The skeleton in the multi-level assembly model for top-down innovation design of mechanical product

Abstract: This paper presents an improved engineering model extended from our previous work to better support top-down product design. The main characteristics of our new model are: (a) several vital kinds of engineering data for top-down design are captured effectively; (b) the top-down design mode is explicitly supported, i.e., a real-embedment of top-down design in CAD system can be achieved based on this model; (c) as it's platform-independent (here the platform means CAD systems), adaptation from existing CAD systems won't be an obstacle. Finally, we also implement a prototype system to prove the correctness and feasibility of our model's new abilities, and optimistic results are obtained.

Keyword: top-down assembly design; skeleton model; inheritance; multi-resolution model.

1 Introduction

Modern engineering products expand rapidly in both scale and complexity. Hence how to design them effectively and efficiently is of great significance. Indeed, an intelligent engineer may be able to design a good product of reasonable scale in a *"plain"* way. This means that a genius could consider all aspects of the product simultaneously; no matter how important or trivial they are, he just accomplishes them all at once. Nonetheless, large and complex product accomplished in this way by just one engineer is apparently an unpractical vision. The design work should be divided into reasonable segments which are accomplished from the very vital to the trivial and from the rough to the refined. And furthermore, cooperation is the key to success in each kind of industry. Actually, cooperation among experts and engineers in different regions becomes more and more important nowadays. As a result, the *"top-down"* design is exactly the mode which can take account of both work division and cooperation naturally.

Unfortunately, with the limited support of today's CAD commercial software to the top-down assembly design, there is still too much work that cannot be powered up by computers rather than finished manually. 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. Our work here is meant to change the situation and improve the design efficiency.

We present a model which captures some useful data and knowledge during top-down assembly design. This model is platform independent and theoretically every commercial CAD system's native model which supports feature-based modeling could be adapted to our model thereby the top-down assembly design could be carried out. And finally, with this model we can do some other interesting and practical jobs incrementally.

The rest of the paper is organized as follows. Section 2 introduces some previous studies related to this paper. Section 3 gives a brief introduction to our extended model. Section 4 describes the skeleton and skeleton architecture. Section 5 illustrates the inheritance mechanism. Section 6 discusses the adaptation and extension for the native model of commercial CAD systems. Section 7 explores some extra applications of the skeleton model. And at last, the implementation and conclusion will be given.

2 Related works

There are several interesting works related to our work on top-down assembly design approach presented in this paper.

Mäntylä [1-2] presented many important concepts and issues about top-down design approach in mechanical engineering, such as abstract geometry, focus change, geometry inheritance among different assembly design stages, redesign problem and so on. A simple 2D prototype system is also implemented. However, the authors mainly focused on the discovery and establishment of significant concepts when there are little details about concrete models and methods about top-down assembly design in these works.

After the recognition of the importance of conceptual design and other early design stages, studies [3-5] mainly concentrated on the linking of the early conceptual design and the other design stages that follow.

Studies [6-7] presented a well-defined family of product models which support the whole range of product life cycle. The NIST Core Product Model (CPM) is the key part, and the Open Assembly Model (OAM), the Design-Analysis Integration model (DAIM) and the Product Family Evolution Model (PFEM) are the extensions. However, there are no considerations for explicit supports on top-down assembly design approach in these works.

Bidarra [8] presented a design environment that supports integration of the part and assembly design. Users can switch conveniently between the single component design and the assembly relations design in this environment. But similarly the top-down assembly design approach is not supported directly in it.

Shyamsundar [9] introduced the virtual space concept and presented a geometric representation AREP for collaborative assembly design. It has two expressions which correspond to server side and client side respectively. However, the real top-down assembly design process is not considered in that work.

Commercial CAD system Pro/Engineer [10-11] has developed some top-down design functions such as the skeleton model (which differs from our skeleton model) and declaration method. These functions are practical but the support to top-down assembly design process is still limited.

Jinsang [12] described the top-down product design process among OEM and suppliers, and distributed the common skeleton model of Pro/Engineer and CATIA to secure the enterprise intellectual property during collaboration. This work is mainly applied to the asynchronous collaborative assembly design among OEM and suppliers, hence the top-down assembly design with real-time synchronous collaboration carried on among the design group members of a department is not considered.

3 The multi-level assembly model overview

Considering the situation that the top-down assembly design approach is so important in mechanical engineering but not supported well enough, we have presented a framework for this in our previous work [13]. And in this paper we present a new model based on our previous top-down assembly model and the OAM [6] to support top-down design approach better. Further, this model could be applied to the real-time synchronous collaboration design easily. But we will not discuss more about collaboration here; the top-down product design itself is our main topic.

3.1 The top-down assembly design process

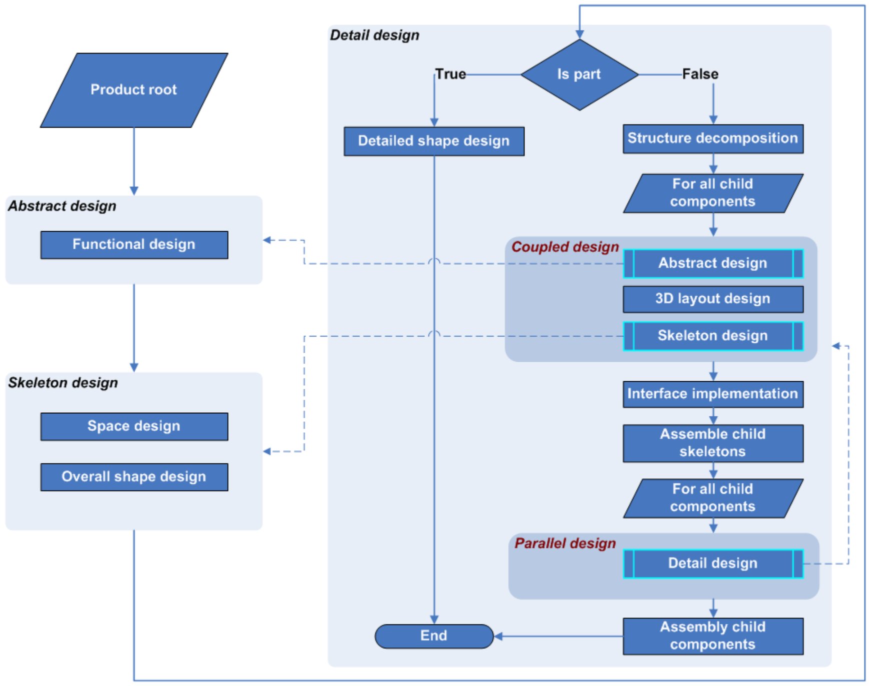
This model is based on the top-down assembly design process which is divided into three phases, i.e. the abstract design, the skeleton design and the detail design. Notably, these three design phases do not happen just once for the product. Instead, they are carried out recursively on each node (it can be either a part or an assembly) in the whole assembly tree of the product. In other words, every component in the product goes through these three design phases. Now for a component under design, a brief description about the design tasks in these phases for this component is given below.

1. *The abstract design phase*
   The designer mainly considers the function of this component in this phase. The component specification is generated as the design result of this design phase.
2. *The skeleton design phase*
   The designer begins to consider the whole shape and the design space of this component based on the functions determined in last design phase. The skeleton model which will be discussed more in section 4 is generated as the design result of this design phase.
3. *The detail design phase*
   Since this component could be either a part or an assembly, the design tasks in this phase are different under these two conditions:
   a) For a part, the detailed shape is gradually generated based on the skeleton determined in the previous design phase.
   b) For an assembly, things are much more complex. The designer does structure decomposition first. Then the abstract designs of each child component are carried out jointly based on the functional decomposition of the assembly's function. Next, the layout design is carried out during which the positions of these child components and the high level assembly interfaces among them are determined. Meanwhile, the skeleton designs of each child component at next level are carried out jointly based on the skeleton of the assembly (because these child components may couple together tightly). Further, the low level interfaces are derived from high level interfaces and implemented on the child skeletons. Therefore the skeletons can be assembled together for various early analyses so far. And then the detail design is carried out concurrently and independently on each child component based on the previous design results, i.e., the skeletons (this step is recursive). At last, the detailed child components are assembled together as a whole and the detailed model of this assembly is achieved.

The whole process described above can be summarized as Fig. 1.

Notably, during practical design, designers may not want to do all these works mentioned above for each component since only "nontrivial" components deserve these efforts and some "trivial" components are just so familiar and simple to them that the high level abstractions could be omitted. But in this paper, for consistency and simplicity, we will not deal with this issue. Actually this flexibility will not be hard to be implemented based on our model.

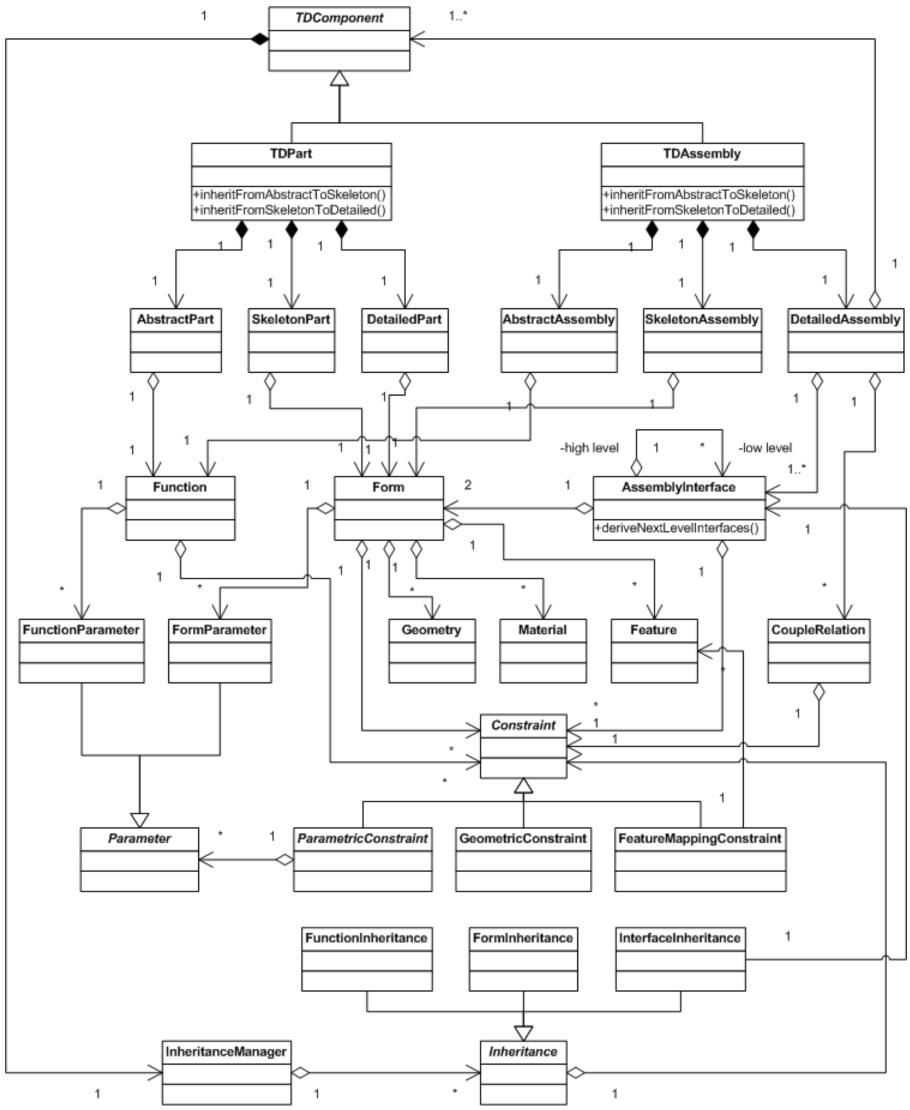
Figure 1 The top-down assembly design process



3.2 *The extended multi-level assembly model for top-down assembly design*

The extended model is shown in Fig. 2. Since the skeleton and inheritance are the emphases in this paper, we only show exhaustively the most related portion in the model and others are abridged.

Figure 2 Class diagram of the extended top-down assembly model



TDPart and ***TDAssembly*** are the main classes controlling the product structure and storing the product data in the whole design process. ***AbstractPart***, ***SkeletonPart*** and ***DetailedPart*** store the three different models which represent the design result corresponding to the three design phases for a part. And ***AbstractAssembly***, ***SkeletonAssembly***, ***DetailedAssembly*** behave similarly as the three classes above for an assembly. ***TDComponent*** is the abstract base class that represents a component which could be either a part or an assembly in the product. Every component has an ***InheritanceManager*** which manages all the inheritances happening in this component. These inheritances reflect the relations among models of different design phases. Each inheritance has an underground constraint as the concrete representation. And ***Inheritance*** and ***Constraint*** are the abstract classes for this use. There could be many kinds of inheritances and constraints inherited from these two classes for different uses in the top-down assembly design process. The abstract model of part and assembly mainly contains function information and the skeleton model mainly contains form information. The detailed model of part contains the final shape of this part, as the detailed model of assembly has all the assembly interfaces and couple relations among its child components beside references to these child components model.

4 Skeleton and skeleton architecture

4.1 Definitions

The skeleton design is the key to correlate the conceptual design earlier and the detailed design later. Therefore the skeleton model is a vital pivot between the abstract model and the detailed model.

Based on our summarization of the top-down product design process, one goal of skeleton design is generating a rough 3D geometry model as the carrier and presenter of significant form parameters; the other goal is generating the design space for later design. And the significant form parameters could come out in several ways shown below.

1. Derived from functional design result.
2. Derived from skeleton of parent assembly.
3. Derived from assembly interfaces defined on this component.

These derivations are all actually inheritances which we will discuss in section 5.

In a word, the *skeleton* can be defined as follows:

> *Skeleton is a preliminary but sufficient 3D geometry model with definitions of significant form parameters at current design level in the assembly tree and it is the design base of the next design phase as both the space restriction and form restriction.*

On the other hand, due to the fact that the skeleton design of a component is always coupled with the skeleton design of his brother components (see Fig. 1) except for the product root assembly, we need another new concept here.

At the beginning of detail design of an assembly, the designer decompose this assembly to determine the child components it has and design the abstract models of these child components together. After that, assembly interfaces among these child components and skeleton models of them could also be determined. So far, an essential architecture under this assembly is obtained. Now we can define this *skeleton architecture* as follows:

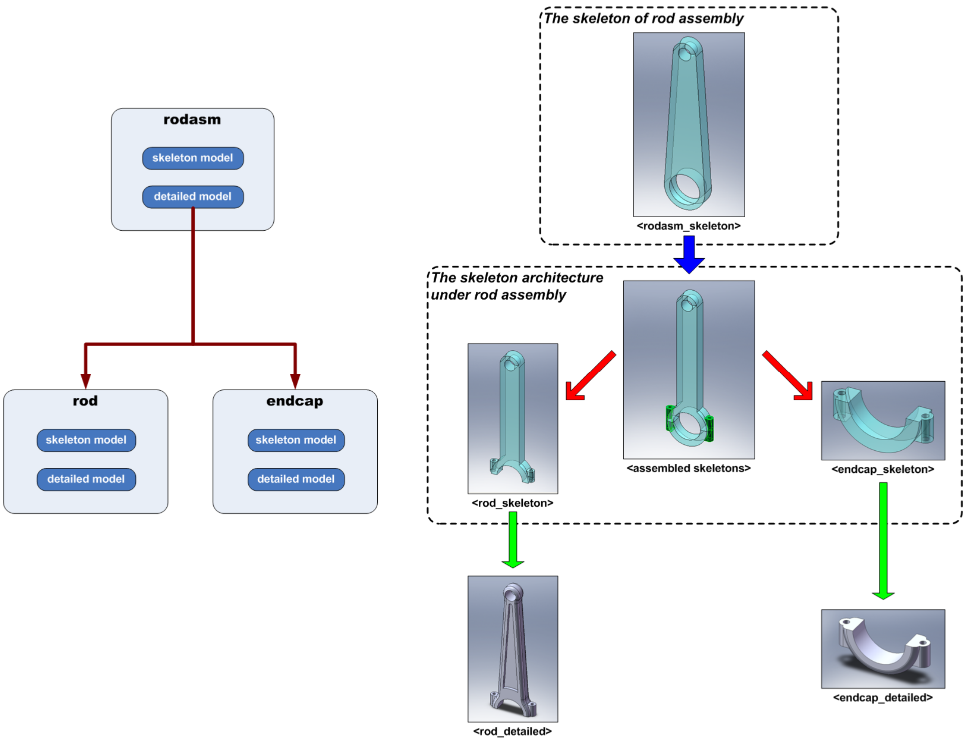
> *Skeleton architecture is the first design outcome in the detail design of an assembly. It is a complete assembled structure under this assembly with all the functions and vital geometries of direct child components defined and all the assembly interfaces and couple relations among these child components established.*

With this skeleton architecture, some high-level analyses, e.g. the kinematic analysis could be carried out quite early. Actually, in our opinion, the design of this skeleton architecture is the direct detail design of this parent assembly and the detail design recursively carried out later on its child components is the indirect detail design of it.

4.2 A sample

A simple sample in Fig. 3 illustrates the skeleton and skeleton architecture of an assembly clearly. The function related information is omitted here since they are quite obvious and straight. This design piece starts from the skeleton of the assembly *"rodasm"* which is generated in upper design (i.e. it's a sub-assembly in the whole product). Then this assembly is decomposed to two parts *"rod"* and *"endcap"* considering the assembling feasibility on upper level of product; next, the pin connection semantic assembly interface is determined. And then, the skeletons of "rod" and *"endcap"* are designed out together under their function requirements and the form restrictions stored in skeleton of *"rodasm"*. After that, the low level form features of pin connection assembly interface are implemented on the two child skeletons respectively and a preliminary assembly can be assembled from the two skeletons (So far, the design of the skeleton architecture under assembly *"rodasm"* is finished and this architecture can be used to carry out some necessary analyses quite early in the whole design process). Further, the detailed models of part *"rod"* and *"endcap"* are designed out based on their skeletons respectively in the end. At last, the detail design of *"rodassembly"* is finished by assemble the two detailed parts together.

Figure 3 The skeleton and skeleton architecture of rod assembly



5 Inheritance

As shown in the sections above, 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. Although we know what the design principles and contents of the corresponding models are in different design phases, how does the design transfer from one phase to another smoothly and accurately? The *inheritance* mechanism is needed here.

5.1 Inheritance occasions and tasks

1. *Inheritance between abstract design and skeleton design of a component*
   When designers switch from abstract design to skeleton design, the functional information needs to be inherited to reveal as form information. That means the constraint between function and form must be established and maintained hence the high level functional information could be used to drive the later design phases.
2. *Inheritance between abstract design and detail design of a component*
   The inheritances happening on this occasion are quite different when the designed component is either a part or an assembly.
   a) For a part, this situation is unusual since in general all the functional information in abstract model will be inherited to its corresponding skeleton model as backbone form information.
   b) For an assembly, functional decomposition will happen when designers are determining the assembly's skeleton architecture. As we know, the abstract designs of this assembly's child components are carried out here thus actually the constraints are maintained among the functions of the parent assembly and the functions of the child components.
3. *Inheritance between skeleton design and detail design of a component*
   Like term 2 above, the inheritances happening on this occasion present differently along with the different kind of the designed component.
   a) For a part, the rough form information needs to be preserved in its detailed model as guidance therefore the constraint connecting the skeleton form information and the detailed form information must be established.
   b) For an assembly, constraints are maintained among the forms of the parent assembly and the forms of the child components.

5.2 Inheritance classification and representation

As shown in Fig. 2, inheritances have several representations fitting different situations described above. The key for the dissimilar capabilities of inheritances is the variety of underground constraints. Details about the different inheritance representations are explained below.

1. *Function inheritance*
   This kind of inheritance mainly grounds on parametric constraint. Non-linear equations and inequalities are established among function parameters and form parameters. They may be solved one by one since only one component's skeleton is

dealt with, or be solved concurrently as a system since many child components couple with each other in the skeleton architecture of an assembly's detailed model.

2. *Form inheritance*

   There may be several possibilities under form inheritance.

   a) Parametric constraints are used as above but there are just form parameters taking part here.
   b) Geometric constraints are used to establish mutual restrictions among geometric elements in skeleton model and geometric elements in detailed model (as before, it may be the detailed model of a part or the skeleton models of an assembly's child components). Geometric constraints solver is needed here and this kind of inheritance happens at 2D sketch design commonly since real 3D geometric constraints solver is not mature and convenient now. Further, this way is suitable for space design when the detailed model need to be leaned inside the design space of skeleton model.
   c) Other higher level constraints are used. Our constraint architecture can be extended to have some more complex constraints such as feature level constraint. In Fig. 2 we have defined a feature mapping constraint which copies an original feature to a target one and maintains the connection between them. It may not be as easy as it looks like. Besides plain copy operation, this mapping will establish parametric constraints between the internal feature parameters of the original feature and the corresponding feature parameters of the target feature. Nonetheless, the geometric references are still needed to be specified to locate the target feature. Occasionally the original feature will have some geometric references implicitly defining some feature parameters and in such case these parameters also need to be extracted and mapped properly.

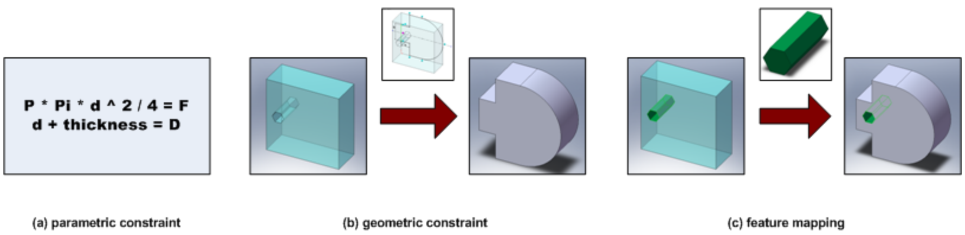
3. *Interface inheritance*

   Interface itself is a complex problem. We believe that interface should have different semantic levels therefore derivations from high level to low level should be considered, and the design of interfaces should be supported explicitly and effectively with the inheritance mechanism on interfaces design. Actually we will discuss the topic about interfaces design in another paper soon.

5.3 *Inheritance sample*

The simple samples in Fig. 4 below are typical illustrations of some kinds of inheritances.

Figure 4 Various kinds of inheritances



6 Adaptation and extension for commercial systems

The presented top-down assembly model is platform independent hereby every commercial CAD system which supports traditional feature based modeling could be theoretically adapted and extended to build a special and effective tool for top-down assembly design based on our model.

Two key items about adaptation are analyzed below.

1. *Assembly structure adaptation*

   As shown in Fig. 5, we can use native part and assembly of commercial CAD systems as the underground model for the top-down assembly model. ***SkeletonPart***, ***DetailedPart*** and ***SkeletonAssembly*** have native parts under them and ***DetailedAssembly*** has a native assembly under it. Every ***TDComponent*** has a host assembly for aggregating the underground native parts and assemblies of skeleton model and detailed model in this component.

   In this way, the original modeling function and data management supported by commercial CAD systems could be naturally involved in the top-down assembly model.

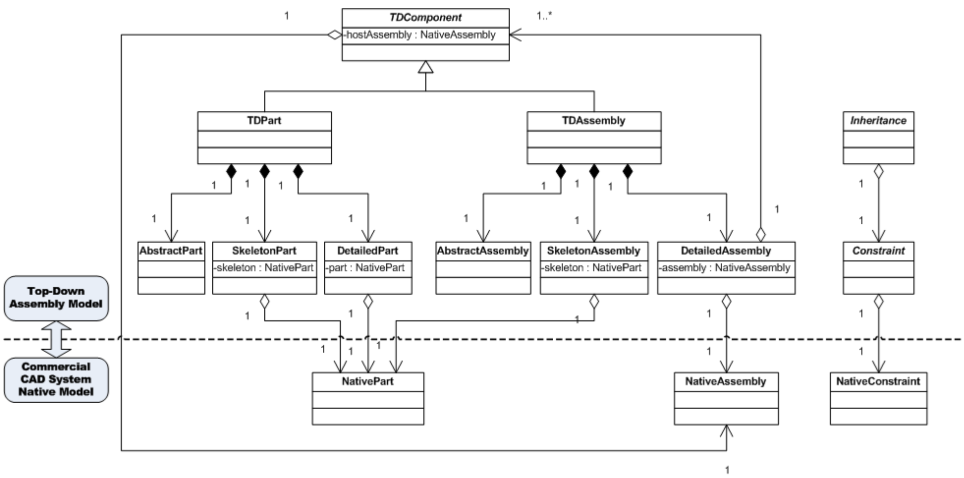
2. *Constraint adaptation*

   Constraint is the base of the inheritance mechanism. Currently, the supports of different commercial CAD systems on constraints are different. For parametric constraint, generally the simple parametric constraint is supported but complex parametric constraint system such as non-linear equation system is absent. For form constraint, external references among parts and assemblies are needed but some systems may lack this support. And for higher-level constraints like feature mapping constraint, there is no direct systematic support in general.

   In all, for those constraints supported by commercial system, just a reference to the native constraint is needed in the constraint of our model (Fig. 5). But for those unsupported constraints, independent modules must be implemented upon the commercial CAD systems to maintain them.

In a word, with reasonable adaptation and extension, a commercial CAD system could be transferred to support the top-down assembly design well.

Figure 5 Adaptation model for commercial CAD systems



7 Applications of the skeleton model

Beside its purpose in the top-down assembly design, the skeleton model could be useful in some other extra aspects. Two examples are illustrated below.

7.1 Generation of the native skeleton model used in Pro/Engineer

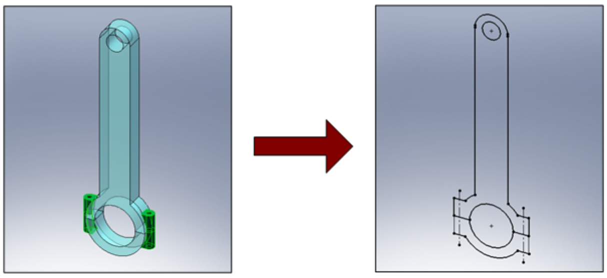
Some commercial CAD systems such as *Pro/Engineer* also have the conception of skeleton. But the meanings of this skeleton are different from our skeleton model. I will mention the skeleton in commercial CAD systems as native skeleton below.

The native skeleton emphasizes on assembly's layout structure such as the positions of child components and the degenerated assembly interfaces among them. Essentially it's a part that defines some shape profiles and some datum geometries for assemble. On the other hand, the skeleton in our model is defined on each component as a shape abstraction (full definition in section 3.2.1) for later design.

Actually the skeleton architecture discussed in section 3.2.1 is the superset of the native skeleton. And by performing skeleton degeneration and interface degeneration suitably, the native skeleton could be generated from the skeleton architecture which implicitly stored in the detailed model of an assembly. This is quite useful when the native skeleton is needed to protect the intellectual property [10].

Fig. 6 shows an example of the native skeleton that could be derived from the rod assembly.

Figure 6 Generation of native skeleton used in Pro/Engineer



7.2 Generation of the multi-resolution assembly model

Traditional multi-resolution assembly model generation methods have some unavoidable shortcomings as shown below.

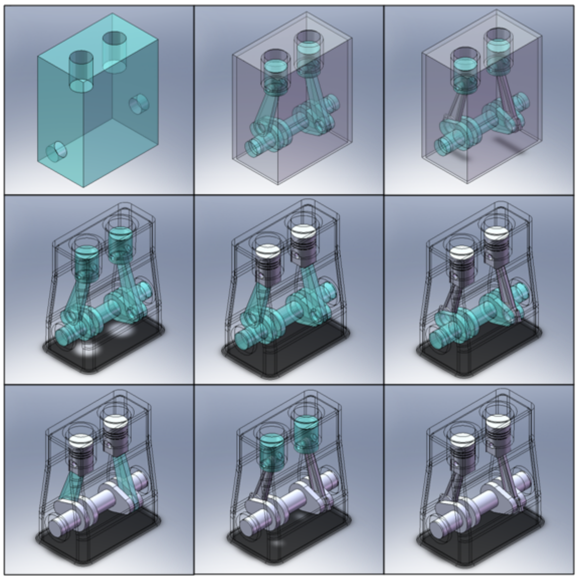
1. Because the multi-resolution assembly model is generated by performing part simplification on each part in this assembly circularly, the assembly interfaces are apt to be destroyed. And even if assembly interfaces retainable simplification is executed, it is still possible to generate an invalid assembly with inner interferences existing due to the lack of global knowledge about this assembly.
2. The generated multi-resolution assembly model has lost the design intents because in general the simplification algorithm considers geometric information only.

And on the contrary, our top-down assembly model preserves sufficient information in the design process to generate valid and knowledgeable multi-resolution assembly model.

Skeleton substitution is the key to achieve this goal. On each node of the whole product assembly tree, either skeleton model or detailed model could appear in the result assembly model. For a part, the skeleton or the detailed part could be used. For an assembly, the skeleton could be used alike, or the assembled child components could present instead. And for each child components, these choices appear recursively. In the end, a series of multi-resolution assembly model could be generated.

Fig. 7 shows a sample of multi-resolution model generation.

Figure 7 Some multi-resolution models generated by skeleton substitution approach

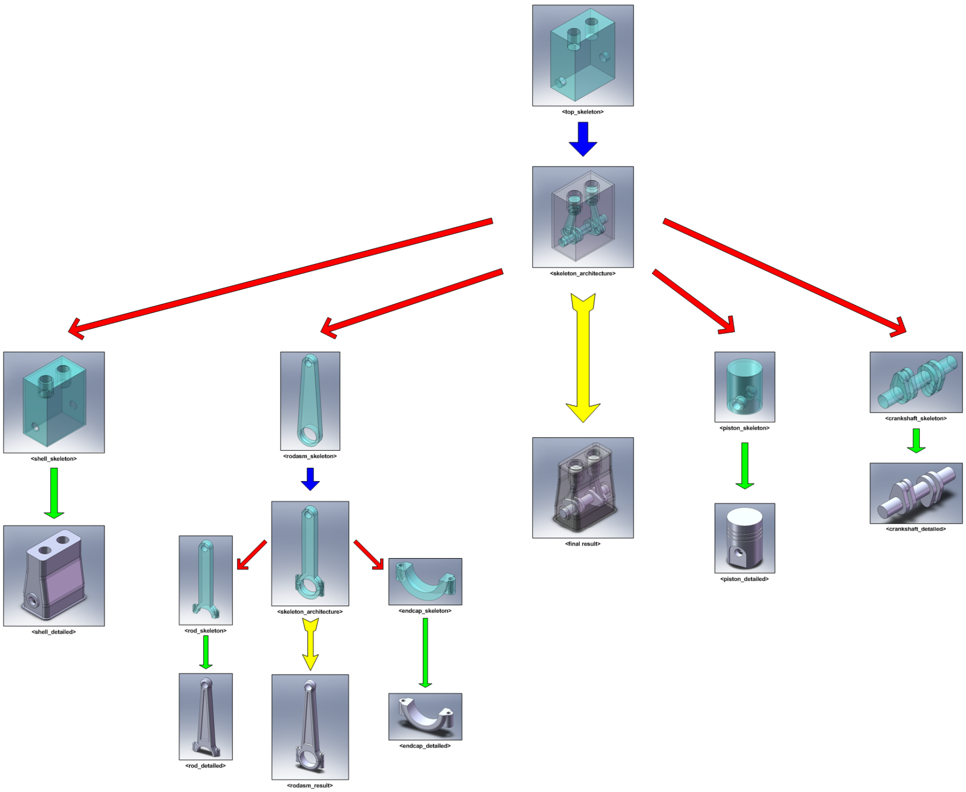


8 Implementation

To verify the feasibility and capability of the top-down assembly model, we constructed a top-down design module by using the adaptation mentioned in section 6 on SolidWorks 2007. This module is written in C# based on Microsoft.NET Framework 2.0. And Microsoft.NET Remoting Framework is used as the underground distribution technology for collaborative design which will be covered in the future.

We have used this top-down design module to design a crank-connecting rod-piston mechanism in motor's engine. It has six components, i.e. two pistons, two rod assemblies, a crank shaft and a shell. Fig. 8 shows a brief profile of the design process.

Figure 8 The top-down design of the crank-connecting rod-piston mechanism



Within this graph, it could be seen that how the target assembly is designed out from a rough single node to the full detailed model. All the vital elements in top-down assembly design such as structure decompositions, skeletons, inheritances, coupled relations, interfaces and so on are managed properly during the design process so the design activities could be well supported and even recovered based on these information.

9 Conclusions

In this paper, an extended multi-level assembly model which better supports top-down assembly design has been proposed. Based on the definition of skeleton and skeleton architecture, the inheritance mechanism could be adopted to maintain the relations and restrictions among the respective information model of different design phases. And the presented assembly model is neutral hereby most current commercial CAD systems could be adapted to implement the specialized top-down assembly design module based on this model.

We have also shown examples of how the skeleton model could be used to do some extra beneficial work. And in the future, we will explore some other places where it could be applied such as the collaboration, the assembly search and so on.

Acknowledgment

The authors are very grateful to the financial support from NSF of China (No. 60574061).

References

1. Mäntylä, M. (1990) `A modeling system for top-down by design of assembled products`, *IBM Journal of Research and Development*, 34(5), pp. 636—659.

2 Gui,J. K., and Mäntylä, M. (1993) `New concepts for complete product assembly modeling`, *Proceedings of Solid Modeling, Second ACM Symposium on Solid Modeling and Applications*, Montreal, Canada, May 19-21.

3 Csabai, A., Stroud, I., and Xirouchakis, P. C. (2002) `Container spaces and functional features for top-down 3D layout design`, *Computer-Aided Design*, 34 (14), pp. 1011-1035.

4 Brunetti, G., and Golob, B. (2000) `A feature-based approach towards an integrated product model including conceptual design information`, *Computer-Aided Design*, 32(14), pp. 877-887.

5 Aleixos, N., and Company, P. (2004) `Contero M, Integrated modeling with top-down approach in subsidiary industries`, *Computers in Industry*, 53(1), pp. 97-116.

6 Rachuri, S., Han, Y. H., Foufou, S., Feng, S. C., Roy, U., and Wang, F., 2006, `A Model for Capturing Product Assembly Information`, *Journal of Computing and Information Science in Engineering*, 6(3), pp. 11-21.

7 Sudarsan, R., Fenves, S. J., Sriram, R. D., and Wang F. (2005) `A product information modeling framework for product lifecycle management`, *Computer-Aided Design*, 37(13), pp. 1399–1411.

8 Bidarra, R, Kranendonk, N, Noort, A, and Bronsvoort, W. F. (2002) `A collaborative framework for integrated part and assembly modeling`, *Proceedings of Solid Modeling, seventh ACM Symposium on Solid Modeling and Applications*, Saarbrucken Germany, June 17-21.

9 N. Shyamsundar and Rajit Gadh (2001) `Internet-based collaborative product design with assembly features and virtual design spaces`, *Computer-Aided Design*, Vol. 33, No. 9.

10 Parametric Technology Corporation, `http://www.ptc.com`

11 PTC, `Pro/ENGINEER Advanced Top-Down Design`, Pro/ENGINEER, Release 2000i^2, T823-310-01.

12 Jinsang Hwang, Duhwan Mun, Byungchul Kim, Soonhung Han (2008) `Securing enterprise intellectual property using a skeleton model in a collaborative product design environment`, *5th International Conference on Product Lifecycle Management (PLM'08)*, July Seoul Korea 9-11 July 2008

13 Shuting Zhang, Xiang Chen, Shuming Gao, Youdong Yang (2007) `A framework for collaborative top-down assembly design`, *Proceedings of ASME DETC2007.*