Journal of Southeast University (English Edition) June 2001 Vol.17 No.1 ISSN 1003—7985

# Logical-Structure Modeling for Conceptual Design*

Sun Zhengxing** Zhang Fuyan

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

**Abstract**: Based on the definition of a logic structure feature to relate logically functional requirements to geometric representation independent upon detailed geometric representation, this paper presents an idea of logical structure modeling for computer aided conceptual design and makes attempt to establish a representation formalism of logic structure modeling. The definition and representation of logical structure feature are given and an assembly module definition for supporting top-down conceptual design is also proposed. The proposed scheme contributes to several aspects of conceptual design research, especially to provide elementarily a formal methodology for computer aided conceptual design system development and operation.

**Key words**: conceptual design, function-form transformation, logic structure feature, representation formalism

Engineering design can be viewed as a transformational process that configures the structure to provide the function and occurs between two main twisted lines while design evolves: a functional aspect which leads to a logical product description and a geometric aspect which forms a physical representation. This transformation is incremental and often iterative and can be divided into several stages such as conceptual design, embodiment design and detailed design. The commonly known fact is that the impact of the conceptual design phase on the rest of the design process is profound. Though CAD techniques have been made a great deal of successful researches and applications since it was born, existed CAD researches focused largely upon geometric modeling has had little impact at conceptual design stage. Recent need for integrated, concurrent and intelligence CAD system requires the support of such a phase.

Because it is very difficult to understand completely the regularity of design thought and the proper process of conceptual design, existing researches for the support of conceptual concentrate mainly upon the use of the man-machine cooperation in solving design problem now. That is, designer makes heuristic analysis firstly and then makes decisions of design scheme in terms of knowledge bases. For example, Sreenivasa, et al.[1] presented a knowledge-based framework for conceptual design. They presented an approach to map an evolving symbolic description of design into a geometric description, which generates multiple alternatives based on the representation of spatial relationships allowed qualitative and numerical specification of constraints and relationships between objects in building engineering. However, such proposed researches resulted only to provide a connection way from function to shape, it currently lacks a formal methodology for computer aided conceptual design system development and operation. Accordingly, based on defining a logic structure feature to relate logically functional description to geometric pattern representation independent upon detailed geometric representation, authors present an idea of logical structure modeling and make a try to establish the representation formalism for computer aided conceptual design in this paper.

## 1 Logic Structure Feature

### 1.1 Definition of logic structure feature

During conceptual design process, a designer may concentrate on the functional viewpoint, who should determine the functional structure of a design object and basic physical mechanisms that realize functional structure, postpone decisions on its physical appearance and avoid the border of expressing geometric details. Our central hypothesis is that support for form conception must closely involve support for functional symbol evolution of design. At the conceptual design stage, this allows a user to: (a) consider several alternative, both in terms of functional decompositions, and in terms of physical implementations of these decompositions; and (b) explicitly capture the essential functional intent behind the design. A physical effect is laid down with regard to a special function or sub-function and at the same time is usually characterized by plotting a certain arrangement

Received 2000-10-10.

* The project supported by the National Natural Science Foundation of China (69903006).

** Born in 1964, male, associate professor, doctor.

and interrelationship of primary functional faces. In brief words, any component used for conceptual design should relate shape elements (syntax) to functions (semantic). Therefore logic structure feature is introduced as an abstract class[2] which connects logically functional description with geometric pattern which embody the preliminary faces and their links of physical effect to meet the requirements of a certain design function or sub-function, as shown in Fig.1. We can represent the logic structure feature LSF formally as a triple of functional description FDL, geometric representation GRE and the relations RFG between them:

$$LSF = \langle FDL, GRE, RFG \rangle$$

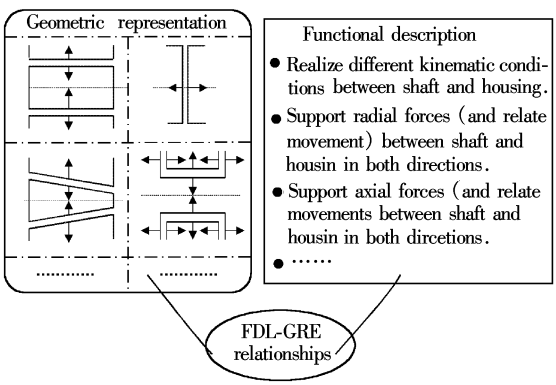


**Fig.1** Definition of logic structure feature

For definition of logic structure feature, we explain some aspects as follows. Firstly, unlike the commonly known assembly units, which always refer to the shape elements found in one individual part, logic structure feature usually contain information about shape elements of several modules (parts, sub-assemblies) and their relation, and not need detailed geometric information. Secondary, unlike traditional principal components, which are always used to indicate main function or principal of component with little geometric meaning, logic structure features represent both the descriptions of functions or sub-functions of design object and the global geometric constructions of product related to correspond functions or sub-functions. Finally, logic structure feature shows hierarchical graduation of functional decompositions and the global geometric assembly relationships of product.

Fig.1 shows an example of a simple logic structure feature called "support a shaft" related to the basic function "input or output torque/speed". The possible type of geometric pattern GRE of logic structure feature is listed in left side of the figure and the semantic meaning FDL related to this function is explained on the right side. As its underlying physical effect can only be realized by at least one pair of functional faces, the logic structure feature consists of two functional faces (more pairs are possible, of course). The necessary separation of functional faces, indicated in the figure by material vectors, will automatically result in separate modules and separate assembly units at later design stages. The relation RFG is multiple and must be knowledge-driven. To use logic structure feature effectively, the data base of logic structure feature must be created firstly based on the classification of functions and the complexity of shape related to functions. This is done as shown in Fig.1 by means of creating the base of functional description and the base of geometric pattern oriented to specific applications respectively, then establishing the relationships between two bases. This can be represented as $m \times n$ incidence matrices: $RFG = [r_{ij}]$, where $r_{ij} = 1$ indicates that function $i$ is related to pattern $j$; $r_{ij} = 0$ represents that function $i$ is not related to pattern $j$. Of course, some other relations such as process related, characteristic related, parameter related can be defined according to application domains. The more details of representation of logic structure feature will be discussed in the following sections.

To define and classify the logic structure feature, we suggest the following general rules.

● Function description The logic structure feature definition should include information about the decision-making process that leads to a particular design. That is, the functional descriptions in logic structure feature definition must be suited for the representation, decomposition and synthesis of function in conceptual design.

● Geometric pattern The geometric constitution in logic structure feature should only be represented by primary geometric elements corresponding to specified functions or sub-functions and can be used to model the global assemble relationships of product to be designed to represent design alternatives independent upon detailed geometry.

● Relationship definition The relationship between function and shape in logic structure feature is multi to multi relations. The granularity of logic structure feature depends upon the mating conditions of components to be designed. In other words, assembly of logic assembly would not consider the intersection of geometric entities and not deal with detailed geometry solution.

● Attribute representation The attributes should contain three aspects: functional, geometric and relationships between them. Functional aspects represent mainly the characteristic parameters of product such as magnitudes of force, speed, etc. geometric aspects mainly concern with the global

geometric constitution of the object to be designed and the relationship aspects must be knowledge-driven mapping between function and shape. In particular, the process of the conceptual design should be adequately represented. Perhaps the most important requirement is that the representation should be computationally tractable.

- Design process The logic structure feature definition must be used to model product in multi-levels suited for the polymorphism or the domain-dependent of design process and to provide a guide for the following detailed design.

In practice, logic structure feature can be classified according to the working form of component used in mechanical system and the requirements of primary characters in product's life circles. Its implementation is dependent on the utilization of restraining, matching of form features or function units with known function such as standard components, common used components, general components and user-defined components. There are different methods for classification corresponding to different viewpoints. In general, they can be divided into basic and complex. They include the working unit, the characteristic unit, structural unit, the manufacturing unit and man-machine character unit according to the types of functions. They can be represented in form feature level, form feature coincidental level, simple part level and complex product level dependent on the complexity of shape structures also.

## 1.2 Representation of logic structure feature

### 1.2.1 Function description

A function represents the relationship among input(s), output(s) and state variable(s) of a system or subsystem. The arrangement of individual functions or the relationship between the overall function and the sub-functions is expressed with the help of the function structure aimed at logic considerations.

Our representation of function uses functional descriptions of products or processes according to a set of linguistic and hierarchic rules. Function representation comprises a three-tiered structure consisting of: function blocks (compact verb-noun descriptors of what the design does rather than what it is with links to other blocks); allocations (constraints, performance requirements, specifications and resources); and components (artifacts that satisfy the given function). The function block (or node) contains the function name (what is done) expressed as generic noun-verb pair to describe the function of the product or process. The function variables consist of verb-noun pairs and link between these.

Since design function in the design of shafts and their adjacent components are easily comprehensible[3], they are taken as an example to demonstrate the definition of logic structure feature shown in Fig.1. This apparently simple construction represents a design in a fundamentally abstract form. Each verb-noun pair represents a conceptual decision, which is subject to the familiar "brainstorming" and creativity techniques, but is conceptually free from physical constraints. It requires a deep understanding of the problem at hand and promotes discussion, especially among a group of designers, because it is often difficult to think about a design in this way without practice. The resulting set of descriptions, connected by links, represents the full intent of the design in a reasoned hierarchy.

Function links represent the relationships among the function descriptions. The nodes to the one side of the function node represent the reason why a function is included: higher level functions. The nodes to another side are functions describing how the function is performed: lower level functions. Links connect each high-level function with its lower level functions according to this how/why relationship. Links other than how/why have been identified among function blocks, viz cause, temporal, informational, alternative, and revisions.

The form variables consist of the allocations attached to these verb-noun pairs. Because the form of the artifacts are deliberately suppressed during value engineering, the allocation list supplies what needs to be known about the design in abstract terms: constraints, performance requirements, specifications and resources. When complete, the allocations specify the behavior of each function and set the constraints for the form of the design. The synthesis of form itself in the component choices and details is not carried out by the conceptual value engineering process, but rather by the parallel process of proposing and testing alternatives.

In brief, this can be formalized as follows:

$$\mathrm{FDL} = \langle (\mathrm{FN}, \mathrm{FID}), (\mathrm{FV}, \mathrm{FO}), (\mathrm{FP}, \mathrm{FN}), \mathrm{ALLO} \rangle$$

where (FN, FID) is an identifier of function description; (FV, FO) is a verb-noun pair; (FP, FN) is a prepositional phrase that represents the function link; ALLO is a set of variables of function.

### 1.2.2 Geometric pattern representation

The geometric pattern of the logic structure features are well defined if the combination of several primary functional faces leads to a generic description of the related design function and physical effect. The primary functional faces thus represent the key

elements of the geometric aspects of logic structure feature. Geometric representation of logic structure feature is mainly function-oriented. Firstly, it is a principal symbol that represents only geometric pattern. Secondly, it is composed of the primary functional faces and some secondary faces that are used to represent the global assembly mating conditions. The process of defining geometric pattern of logic structure feature in the design process can be summarized as follows: (1) Starting from the lowest function block of a technical product to be designed in function descriptive hierarchy, physical effects have to be found to realize all the functions and predefined geometric pattern of logic structure feature can firstly be the primary functional faces of each physical effect. (2) The division of the principal solution into realizable modules can be accomplished by combining related functional faces. (3) Filling faces have to be inserted to obtain a preliminary layout of each module. (4) In order to meet manufacturing, assembly and standardization requirements, geometric elements (primary functional faces and filling faces) have to be separated by inserting (pairs of) secondary functional faces. Consequently individual assembly units and their relations to each other are determined.

The main mathematical characteristics of geometry of logic structure feature are expressed as shown in Fig.2 by vectors. The original coordinate of vector determines the position of logic structure feature in assembly of product and is used as origin of local coordinate system of unit. The direction of vector indicates the assembly directions of logic structure feature in product assembly, that is, the direction of the actions of functions or the mating directions between the units in product construction. The choice of origin and direction of logic structure feature must reflect the symmetry of geometric construction and/or the nature of realization of functions. Other parameters for determining logic structure feature are expressed by composite norm. The composite norm is a multiple vector recording the parameters of construction and other attributes such as geometric box, main dimensional and functional characteristics (surface finish, precision and so on). Therefore, the geometric pattern of logic structure feature can be formalized as:

$$\mathrm{VF} = \langle (X_0, Y_0, Z_0), (\alpha, \beta, \gamma), (\mathrm{BOX}, \mathrm{MD}, \mathrm{PRE}, \cdots) \rangle$$

where $(X_0, Y_0, Z_0)$ is an origin of local coordinate of logic structure feature; $(\alpha, \beta, \gamma)$is a direction of vector, that is the angle between vector and the axial of coordinate, and $0 \leqslant \alpha, \beta, \gamma \leqslant 180°$. In composite norm, BOX represents geometric box of logic structure feature, MD are the main dimensions in geometric construction of logic structure feature, PRE indicate the main precision of mating condition for assembly of units.

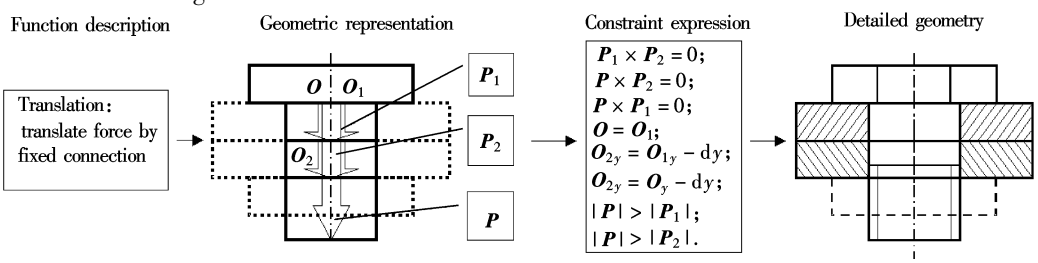


**Fig**.2 A simple example of logic structure feature representation

Accordingly, geometric representation of logic structure feature can be formalized as follows:

$$\mathrm{GRE} = \langle \mathrm{FN}, \mathrm{FID}, \mathrm{FFL}, \mathrm{GPT}, \mathrm{RPT}, \mathrm{VF} \rangle$$

where FN, FID represent the name and identifier of logic structure feature, respectively; FFL is a list of the functional faces that construct the geometric pattern; GPT is a pointer to display the geometric elements; RPT indicates the relationships between logic structure feature mating each other in product; VF is a vector.

## 2 Logic Structure Modeling for Conceptual Design

### 2.1 Description of conceptual modeling

Based on the proposed formal definition scheme in previous sections, the conceptual design can be seen as assembly process of logic structure features, which includes two aspects: logic assembly modeling and geometric pattern assembly modeling. We can formalize this process in terms of formal language theory and frame theory as follows:

$$\begin{aligned} \mathrm{DP} &= \langle \mathrm{DP}_{\mathrm{FRL}}, \mathrm{DP}_{\mathrm{GRE}}, \mathrm{RFG} \rangle \\ &= \langle \mathrm{DOM}, \mathrm{MS}_0, \mathrm{MS}, \mathrm{DT}, \mathrm{ROL} \rangle \end{aligned}$$

in which, DOM indicates the scope of design problem definition; MS is a set of middle states of product model produced during conceptual design process. That is, $\mathrm{MS} = (\mathrm{MS}_1, \cdots, \mathrm{MS}_i, \cdots, \mathrm{MS}_T)$, $\mathrm{MS}_0$ is an initial model of design (It is a model of compared product in analogous design, an old product in alternative design and a conceptual product in revolutionary design respectively) and $\mathrm{MS}_T$ is an ordered set of model of product to be designed in terms

of the defined data structure according to ROL; ROL specifies the relations between MS and DT. Usually, ROL is networked parent-child graph and can be expressed as: ROL = $\{N, E, A\}$, where "$N$" indicates logic structure feature node, "$E$" is the directive edge which points from parent unit to child unit according to design sequence, "$A$" denotes the rules or attributes for establishing unit dependency; DT indicates a set of design translation actions on the unordered paired elements of $N$ depended on rules in $A$ when design evolves. One DT may require one more inputs and may produce one more outputs dependent upon the type of DT. Every DT results to the change of product model. The accomplishment of a DT depends upon the application of knowledge KD including general knowledge, special knowledge, design methodology and standard, experimental data and so on, reasoning methods RM including qualitative, quantitative, fuzzy and intuitive reason and so on and calculating tools CT. That is, DT = $\langle$KD, RM, CT$\rangle$. All the elements in DP definition deal with both logic aspect and geometric aspect, that is

$$\mathrm{DOM} = \mathrm{DOM}_{\mathrm{FDL}} \bigcup \mathrm{DOM}_{\mathrm{GRE}}$$

$$\mathrm{MS}_0 = (\mathrm{VI}, \mathrm{MG}_0), \mathrm{MS} = (\mathrm{VM}, \mathrm{MG})$$

$$\mathrm{DT} = (\mathrm{FDL}, \mathrm{GRE})$$

$$(\mathrm{MS} \bigcap \mathrm{DT}) = (\mathrm{DT} \bigcap \mathrm{MS}) = \varnothing$$

$$\mathrm{ROL} = (\mathrm{ROF}, \mathrm{ROG}) \subset (\mathrm{MS} \times \mathrm{DT}) \bigcup (\mathrm{DT} \times \mathrm{MS})$$

The meaning of each element is explained according to two aspects in the following.

Elements in the logic modeling are described as follows. VI is a set of initial vocabulary that describes the global functions of product to be designed in function descriptive grammar. VA is an ordered set of vocabulary in terms of the defined data structure to describe the middle functions during function decomposition in functional descriptive grammar and can be divided into non-terminated vocabulary and terminated vocabulary that represents the basic function that can be accomplished directly by special geometric component. Each vocabulary is corresponding to a function descriptive grammar FDL to describe the engineering meaning and characteristics of this function and the relationships in the function decomposition hierarchy. FDT is a set of functional decomposition translation actions according to the functional descriptive grammar FDL. Each action in FDL deals with two aspects: parsing and calculating. Grammar parsing is used to create vocabulary and result to function hierarchy of product, and attribute calculation is used to calculate the characteristic parameters and their inheritance and derivation relationships between functions in function hierarchy. ROF is a set of rules of production to describe the hierarchical relationships of function decomposition and regularities of calculation to determine the relationship between different attributes of different functions for the functional descriptive grammar FDL. The rule of production determines the method and sequence of the creation of vocabulary or the order of function decomposition. The regularity of calculation represents the method of inheritance and derivation of attribute of function.

Elements in geometric pattern assembly are explained following. $\mathrm{MG}_0$ denotes the basic geometric assembly unit, which is selected if any unit can accomplish the most important function of product or is connected with the most amounts of other assembly units. MG is the ordered set of geometric pattern assembly units according to mating conditions in terms of the defined data structure. It represents the state of middle model of product during assembly process. GDT is a set of assembly operator acted on GRE such as select, create, modify, add, delete and so on to be used to specify the design constraints or mating condition between assembly units. ROG indicates the mating condition defined in GRE such as sliding, clearance, transition, interference, fastener, gear, sleeve, ball, roller, hydrodynamic and so on, which are represented in terms of geometric pattern vector such as position relation ($\mathrm{d}x$, $\mathrm{d}y$, $\mathrm{d}z$), angular relation ($\mathrm{d}\alpha, \mathrm{d}\beta, \mathrm{d}\gamma$), and the relation of composite norm that may be equal, greater than and less than. There are many benefits using mating condition to express design constraint. Firstly, it makes it possible to assemble at a higher abstract level in terms of symbolic representation. Secondly, it makes designer assemble avoiding the use of the complex geometric transformations and detailed geometric representation, and can be refined the results during detailed design. Finally, it can support parameterization well because symbol representation can easily propagate all constraints to all components and maintain all these mating conditions during design evolve.

## 2.2 Representation of logical structure model

There are two typical structures for the representation of assembly model: graph[4] and virtual link[5]. However, these are oriented to detailed geometry design, in which detailed part geometry must be defined firstly, then can be used to represent assembly, and cannot be suited completely for the representation of assembly in conceptual design. So existing structures must be extended to store functional and physical data.

Actually, an assembly model of product is a complex and hierarchical network and can be decomposed in the special form into a regular data

structure according to the feature of geometric construction. For example, assembly model can be translated into a tree structure that consists of a list and many binary-trees by arranging their assembly units in their position or attached faces for case-class product or in their axial direction attached faces for rotation-class product[6]. Accordingly, we define a data structure called assembly binary tree. To avoid the complexity of tree when complex product is modeled, we divide the representation of assembly binary tree into two levels: global level and local level. The global level is a standard binary tree and represents the main components or sub-assembly in the construction of product only. Local level is a CSG-like tree and represents the construction of sub-assembly. During the creation of assembly binary tree, "single step assembly discipline" must be observed. This means that only two assembly units are dealt for every assembly operator DT and every operator is based on the result of previous assembly operator. Additionally, to simplify the storage and access of assembly model, we define a type of node called sub-assembly, which are defined as assembly class module later and contain the complete information of assembly model, to make it easily be copied and reused. In an assembly binary tree, a root represents an assembly or sub-assembly of product to be designed; a non-leaf node represents a composite assembly unit or the middle result of assembly operator; and a leaf node represents a basic assembly unit. A leaf node can contain more than one parents pointers to indicate one assembly unit can be used more than one time in one assembly or by different assemblies. The main information of each node may be composed of the identifier of the type of node, the name of node, the description of functional intent, the attributes of assembly or sub-assembly, the representation of geometric pattern, its right and left binary sub-tree, and so on. The difference between leaf and non-leaf node in node description is that there is no left or right sub-tree for leaf node. However, there is a pointer to its left brother node that represents its assembly semantic relationships or functional intent. In other words, there is a pointer to left child-node or composite assembly node in right child-node or basic assembly unit node for every non-leaf node.

To realize the assembly model, we define a class called assembly module to represent the assembly binary tree as shown in Fig.3. Actually, assembly module is an abstraction of assembly model and can be used to describe the result of conceptual design. This description not only stipulates the geometric mating condition and engineering constraints between logic structure features, but the function, shape and

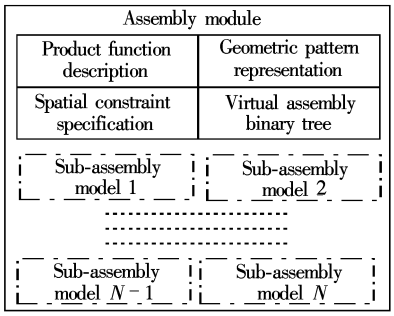


**Fig**.3 Definition of assembly module

dimensional characteristics of logic structure features in the construction of assembly or sub-assembly also. The establishment of assembly module is based on the definition of "virtual object" in software programming. Here, the word "virtual" means that all information in the object including functional description and shape representation are represented in terms of a set of variables and these variables have no means in engineering before they are instantiated. The assembly module can be formalized as follows:

$$AM_{CONCEPTUAL} = \langle BT_A, FDL, V_{FDL}, GRE, MA_P \rangle$$

where $AM_{CONCEPTUAL}$ is assembly model module; $BT_A$ is a virtual assembly binary tree; FDL is a set of functional descriptive grammar; $V_{FDL}$ represents a set of assembly constraint verb or functional verb-name pair; GRE is a set of geometric constraints, which includes not only dimensional constraints on itself of assembly unit, but the virtual dimensional constraints between different assembly units also; $MA_P$ represents the virtual component model to describe the information of function, geometric pattern and assembly relationships in the form of symbol.

After conceptual design, through the division and derivation of assembly model and the creation of virtual assembly model, the detailed design of product or its components can be made by instantiation of virtual assembly model. All components or parts can be refined simultaneously in the scope of global constraints of product to be designed in terms of definition of virtual assembly by instantiation of logic structure feature definition. This can be done ignore of when or after form-feature-based modeling set up. After all virtual assembly models have been instantiated, product can be designed by instantiation of assembly module, which is done easily through matching if the interrelationships between geometric-pattern and detailed form-features had been defined. Obviously, this process is in top-down form and can realize and maintain functional requirements.

## 3 Conclusion

This paper presents an appropriate formal

representation scheme for computer aided conceptual design and contributes to several aspects of conceptual design researches, especially to clear elementary away the obstacle that conceptual design currently lacks a regular and systematic theory. Main features of the proposed scheme in the paper are summarized as follows. (a) The definition of logic structure feature makes it possible to relate logically functional description to geometric pattern representation independent upon detailed geometric representation. (b) The establishment of a formal description of conceptual design provides a systematic method for computer aided conceptual design system development and operation. (c) The proposition of assembly module makes the design representation be based on the appropriate logical and the semantic and syntactic consistency of design can be maintained throughout the design process and the product database can be organized in a concise and structural way.

However an intelligent modeling environment, which is identical to man's realization and thought, must be set up for conceptual design before benefiting from advances in conceptual design can be expected. There are many issues to be further researched. Our main further work is to extend the existing database of logic structure feature and improve further the mechanism of data management and transmission.

## References


1 R. G. Sreenivasa, and R. D. Sriram, From symbol to form: a framework for conceptual design, *Computer Aided Design*, vol. 28, no. 11, pp. 853 – 870, 1996

2 Z. X. Sun, and F. Y. Zhang, A formalized approach to describing feature-based design methodology, *Journal of Computer Science & Technology*, vol. 13, Supplement, pp. 26 – 30, 1998

3 S. Michal, W. Christian, and S. Rainer, Functional features for design in mechanical engineering, *Computer in Industry*, vol. 23, no. 1, pp. 15 – 24, 1993

4 M. A. Wesley, P. T. Lozano, and L. I. Lieberman, A geometric modeling system for automated mechanical assembly, *IBM J. Res. Dev*, vol. 24, no. 1, pp. 64 – 74, 1980

5 K. Lee , A hierarchical data structure for representing assemblies, *Computer Aided Design*, vol. 17, no. 1, pp. 15 – 24, 1985

6 Z. X. Sun, Q. L. Ding, and T. Hong, Designation and Research of feature-based construction machinery CAD System, *Construction Machinery* (In Chinese), no. 6, pp. 8 – 12, 1996


# 面向概念设计的逻辑结构造型方法


孙正兴　　张福炎

(南京大学计算机软件新技术国家重点实验室,南京 210093)



**摘　要**　本文阐述了一种面向概念设计的逻辑结构造型方法,提出了独立于几何细节而又表达功构关系的逻辑结构特征定义和表示方式,给出了支持自顶向下概念设计的装配模板定义,并采用形式化表示形式,初步建立了计算机辅助概念设计理论.文中所提出的方法为计算机辅助概念设计系统的实现提供了初步框架.



**关键词**　概念设计,功构映射, 逻辑结构特征, 形式化表示

**中图分类号**　TP391