Conceptual Design Support System by using Object-Oriented Approach

Conceptual Design Support System by using Object-Oriented Approach

Sik PARK Choong*, Showzow TSUJIO** and Yoshisada MUROTSU**

This paper describes a development of a conceptual design support system. The system is written in an object-oriented programming language, Smalltalk-80, and all operations of the system are executed by a consistent manner, i.e., message passing. A design model is constructed of objects representing attributes and relationships among attributes, so that the model can be applied to any design objects. An interactive graphical user interface allows users easily to make and operate a design model as they like. The effectiveness of the system is demonstrated through a case study.

1. Introduction

Use of CAD systems has contributed to reduction of design efforts. Recently there are many studies of CIM (Computer Integrated Manufacturing) systems which totally manage the jobs from design planning to manufacturing. Information processing technologies provide effective methods for analysis and drawing, but they can not effectively deal with a synthesis problem which is essential in design.[1,2] Especially, conceptual design consists of a trial and error process including design tasks to assume rough structure and shape of a design object based on some dynamic and physical constraints. Consequently, it is difficult to process such a non-structured design task in a procedural manner used in conventional programming techniques.

The keys to construct a system for supporting conceptual design are: (1) A changeable design object in real world is described on a program, and (2) The interactive user interface allows users to do any design tasks.

Recently in software technologies, an object-oriented approach[3] attracts notice and some conventional computational languages also take the approach. It is a programming paradigm that describes objects corresponding to things in real world, instead of algorithm used in conventional programming. Objects are data with special functions. Communication among objects by message passing can represent various and complicated functions.

The purpose of this study is to propose the representation of conceptual design

* Graduate Student, Department of Aeronautical Engineering, College of Engineering.

** Department of Aeronautical Engineering, College of Engineering.

model and to implement the design support system using an object-oriented approach. The design model consists of attributes and relationships among attributes which can be defined by message passing from users.

This paper consists of six sections. The rest of this paper is organized as follows. Section 2 describes some aspects of conceptual design and the representation of the design model. In section 3, design tasks are represented by using an object-oriented programming. Section 4 presents a case study using the developed system and discusses the effectiveness of the system. In the final section, some concluding remarks are given.

2 Representation of conceptual design model

2.1 Aspects of conceptual design process

Design objects consist of attributes and relationships among attributes.[4] The attributes mean the character of design objects such as functions, performances, structure and shape. The relationships are given as dynamic and physical constraits, statistical and empirical formulae, regulations, etc. Such an abstraction with a design object is suitable for representing different design objects in the same manner. Consequently, a design object can be described as follows: (1) Define a design object, including description of attributes and relationships, (2) Input and calculate values of attributes, and (4) Evaluate and modify the design object.

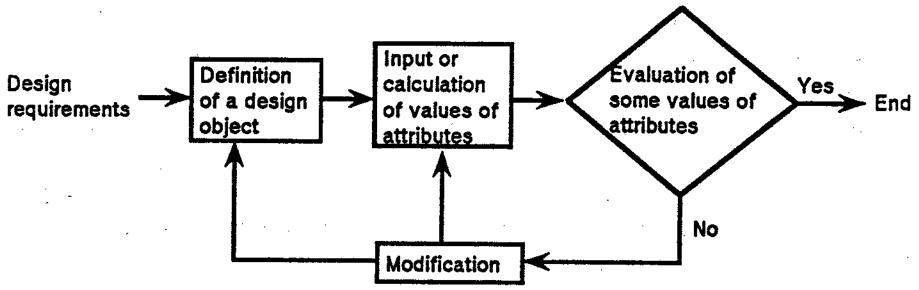
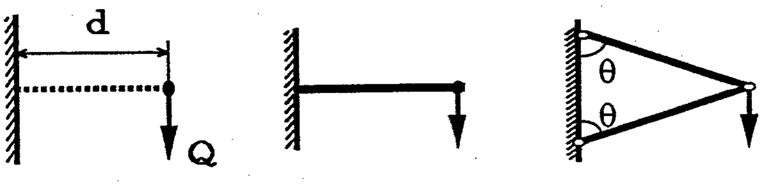

Design requirements
Definition of a design object
Input or calculation of values of attributes
Evaluation of some values of attributes
Yes
End
No
Modification

Fig. 1 Flow of conceptual design tasks

Figure 1 shows the flow of conceptual design tasks.

Since values of attributes and components of attributes may be changed flexibly according to designers, design processes are often confused. For example, consider a design of structure which transmits a load to a wall as shown in Fig.2. Two design stages are taken. The first stage is to determine a shape. The next stage is to determine its size. Some types of shape are considerd such as a beam, a truss and so on. Selection of these shapes depends on designers.



(a) Beam (b) Two member truss

Fig. 2 Types of shape in a structural design

2.2 Design model

2.2.1 Tree model

Figure 3(a) shows a tree model. The tree model does not give a concrete method to solve a design but represents the structure and the characteristics of a design object.

The requirements to describe a tree model are defined as follows:

(1) Each node is labelled its name.

(2) Only a root of a tree model is a design object. As an exception, a part of hierarchy may be described separately during a design process.

(3) The hierarchical relationships between an upper node and lower nodes are categorized into two types : (a) A relationship between the whole and the part, such as relationship between a mechanical unit and mechanical parts, and (b) A relationship between the abstract and the concrete, such as a relationship between the size and dimensions, or a length and an area.

(4) An end node has a concrete value, such as a numerical number and a symbolic. All the values of the end nodes are determined when a design is completed.

2.2.2 Network model

Figure 3(b) shows a network model. The network model represents constraints among attributes such as dynamic and physical constraints. The network model gives relationships to restrict a range of the values for the end nodes.

The requirements to describe a network model are defined as follows:

(1) A constraint is described in a form of such as an equation, an inequality, a conditional formula, etc.

(2) The network model described in an equation gives the following function:

- Only an unknown node's value is automatically calculated.
- When a value of node is changed, the corresponding values for other nodes are automatically changed.

(3) The network model described in an inequality gives a judgement whether it is satisfied or not.

The network model manages values of attributes so that designers can operate any attributes and know the change of a design model without executing a procedure to calculate the related values of the attributes. The model can allow the designers easily to perform a trial and error process to determine the values of the design variables in a conceptual design.

In this study, a design model combined with these two models is utilized as illustrated in Fig.3(c)

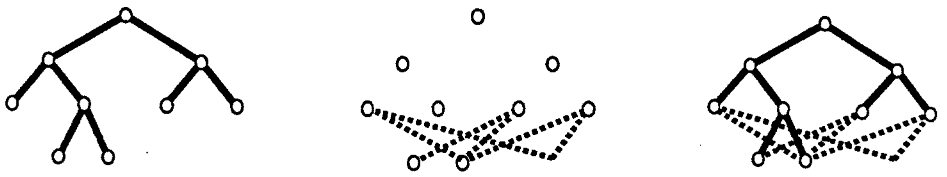
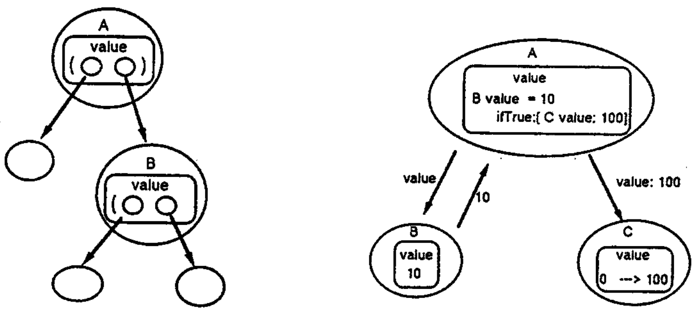

(a) Tree model (b) Network model (c) Combination of two models

Fig. 3 Design model

3. Generation and operation of attributes

The developed system is written in an object-oriented programming language, Smalltalk-80,[5,6] and all operations of the system are performed by message passing.



(a) Representation of a tree model (b) Representation of a network model

Fig. 4 Types of value

An instance of class DesignVariableObject defines an attribute of a design object as a node of a tree model. The instance holds a name of an attribute and its value. The name is a design object's name, when an instance is a root node. There are four types of values: (1) numerical, (2) symbolic, (3) nodes in the next down rank, (4) a constraint. The third type represents a hierarchy among attributes as shown in Fig.4(a). A constraint in the fourth type, is described in message expressions which define : equations, inequalities and conditional formulas, as shown in Fig.4(b). When the message expression is evaluated, related attributes are sent to messages and are changed. In the current system, users select attributes to change when a constraint is not satisfied.

Users can operate attributes by sending message as shown in Table 1 whenever they need those tasks during system running.

Table 1. List of messages

message	function
new	create an instance of DesignVariableObject
value	return a value of an instance (a value of an attribute)
value: arg	set arg to a value

In order to demonstrate the way of message passing, consider a problem to design an outer and an inner diameter of a bolt when it is subjected to 1200 kgf tensile load. An allowable tensile stress σ is 6 kgf/mm^2 . It is assumed that the outer diameter is 1.2 times of the inner diameter.

Description of message expression is shown in Fig.5, and it is also given below.

```
|d dl w|

d <- DesignVariableObject new.  ┐
dl <- DesignVariableObject new. ├------------- (1)
w < - DesignVariableObject new. ┘

d value: [ 1.2 * dl value ].  -------------------- (2)
dl value: [ (4 * w value / 3.14 / 6) sqrt ]. ------- (3)
w value : 1200.  --------------------------------- (4)

dl value    15.96  ------------------------------- (5)
d value     19.15  ------------------------------- (6)
```

Fig. 5 Message expression of a bolt design

(1) Generate instances of class DesignVariableObject corresponding to an outer diameter *d*, an inner diameter *dl* and a tensile stress *w*.

(2) Set a message expression as a constraint between the outer diameter and the inner diameter:

$$d = 1.2 \times dl$$

(3) Set a message expression for a constraint between the inner diameter and the tensile load, which is given by

$$dl = 2\sqrt{\frac{w}{\pi\sigma}}$$

(4) Input the value of the allowable tensile stress 6.

(5) Return the inner diameter 15.96.
(6) Return the outer diameter 19.15.

4. Case study

As a case study, two different structures are designed under the same condition in order to demonstrate that the system allows users to construct a design model easily.

The cross sectional areas of two different simple structures,i.e., a cantilever beam and a two member truss, are determined to transmit a given load Q to the wall away from the load point as shown in Fig.2.

The design conditions are given by :

given load $Q = 1000$ kgf, allowable tensile stress $\sigma = 50$ kgf/mm^2, distanced $d = 1$m

It is assumed that the maximum applied stress is equal to the allowable tensile stress. Relationships between cross sectional area A and other attributes are expressed as

$$A = \frac{6 \times Q \times d}{\sigma \times t} \qquad \text{(cantilever beam)} \qquad (1)$$

$$A = \frac{Q}{2 \times \sigma \times \cos\theta} \qquad \text{(two member truss)} \qquad (2)$$

where t is thickness

The system provides a user interface to define and operate a design model as shown in Fig.6. At first, users roughly define a tree model of a design object by using a mouse and a menu on the window ①. Next, they set message expressions as relationships on the window ②. and a network is made among the related attributes through the relationships. Users input a value of design variable, and if the design variable has a relationship, the related design variables are automatically calculated through the relationship.

(1) Cantilever beam

Fig.7 shows the design results. The window ① displays a defined tree model as a cantilever beam. The model is not precise, but it has a structure sufficient enough to grasp the design object. A message expression of Eq.(1) is described in the window ② A user inputs values of a length (equal to distance), a thickness= 10 mm, an allowable tensile stress and a load so that the cross sectional area is automatically calculated and displayed on the window ③.

(2) Two member truss (Fig.8)

Slope angles and two members are assumed to be the same. The design model

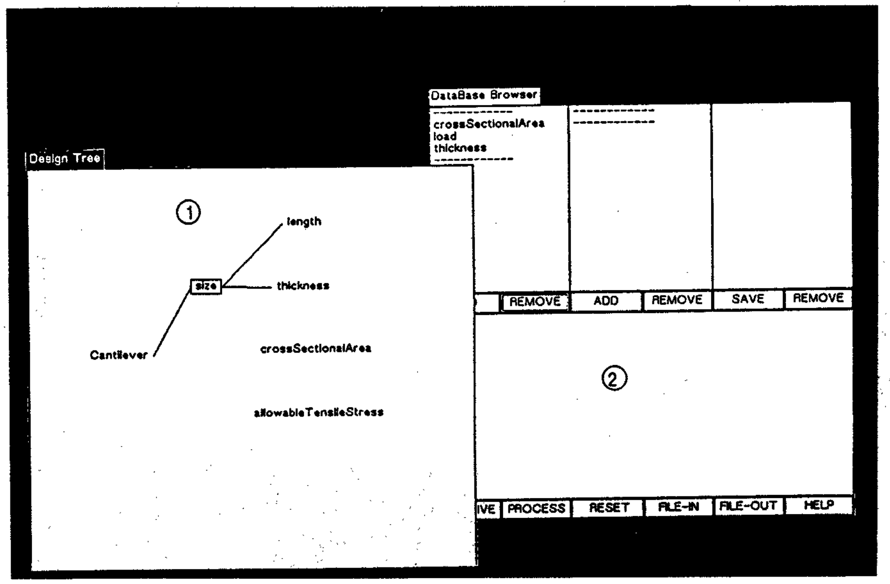
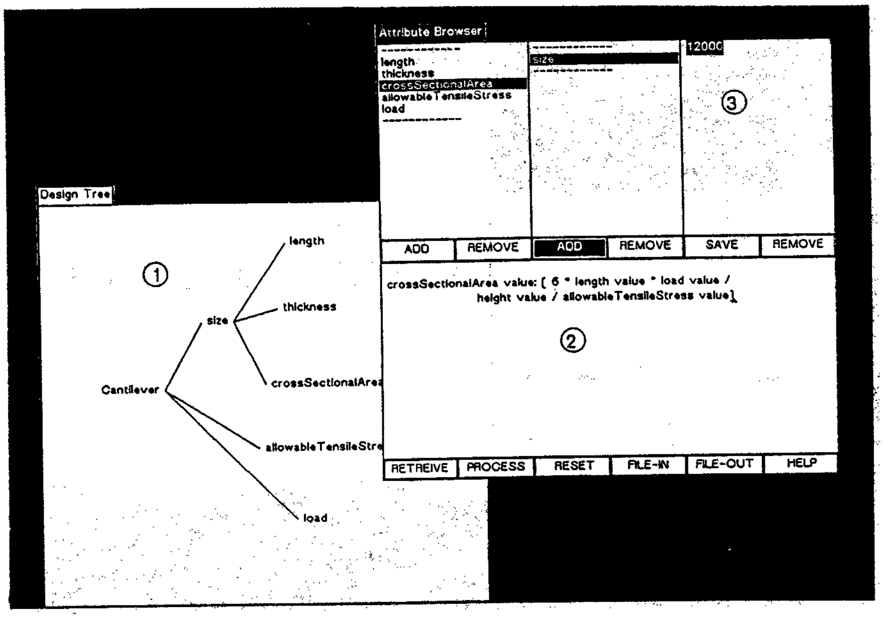
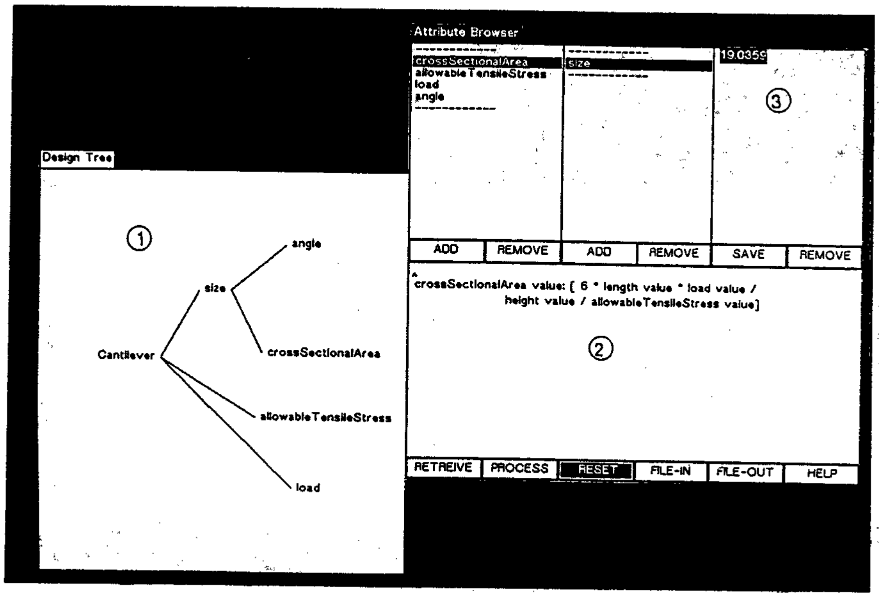


Fig. 6 User interface



```
crossSectionalArea value: [ 6 * length value * load value /
                 height value / allowableTensileStress value]
```

Fig. 7 Design results of the cantilever beam



```
crossSectionalArea value: [ 6 * length value * load value /
          height value / allowableTensileStress value]
```

Fig. 8 Design results of the two member truss

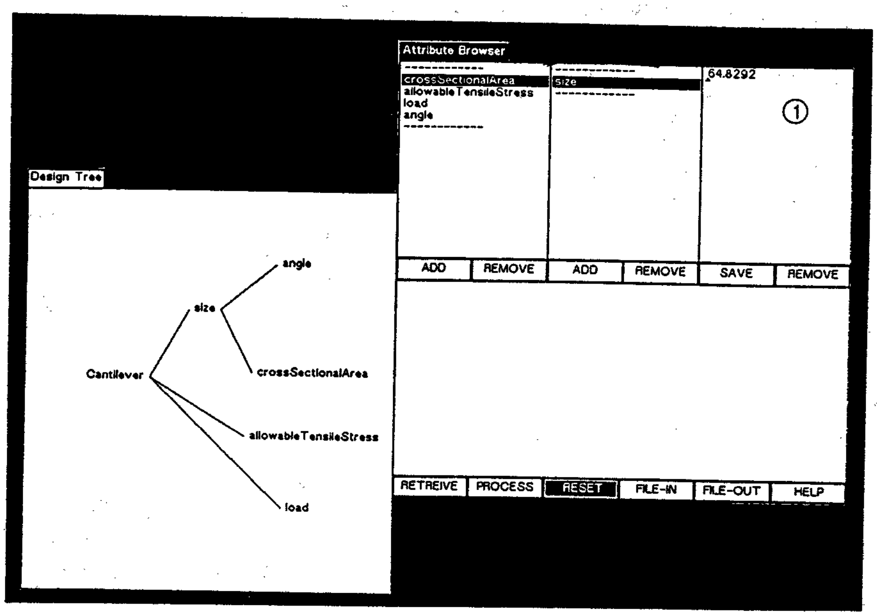


Fig. 9 Changed design results

made in (1) is changed to a design model as a two member truss by adding an angle to the window ①, deleting length and thickness from the window ①, and describing a message expression of Eq.(2) in the window ②. When the angle is $\pi/4$, the calculated cross sectional area is about 19 mm^2 as given on the window ③. Next, when the angle is changed to $\pi/6$, the cross sectional area is calculated automatically and the value is displayed on the window ① as shown in Fig.9

5. Concluding remarks

The conceptual design support system has been developed. The system flexibly executes non-structured design tasks required from users.

The results of the present study are summarized as follows:

(1) The design model corresponding to a design object in the real world is proposed to facilitate any design support task.

(2) The system is modularly written in an object-oriented programming language, Smalltalk-80, so that it easily allows modular growth for the extention of its function.

References

1) Y.Murotsu and C.S.Park, An Expert System for Instruction of Preliminary Aircraft Design, AIAA Paper No.90-3261, presented at AIAA／AMS／ASEE, Aircraft Design, System and Operations Conference, Dayton, Ohio., USA (1990)

2) K.Okada, D.Hitokoto and E.Arai, Description and Modeling of Design Process Information, Proceedings of 9th Design Symposium, pp.147-156 (1991)

3) S.E.Keene, "Object-Oriented Programming in Common Lisp", Addisson-Wesley Publishing (1989)

4) T.Ishikawa, An Apllication of Visual Programming to Attribute-Based Modelling, Proceedings of 9th Design Symposium, Japan, pp.39-48 (1991)

5) A.Golberg and D.Robinson, "Smalltalk-80, The Language and its Implementation", Addison-Wesley Publishing (1983).

6) Smalltalk-80 Virtual Image Version VI2.5, ParcPlace Systems Inc (1987)