features to be inherited based on FDAG.

modified, the inheritance will regenerate the body of the target feature.

Partially-parametric mode: Not only the shape but also the parametric information of the source feature is copied to the target model (Fig. 14). When designers want to control the shape of the target feature in the absence of the source model, this mode is useful. The parametric information of a feature includes feature type, parameters and geometric references. Copying of the feature type and parameters is straightforward, while copying of the geometric references is achieved through geometry inheritance described above. Associations between the feature-parameters are established through algebraic equations, while the associations between the geometric references are established the same as the association in geometry inheritance.

Fully-parametric mode: The difference between the partially-parametric mode and the fully-parametric mode is whether the geometric references of a feature are also copied parametrically along with the feature itself. The geometric elements used as references in defining a feature F are generated by some existing features before F (in the design history); hence the modification of parameters in the existing features may change the shape or position of F. In other words, feature F depends on some features before it in the design history. If designers want to get more control on an inherited feature, not only the inherited feature but also the features on which the inherited one depends should be copied to the target model parametrically (Fig. 15). The feature dependency is often a long link since the features on which a feature depends may still depend on other features, and FDAG (feature dependency directed acyclic graph [42]) is a quite suitable representation to express it. Therefore, when a feature is selected to be inherited in the fully-parametric mode, the dependency link of the feature is extracted from the FDAG, which decides all the features that should also be inherited parametrically (Fig. 16).

(d) *Interface inheritance*

Since assembly interface has different levels of representations, multiple inheritance mechanisms should be provided for them. Once an assembly interface is defined in an assembly, skeleton features or concrete features are created on relevant components of the assembly interface. After that, if one of such components is a sub-assembly and further subdivided to sub-components, the skeleton features or concrete features on the sub-assembly should be inherited to sub-components as the existing design knowledge and constraints. Moreover, all components have their shape skeletons. Since the skeleton features or concrete features for assembly interface are always created on shape skeletons first, the features should also be inherited to the detailed model from the shape skeletons.

Skeleton feature inheritance: As described before, the skeleton feature is composed of a group of abstract geometric elements used as an anchor for defining kinematic constraints. Therefore, the inheritance of skeleton features is actually the inheritance of these geometric elements (Fig. 17), while the geometry inheritance mechanisms described above could help.

Concrete feature inheritance: A concrete feature for assembly interface is the physical form in a component model for connection with other components. Hence the feature inheritance

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

Fig. 17. The inheritance of skeleton feature.

Fig. 18. The inheritance of concrete feature.

mechanisms described above is the natural way to inherit the concrete feature for assembly interface (Fig. 18).

Skeleton feature to concrete feature inheritance: In an assembly interface, the skeleton feature is not exactly the same as the concrete feature but the abstract form of it, hence the inheritance described here is actually an implicit inheritance. Concrete features should be generated according to the already designed skeleton features. We define a common skeleton feature which consists of a plane, a point, a normal axis and an extra orientation axis in the plane. The predefined skeleton feature could be used as the geometric anchor for definition of six lower-kinematic pairs, i.e. prismatic pair, revolute pair, cylindrical pair, screw pair, planar pair and spherical pair. Different geometric elements in the common skeleton are activated for different types of kinematic-pair definitions, while the inactivated elements are usually used for determining positions (Fig. 19). The definition of higher kinematic-pairs is much more difficult and has unlimited types theoretically, hence it will not be discussed here.

In fact, inheritance from skeleton feature to concrete feature depends on the type of concrete interface defined to implement the corresponding skeleton interface. The same skeleton interface could have several different types of concrete interface, due to the fact that the same kinematic relation could have different physical forms to implement it (e.g. the two different forms in Fig. 20 for prismatic pair). In order to inherit skeleton feature to concrete feature, the internal knowledge of different skeleton features and concrete features could be used to establish different rules for generating concrete features based on specific skeleton features. For example, Fig. 21 shows the generation of concrete features in a pin–hole interface, in which the positions and orientations of pin and hole are determined by the skeleton features, while the common parameters (diameter and length here) are defined in the concrete interface for ensuring consistency between pin and hole.

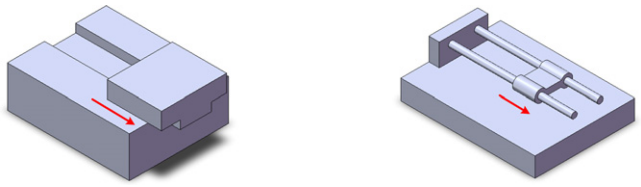

Fig. 20. Two different implementations of prismatic pair.

6. Top-down assembly design sample

Here a top-down assembly design sample is used to illustrate the usefulness of the multi-level assembly model and the inheritance mechanisms. In this sample an engine for an automobile is designed from the overall function and shape skeleton to the most detailed geometry. There are six components in the engine assembly, which includes one shell part, two rod sub-assemblies, two piston parts and one crankshaft. Moreover, the rod sub-assembly is composed of the rod body and the end cap.

Fig. 22 shows a brief evolution process for the engine design, which does not give the function related information for clarity. It can be seen that in the design process, engine evolves from the shape skeleton to the layout skeleton which consists of shape skeletons of components in the engine, and then the components of the engine continue the evolution from their shape skeletons. In the components, the shell, piston and crankshaft are all parts and the detailed models are generated from their shape skeletons. On the other hand, the rod is a sub-assembly hence the layout skeleton of the rod is generated from the shape skeleton of rod. After that, recursive design is carried out on the rod body and end cap respectively. Finally, the design finishes when detailed models of the rod body and end cap are generated from their shape skeletons. The detailed model of an assembly is easy to get by assembling the detailed models of components in the assembly.

The assembly structure and relevant model information is shown in Fig. 23. It can be seen that each component has its own shape skeleton. Besides that, each assembly has an extra layout skeleton controlling the spatial arrangement the subcomponents' shape skeletons and the assembly interfaces between them. Skeleton interfaces between shape skeletons in the layout skeletons of engine model are shown in Fig. 24, in which the concrete features for connection are already added on to the shape skeletons based on the skeleton features.

The inheritance mechanisms are used throughout the engine design (Fig. 25). The shape skeleton of the crankshaft uses two faces in the overall shape skeleton to define the start and end of the shaft, hence two reference planes inherited from the two selected faces

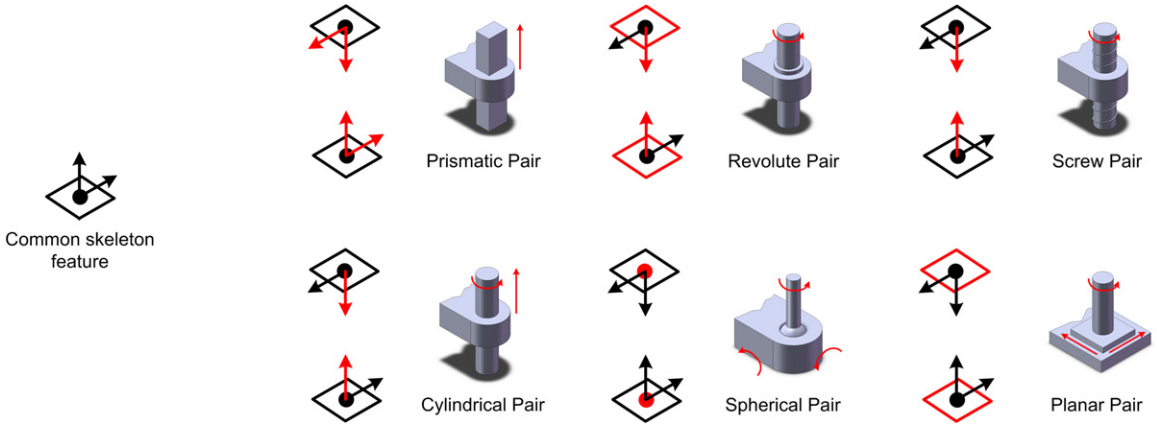

Fig. 19. Skeleton feature for lower-kinematic pairs (activated elements are shown in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

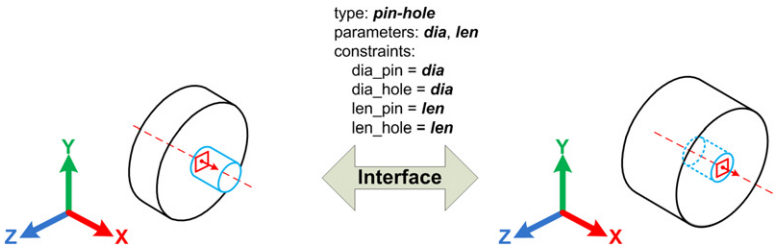

Fig. 21. Generation of concrete features in a pin–hole interface.

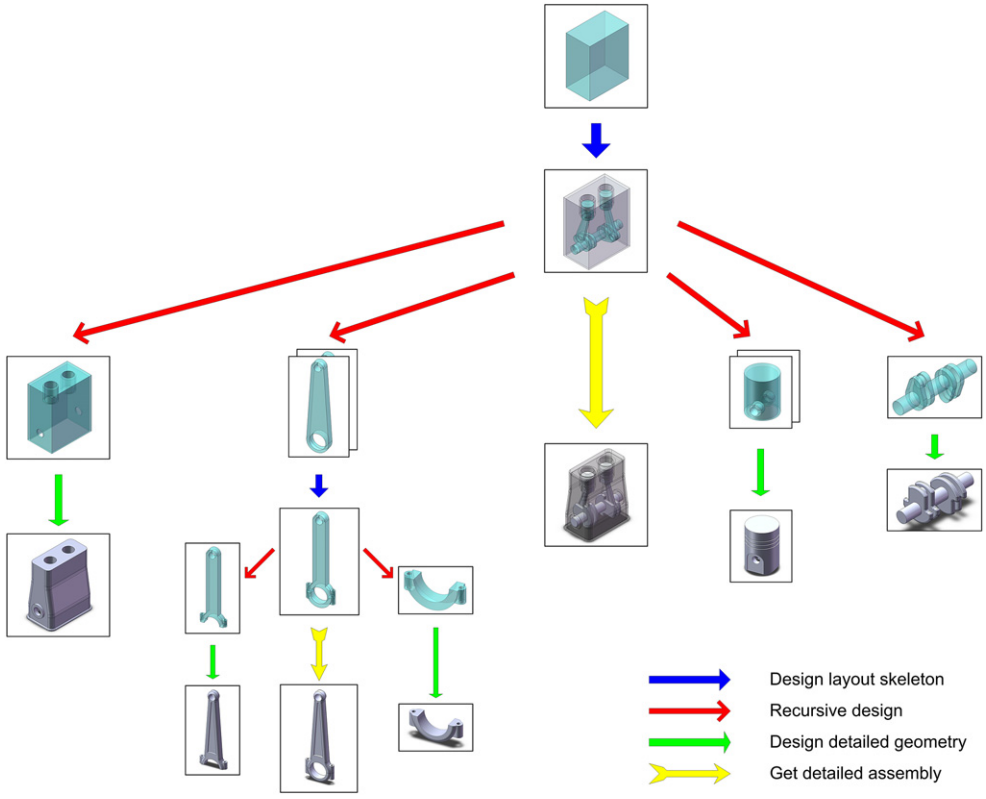

Fig. 22. Top-down assembly design process of engine.

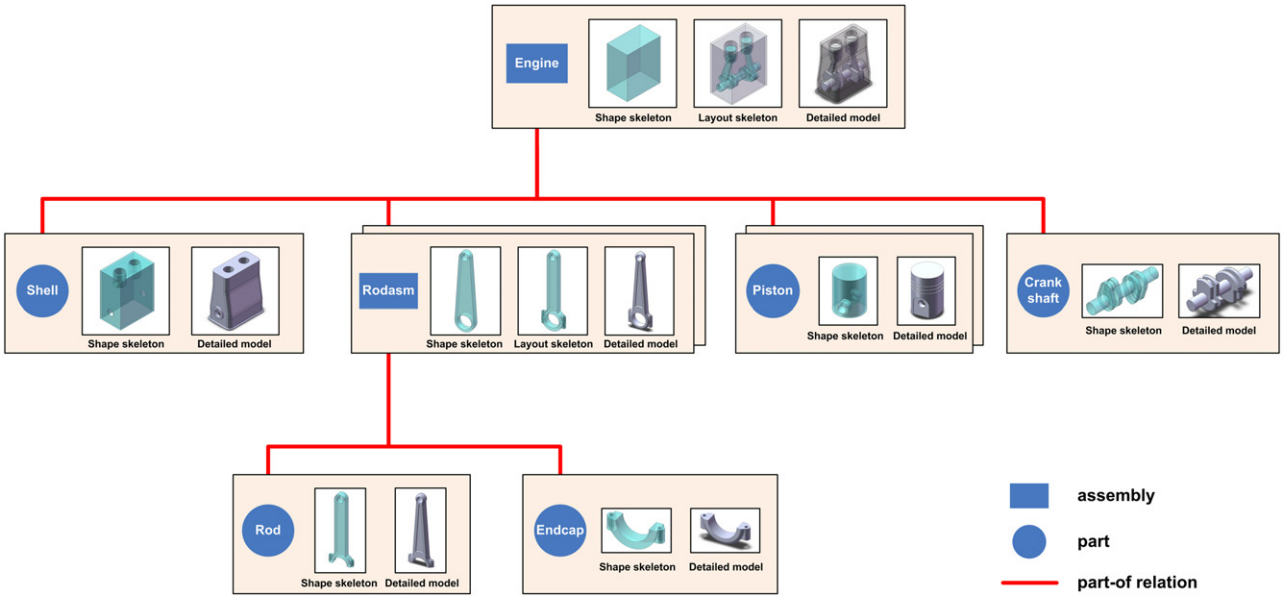

Fig. 23. The multi-level assembly model of engine.

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

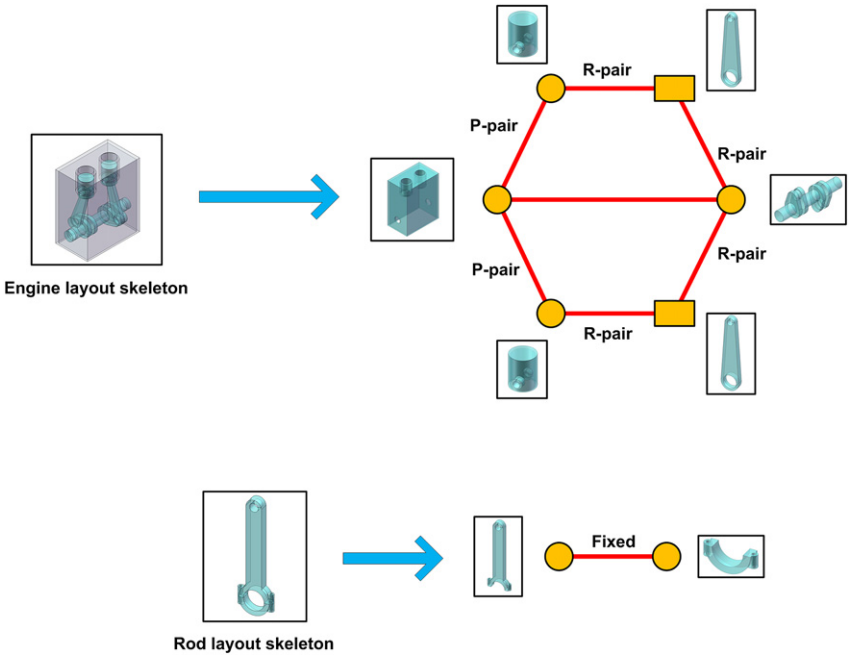

Fig. 24. Skeleton interfaces in the layout skeletons of engine model.

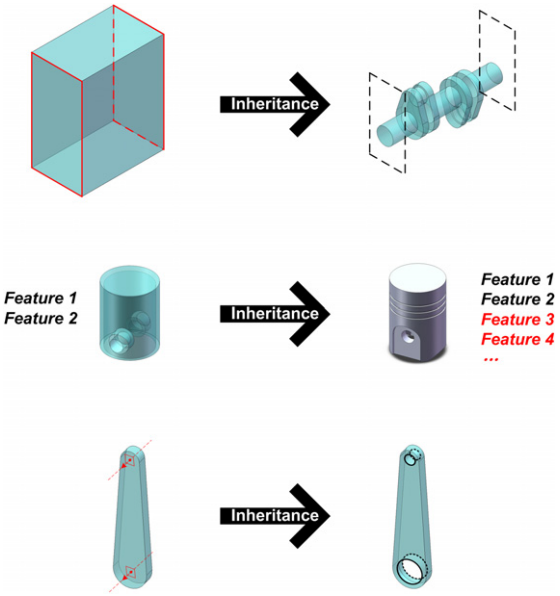

Fig. 25. Some inheritances in engine design.

are generated in the crankshaft model (geometry inheritance). The design of the piston inherits the features of shape skeleton and adds new features to generate the detailed model of the piston (feature inheritance). During the design of the rod, there are two skeleton features defined on the shape skeleton of it. One is for the revolute-pair between rod and piston, and the other is for the revolute-pair between rod and crankshaft. Then at a suitable time designers can inherit these two skeleton features to generate the concrete holes on the shape skeleton (interface inheritance).

When the top-down assembly design of engine finishes, the layout skeleton can be used as the main controller for the engine assembly, i.e. the parameters defined in it can be modified according to specific needs. The multi-level assembly model and the inheritance mechanisms used in the engine design ensure that the engine model changes correctly when the layout skeleton is modified (Fig. 26).

Moreover, the multi-level assembly model has another benefit that deserves attention. The multi-resolution representation of an assembly could be easily generated based on the multi-level assembly model. As each component in the assembly has its own shape skeleton, the multi-resolution assembly model can be generated through the combination of shape skeletons and detailed models (Fig. 27).

7. Discussion

In order to make next generation computer-based tools supporting top-down assembly design more useful and receivable, some practical problems need to be considered. One of them is the adaptation and extension for existing CAD systems. Since current CAD systems have good support for detailed geometric modeling, how to adapt and extend the abilities of the existing CAD systems to build the top-down assembly design tool, rather than to create brand new systems, is quite meaningful.

In the work, the presented multi-level assembly model is essentially a neutral high-level framework built on top of some common infrastructures of today's mainstream CAD systems such as solid models, features and constraints. It has no specific requirements on these infrastructures although there are different implementations of them. Therefore the presented multi-level assembly model can be seen as independent of any CAD systems which support feature-based modeling well, and native models of these existing CAD systems could be adapted and extended to construct a top-down assembly design environment based on the presented model.

Fig. 28 displays the adaptation and extension for native models of existing CAD systems. In the picture, the main classes in the multi-level assembly model are shown and classified into two categories, one is the part that is not hard to be adapted and the other is the part that needs extension.

The classes which could be generally adapted from native model of CAD systems are: TDComponent, Form, ShapeSkeleton, ConcreteFeature and Constraint. TDComponent could be adapted from a native assembly as a host assembly, which aggregates the shape skeleton and other more detailed models. Form could be supported well by the traditional B-rep model and the material

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

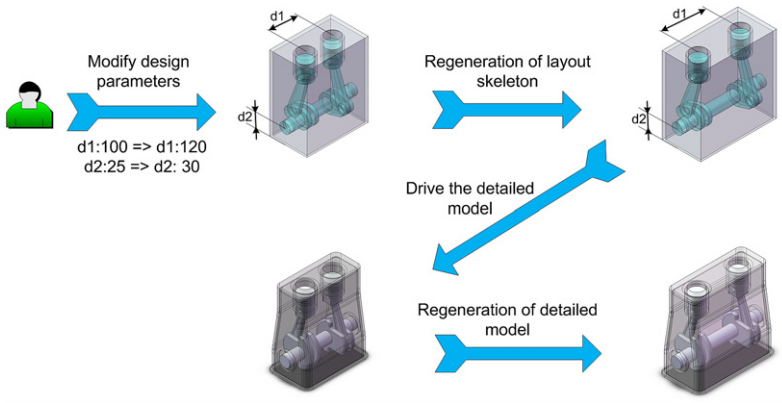

Fig. 26. The change propagation on modification of parameters in the layout skeleton.

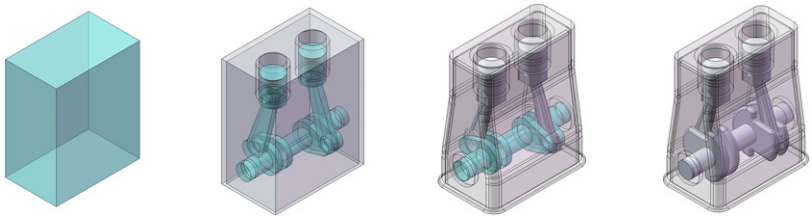

Fig. 27. Some multi-resolution models of engine.

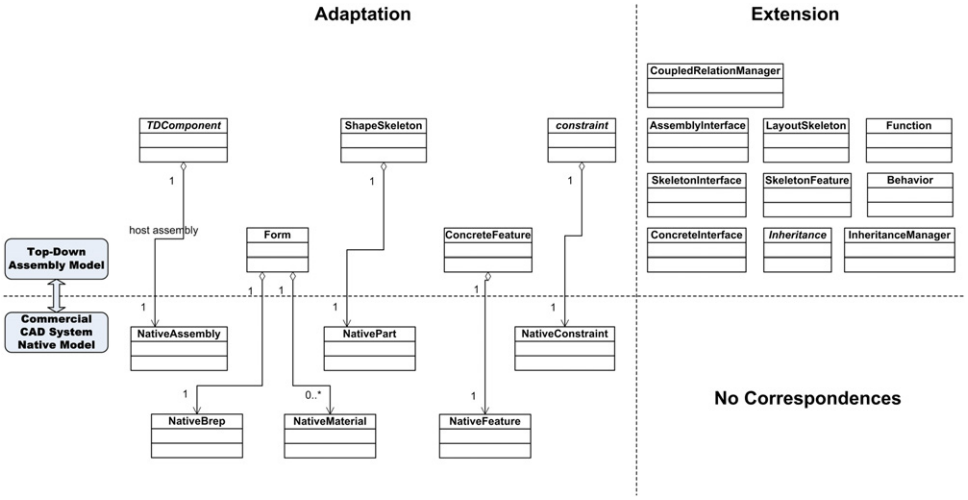

Fig. 28. The adaptation and extension for native models of existing CAD systems.

information of an existing CAD system. ShapeSkeleton is intrinsically a simplified shape model, hence could be adapted from NativePart easily. ConcreteFeature is the same as traditional form feature which is used in most existing CAD systems. Constraint is an abstract class for concrete constraints such as parametric constraint and geometric constraint which could be supported well in existing CAD systems. However, there are still some complex constraints such as a non-linear equation system which cannot be supported in some CAD systems. Therefore the adaptation for constraint is still partial.

On the other hand, the classes which have no direct correspondences in existing CAD systems are: AssemblyInterface, SkeletonInterface, ConcreteInterface, LayoutSkeleton, SkeletonFeature, Inheritance, InheritanceManager, CoupledRelationManager, Function and Behavior. The limited support to the listed information is obviously an obstacle. Existing CAD systems need extensions for the unsupported information to construct effective top-down assembly design modules. Once the extensions succeed, even the computer-based environment for collaborative top-down assembly design between heterogeneous CAD systems is likely to be created, which could provide more powerful support to complex product design.

8. Conclusion and future works

In this paper, a multi-level assembly model for top-down assembly design is presented. Differing from existing assembly models, it captures abstract information, skeleton information and detailed information that belongs to different design phases, and thus can effectively support top-down assembly design. As the base for constructing the multi-level assembly model, the traditional product design process is a bit too coarse and has no clear borderlines between design phases. Therefore, a top-down assembly design process is refined from the traditional product design process. The top-down assembly design process consists of many fine-granularity TDCD (“top-down component design”)

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

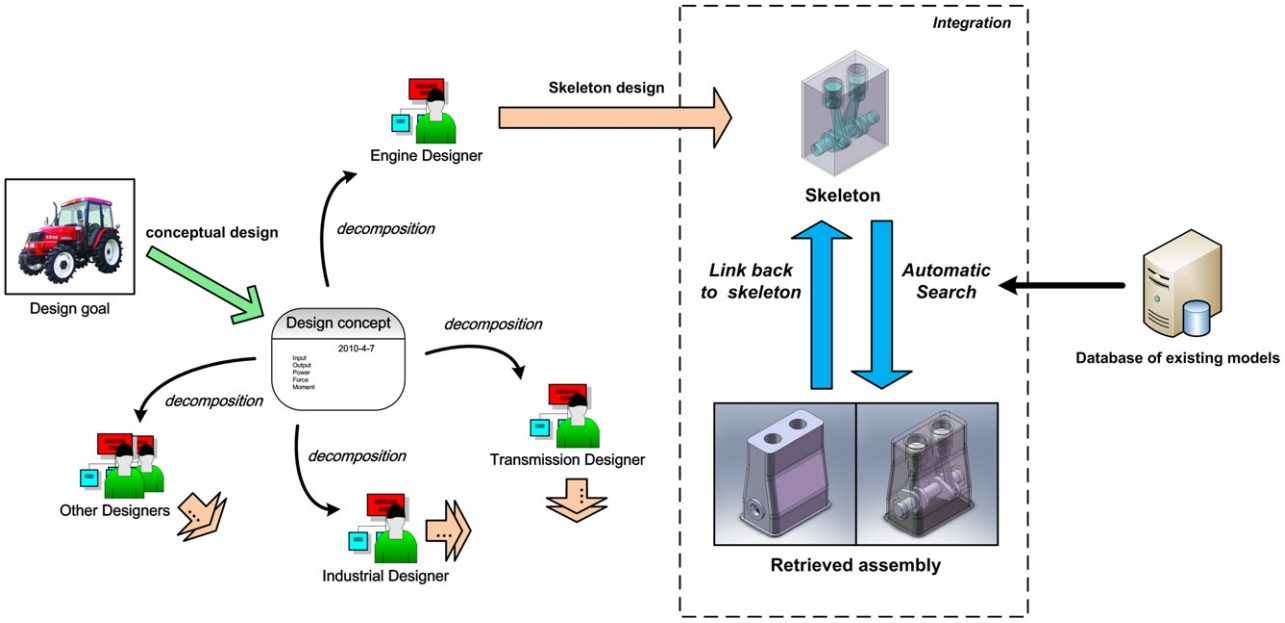

Fig. 29. The framework for integration of top-down design and bottom-up design.

design steps which are executed recursively along with the expansion of an assembly tree structure. In our opinion, the top-down assembly design process could depict the product design process more simply and compactly at the fine-granularity level which eases the computerization of top-down assembly design. Furthermore, various inheritance mechanisms for transferring and converting information pertaining to different design phases are also classified and described respectively. Function inheritance, geometry inheritance, feature inheritance and interface inheritance are the main items of the classification, each with its own specialties. These inheritance mechanisms could help to propagate design knowledge for designers and ensure design consistencies between different design phases. Practical uses of top-down assembly design are discussed for extending the scope and deepness of its application. The adaptation and extension for native models of existing systems is helpful for the rapid development of top-down assembly design modules on mature CAD systems.

In the future, the multi-level assembly model with inheritance mechanisms will be implemented based on an existing CAD system to check its validity in real complex design tasks. Besides that, great efforts are needed to address the following issues, which could bring vast benefits to next generation CAD systems supporting top-down assembly design:

Complementing the abstract information: The abstract information in current top-down assembly models is still preliminary and does not contain all the information involved in functional design or conceptual design. Therefore, the multi-level assembly model needs further expansion in order to effectively support the above two design phases, e.g. incorporating the Function-Behavior-State (FBS) model. Furthermore, the connections and smooth transitions between the abstract information and other information in the top-down assembly model are required. Meanwhile, the relevant inheritance mechanisms for abstract information should also be explored.

Incorporating other design paradigms: Since today's design is dynamic and incremental, product requirements always change during the design process, as well as the product solutions which fulfill the product's requirements. Several works have addressed these problems in various aspects, such as the works by Zeng et al. [43,44] which give the set-theory based mathematical formulation of design process and describe the design specification, product description and product performances involved in the dynamic design processes, the work by Otto et al. [45] which discusses the reverse engineering and redesign problem and the work by Zeng et al. [46] which establishes the mathematical foundation of the freehand design sketches evolving in conceptual design. The multi-level assembly model presented in this paper should be enhanced with more assistant design-variation mechanisms to support the incremental character of design process. For example, the mechanisms for adjusting a product's structure are needed, which could combine several parts to a sub-assembly or split a sub-assembly into several parts during the top-down design process, while the relevant skeletons, constraints and inheritances are reasonably maintained. Besides the incremental design, our model should also consider and incorporate the characteristics of some other design paradigms, such as design in context and mechatronical design.

Combining the top-down and bottom-up design: Although top-down design and bottom-up design are two distinct ways for constructing products, in many real-world designs of complex products, they are used together. Therefore, the methods for smooth integration of the two design approaches are needed to take full advantages of both. A demonstrative framework for the integration of top-down design and bottom-up design is shown in Fig. 29. The main ideas behind are skeleton-based assembly retrieval and linking: (a) during the design process, the skeleton which controls the product layout and component shapes is a quite suitable query for searching assemblies. The skeleton model consists of key components with important design parameters and the relations between them. In fact, the skeleton model is the well-defined common result at some milestones in the whole design process and plays the role of leading the following design activities; (b) after assembly retrieval, the returned models need to be filtered and adjusted in order to satisfy design constraints and specifications the designers require. Therefore, methods and tools for establishing the link between the query skeleton and the retrieved assembly should be provided, which make the high level control of the assembly through the skeleton possible. Based on assembly retrieval and skeleton linking, innovative design and model reuse could be seamlessly integrated, which is actually a way for achieving the integration of top-down design and bottom-up design.

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008

Acknowledgements

The authors are very grateful for the financial support from NSF of China (No. 60736019).

References

[1] Libardi E, Dixon J, Simmons M. Computer environments for the design of mechanical assemblies: a research review. Engineering with Computers 1988; 3:121–36.
[2] Wen Jian L, Tian guo J. Research state and development directions of product top-down design. Computer Integrated Manufacturing Systems 2002.
[3] Mäntylä M. A modeling system for top-down design of assembled products. IBM Journal of Research and Development 1990;34:636–59.
[4] Sturges R, O'Shaughnessy K, Reed R. A systematic approach to conceptual design. Concurrent Engineering 1993;1:93.
[5] Umeda Y, Ishii M, Yoshioka M, Shimomura Y, Tomiyama T. Supporting conceptual design based on the function-behavior-state modeler. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 1996;10: 275–88.
[6] Karnopp D, Margolis D, Rosenberg R. System dynamics: a unified approach. John Wiley & Sons; 1990.
[7] Gui J-K, Mäntylä M. Functional understanding of assembly modelling. Computer-Aided Design 1994;26:435–51.
[8] Wang L, Shen W, Xie H, Neelamkavil J, Pardasani A. Collaborative conceptual design — state of the art and future trends. Computer-Aided Design 2002;34: 981–96.
[9] Lashin G, Feldhusen J. A CAD-based tool for development of large layouts. Research in Engineering Design 1996;8:217–28.
[10] Csabai A, Stroud I, Xirouchakis PC. Container spaces and functional features for top-down 3D layout design. Computer-Aided Design 2002;34:1011–35.
[11] Csabai A, Taiber J, Xirouchakis P. Design support using constraint-driven design spaces. In: Br uderlin B, Roller D, editors. Geometric constraint solving and applications. Springer-Verlag; 1998. p. 82–106.
[12] Mantripragada R, Whitney D. The datum flow chain: a systematic approach to assembly design and modeling. Research in Engineering Design 1998;10: 150–65.
[13] Clement A, Desrochers A, Riviere A. Theory and practice of 3D tolerancing for assembly. In: CIRP seminar on computer aided tolerancing. USA: Penn State University; 1991.
[14] Shah J, Rogers M. Assembly modeling as an extension of feature-based design. Research in Engineering Design 1993;5:218–37.
[15] Holland W, Bronsvoort W. Assembly features in modeling and planning. Robotics and Computer Integrated Manufacturing 2000;16:277–94.
[16] Van Holland W, Bronsvoort W, Jansen F. Feature modelling for assembly. Graphics and Robotics 1993;131–48.
[17] Shyamsundar N, Gadh R. Internet-based collaborative product design with assembly features and virtual design spaces. Computer-Aided Design 2001; 33:637–51.
[18] Singh P, Bettig B. Port-compatibility and connectability based assembly design. Journal of Computing and Information Science in Engineering 2004;4:197.
[19] Kim K, Manley D, Yang H. Ontology-based assembly design and information sharing for collaborative product development. Computer-Aided Design 2006; 38:1233–50.
[20] Kusiak A, Szczerbicki E. Transformation from conceptual to embodiment design. IIE transactions 1993;25:6–12.
[21] Brunetti G, Golob B. A feature-based approach towards an integrated product model including conceptual design information. Computer-Aided Design 2000;32:877–87.
[22] Roy U, Pramanik N, Sudarsan R, Sriram R, Lyons K. Function-to-form mapping: model, representation and applications in design synthesis. Computer-Aided Design 2001;33:699–719.
[23] Bronsvoort W, Noort A. Multiple-view feature modelling for integral product development. Computer-Aided Design 2004;36:929–46.
[24] PTC, Pro/engineer advanced top-down design. In: Release 2000i2, T823-310-01.
[25] Aleixos N, Company P, Contero M. Integrated modeling with top-down approach in subsidiary industries. Computers in Industry 2004;97–116.
[26] Hwang J, Mun D, Han S. Representation and propagation of engineering change information in collaborative product development using a neutral reference model. Concurrent Engineering 2009;17:147.
[27] Mun D, Hwang J, Han S. Protection of intellectual property based on a skeleton model in product design collaboration. Computer-Aided Design 2009;41: 641–8.
[28] Lee J, Lee J, Kim H, Kim H. A cellular topology-based approach to generating progressive solid models from feature-centric models. Computer-Aided Design 2004;36:217–29.
[29] Fenves S. A core product model for representing design information, US National Institute of Standards and Technology Internal Report, 6736; 2002.
[30] Fenves S, Foufou S, Bock C, Bouillon N, Sriram R. CPM2: A revised core product model for representing design information. National Institute of Standards and Technology, NISTIR, 7185; 2004.
[31] Sudarsan R, Fenves S, Sriram R, Wang F. A product information modeling framework for product lifecycle management. Computer-Aided Design 2005; 37:1399–411.
[32] Fenves SJ, Foufou S, Bock C, Sriram RD. CPM2: a Core Model for Product Data. Journal of Computing and Information Science in Engineering 2008;8: 014501–6.
[33] Rachuri S, Baysal M, Roy U, Foufou S, Bock C, Fenves S, et al. Information models for product representation: core and assembly models. International Journal of Product Development 2005;2:207–35.
[34] Rachuri S, Han Y-H, Foufou S, Feng SC, Roy U, Wang F, et al. A model for capturing product assembly information. Journal of Computing and Information Science in Engineering 2006;6:11–21.
[35] Manbub Murshed SM, Shah JJ, Jagasivamani V, Wasfy A, Hislop DW. OAM+: an assembly data model for legacy systems engineering. In: ASME conference proceedings, american society of mechanical engineers. 2008. p. 869–81.
[36] Shah J, Mäntylä M. Parametric and feature-based CAD/CAM: concepts, techniques, and applications. Wiley-Interscience; 1995.
[37] Pahl G, Beitz W, Wallace K, Blessing L, Bauert F. Engineering design: a systematic approach. Springer Verlag; 1996.
[38] Lee K, Gossard DC. A hierarchical data structure for representing assemblies: part 1. Computer-Aided Design 1985;17:15–9.
[39] Chen X, Gao S, Yang Y, Zhang S. The skeleton in the multi-level assembly model for top-down innovation design of mechanical product. In: 2009 international conference on product lifecycle management. 2009.
[40] Zhang ST, Chen X, Gao SM, Yang YD. A framework for collaborative top-down assembly design. In: Proceedings of the asme international design engineering technical conferences and computers and information in engineering conference 2007, Vol 6, Pts a and B, 2008. p. 139–49.
[41] Szykman S, Racz J, Sriram R. The representation of function in computer-based design. In: Citeseer. 1999.
[42] Li M, Zhang YF, Fuh JYH, Qiu ZM. Toward effective mechanical design reuse: CAD model retrieval based on general and partial shapes. Journal of Mechanical Design 2009;131:8.
[43] Zeng Y, Gu P. A science-based approach to product design theory part I: formulation and formalization of design process. Robotics and Computer Integrated Manufacturing 1999;15:331–9.
[44] Zeng Y, Gu P. A science-based approach to product design theory part II: formulation of design requirements and products. Robotics and Computer Integrated Manufacturing 1999;15:341–52.
[45] Otto K, Wood K. Product evolution: a reverse engineering and redesign methodology. Research in Engineering Design 1998;10:226–43.
[46] Zeng Y, Pardasani A, Dickinson J, Li Z, Antunes H, Gupta V, et al. Mathematical foundation for modeling conceptual design sketches. Journal of Computing and Information Science in Engineering 2004;4:150.

Please cite this article in press as: Chen X, et al. Multi-level assembly model for top-down design of mechanical products. Computer-Aided Design (2011), doi:10.1016/j.cad.2010.12.008