Memoirs of the Faculty of Engineering, Okayama University, Vol.26, No. 1, pp.95-108, November 1991

The Systematic Method for Constructing the IDEA BANK Based on the EBL

Hiroshi Kawakami * and Tadataka Konishi *

(Received October 11 , 1991)

SYNOPSIS

This paper describes a method to construct IDEA BANK automatically. IDEA BANK is the data base of the "function-structure module" which is utilized in systematic conceptual design from Value Engineering perspectives. The method based on the Machine Learning EBL technique was evaluated and implemented for the IDEA BANK using SUN workstation. The practical implementation of the IDEA BANK acquisition was discussed after elucidating the problem and solution of the EBL technique in engineering design. In the IDEA BANK system, the structural features of an existing article are analyzed by hierarchically organized domain specific knowledge to yield a systematic explanation of how they function and attain their design goals. The explanation resulted in a generalized version of the Functional Diagram used in Value Engineering from which "function-structure module" can be extracted systematically.

1 INTRODUCTION

The rate of industrial development forces designers to design new articles rapidly. And to support their efforts, the so-called CAD system was developed. The CAD system achieved its success as a replacement for drafters, but is limited to only the detail design which is at the lower reaches of the design process. The CAD system should acquire enough intelligence that it can be linked directly with human intellect, in order to support the upper reaches of the design process.

But it is hard to develop a framework of knowledge information process which executes designs because of difficulties in formalizing the design process which is very creative and synthetic in nature. Implicit knowledge like common sense, inspiration, and the ability to organize the design

* Department of Information Technology

specific knowledge space which is very vast and varied; and to encompass fundamental knowledge like physical law also compounds the problem of creating the framework.

One of the differences between an expert designer and others are his volumes of design knowledge and its organization. A designer rearranges combinations of this knowledge when he encounters a new problem. As a result, we can see the rearrangements as a framework not only for the improvement of the objects in question but for the creation of novel ones as well.

In supporting the above design process, it is essential to prepare a data-base containing the design domain knowledge which is formalized as 2-tuple consisting of a function and structural attributes. We developed a method to acquire these tuple automatically which we call "function-structure module" based on the EBL(Explanation-Based Learning) method, and which can be implemented with a computer.

This paper is divided into three sections. The first section examines the Functional Analysis method of Value Engineering as a framework for the design process, with introduction of the IDEA BANK.

The second section discusses the implementation of knowledge acquisition system based on the EBL method. The problems and solutions based on this method were elucidated here.

The "function - structure module" acquisition system implemented on the multi-window environment with multi-process and graphical user interface is described in the last section.

2 FUNCTIONAL ANALYSIS

It is generally acknowledged that a detailed analysis of existing articles is beneficial not only for the improvement of the articles but for the creation of new ones as well. Value Engineering introduced by Miles [1] provides a group project called Functional Analysis. This technique analyzes an article systematically viz. how the design goal (primary function of an article) is attained by the use of subfunctions and how each subfunction is attained by the use of substructures. The analysis results in the so-called Functional Diagram (fig.1). The diagram provides members of a

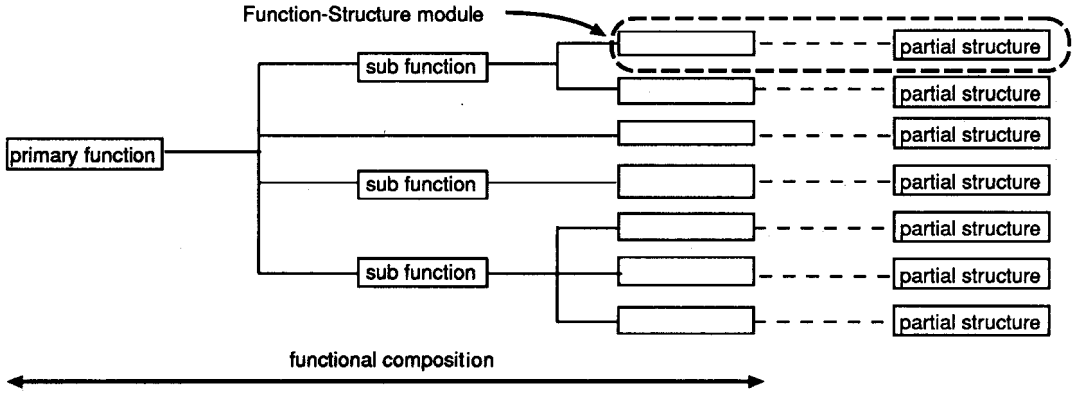


Figure 1: Functional Diagram

group project with a common view of an article.

Based on this diagram, improvements such as reduction of production costs, supplementing a new function and eliminating user claims are carried out as follows:

First, the old version of Functional Diagram and the list of terminologies which is used to represent the function in the diagram is prepared. Then, the required functions are added to the diagram arbitrarily. The required function would be a new idea which is obtained from the functional analysis of competitive companies' products, should be incorporated into the company's articles and be necessary to the user. Next, functional interference and redundancy are checked and eliminated from many evaluation factors such as user requirement or competitive requirement.

After correcting and checking the required functions and specifications, some ideas in attaining the required functions by structures must be devised. This idea can form 2-tuple which consists of a function and structures. Not all the ideas will be adopted, but all of them are preserved in the data-base called IDEA BANK for use in the design of other articles or the functional analysis of the article/articles in question.

For effective utilization of the computer facility for Functional Analysis, the computer should serve as an IDEA BANK which contains old ideas represented in a well organized common form.

This paper proposes a way to automatically acquire "function-structure module" and to construct the IDEA BANK. The proposed acquisition method was implemented in a computer system with user friendly graphical interface. The input to the system is the structural feature and primary function of an existing article. The system derives the explanation tree which shows the way primary function is attained by the use of subfunctions and structures. This explanation tree was generalized by the EBL method. Focusing on the boundary between functional representation and structural representation in this generalized version of explanation tree, the system derived the "function-structure module" which are ideas incorporated into the analyzed article. These ideas can be used generally and operatable by a computer.

3 THE KNOWLEDGE ACQUISITION MECHANISM

3.1 Organizing The Domain Theory

One of the most difficult problems which affect the construction of knowledge information processing for design is knowledge definition and organization. The design process relates varied and vast domain, and without the fundamental knowledge like the physical causal law, the concept of designed article can not be captured. This section describes the organization of the domain specific knowledge encompassing the physical causal law for measuring domain.

3.1.1 Structure (S)

Geometrical representation of structure is mostly superficial, obvious and easy. STEP(STandard for the Exchange of Product model data), which is the international standard for CAD data, defines the 3 dimensional solid, surface and wire frame models and 2 dimensional figure and shape representations without topology as the structural representation [2].

Recent CAD systems are directed towards supporting intellectual design activity instead of just being used as an excellent drafter. Therefore, structures should be represented not only by their geometrical shapes but also abstract attributes, and they should also be explicitly represented on the computer. The structural representations must be easily inferred to its functions used by the computer system systematically.

The structural representation are as follows. First, it is assumed that the "structure and attributes knowledge set S" consists of structural label, structural attributes, attribute values and structural specific knowledge (fig.2). The structural attributes contain the *is_a*, *part_of* relation, which represents implicational relationship between structural labels and provides the system with attributes inheritance. This knowledge is represented by a semantic network as shown in fig.3a, in which the nodes represent the structural label and attributes value and the arc represents structural attributes. On the other hand, it is also represented as a horn-clause which can be operated by horn-clause logic in a computer system as shown in fig.3b.

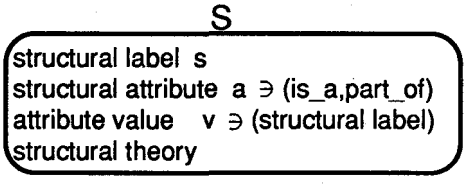


Figure 2 Structure and Attributes Space S

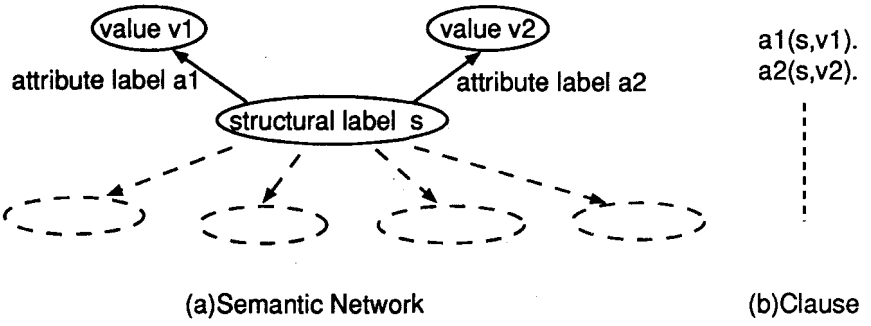


Figure 3 Knowledge Representations for S

The knowledge space S contains structural knowledge fragments (fig.4a), which are combined organically when the structural descriptions of existing articles (fig.4b) are added. The resultant combination enables us to infer implicational relation and attributes inheritance.

3.1.2 Physical Causal Law (L)

In order to support human intelligence, articles must be described by its function. For example, if one only knows a flint lighter, it is impossible to infer, from the structural description,

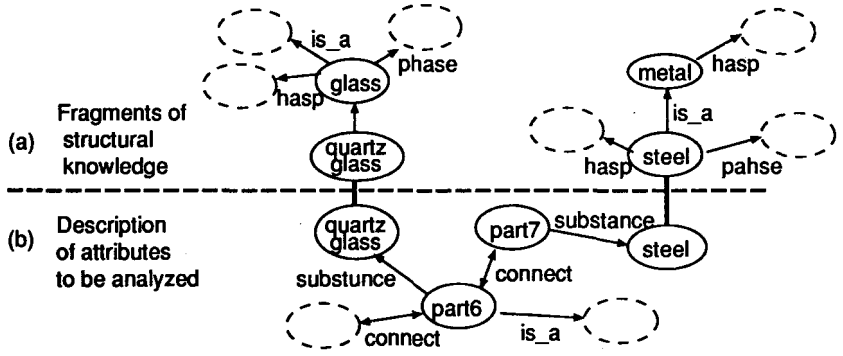


Figure 4 Self Organization of S

whether an electric discharger is also a lighter. The functional description provides the inference in the case of a flint lighter and an electric discharger. Both are described as instances of entities that function as igniter. This shows the necessity for knowledge which mediates between structural and functional descriptions. Enumerating all the relationship between each function and the structures which give rise to the function, is a simple but impractical strategy.

Function and structure should be inferred by a simple mechanism, with a general law which can mediate between them. The Physical Causal Law, which is logically well organized, is an example of an ideal mediating knowledge. Fig.5 shows the most general forms of physical causal law, which can be read as follows.

1. If the entity (structural component) X has a physical quantity P, then a set of physical properties R owned by X may cause X itself to have a physical quantity Q.
2. If the entity X has a physical quantity P, then a set of physical relations between X and Y may cause the entity Y to have a physical quantity Q.

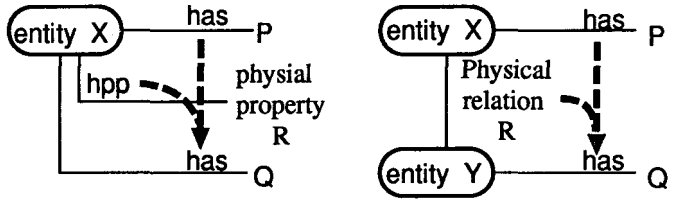


Figure 5 Most General Forms of Physical Causal Law

These general laws are instantiated to implicational law, such as "Elastic objects are under tension when they are bent", or "Solid objects will be bent when they are fixed with bending objects".

$$physical\_property(X, bend, X, tension) \leftarrow has(X, bend), hasp(X, elasticity)$$

$$physical\_relation(X, bend, Y, bend) \leftarrow phase(X, solid), phase(Y, solid), fixed(X, Y)$$

3.1.3 Function (F)

It is acknowledged that constructing a general framework which can handle functions rationally is important, but the strict definition of functions involves various difficulties. The definition here is examined in various areas.

In the area of Design Methodology, functions are regarded as "the conversion of energy, information and substance ". This idea assumes that function can be divided into subfunctions and each subfunction can be further divided until the partial structure is reached. This view of function has an affinity with the Functional Diagram of VE which represents graphically the order relation of functions of an object. Considering that each function must reach the partial structures and that physical causal law mediates their relation, functions are therefore defined as **the conversion of physical quantity**. It is represented as

$$F \ni function = f(s_1, p_1, s_2, p_2)$$

It means that structural component represented by entity s_1 has physical quantity p_1 and s_2 p_2, and they are related by a function f, which is an element of $\mathcal{F}$(which is a collection of verbs).

$$\mathcal{F} \ni f$$

VE defined standard verb vocabulary set for functional analysis [4], and it is here reorganized into $\mathcal{F}$ in our system. A single entity function is represented as

$$F \ni function = f(s, p)$$

3.1.4 Planning Knowledge (P)

Enumerating all the functions and its subfunctions is practically impossible. Hence, an approach is adapted in which the planning knowledge, that relates functions and their subfunctions, are defined in a most general form and refined by the method that will be discussed. For example, in the measurement domain, planning knowledge forms a series of functions i.e. $detection \rightarrow transmission \rightarrow transducing$.

3.1.5 Goal (G)

The goal of design is the primary function of an article, and it can be formally used to represent the function.

$$G \ni g = f(s, p)$$

3.2 A Brief Review of EBL

Explanation-Based Learning (EBL) is a technique used to acquire general concepts on the basis of a specific training example and an underlying theory of how the example is an instance of the concept. Recently, a unifying framework for EBL has been developed [3] under which many of the earlier formulations can be subsumed. In the EBL technique, the following items are inputted:

Goal Concept a definition of the concept to be learned;

Training Example a specific instance (example) of the goal concept;

Domain Theory a set of rules to be used in explaining why the training example is an instance of the goal concept;

Operationality Criterion a predicate over concept definitions, specifying the form in which the learned concept definition must be expressed, this criterion defines a set of easily evaluated predicates from the domain theory.

3.2.1 "CUP" Concept Simulation

Fig.6 shows the simulation of "cup" concept acquisition. Previously, the EBL system contains domain theory. The system requires the goal concept defined functionally as,

$$cup(X) \longleftrightarrow liftable(X) \wedge stable(X) \wedge open\_vessel(X)$$

and the structural description of a positive training example $obj1$.

$$\begin{array}{lll} owner(obj1, katuo) & \wedge & part\_of(part1, obj1) \quad \wedge \\ is\_a(part1, concavity) & \wedge & color(obj1, red) \quad \wedge \\ pattern(obj1, stripe) & \wedge & \cdots \end{array}$$

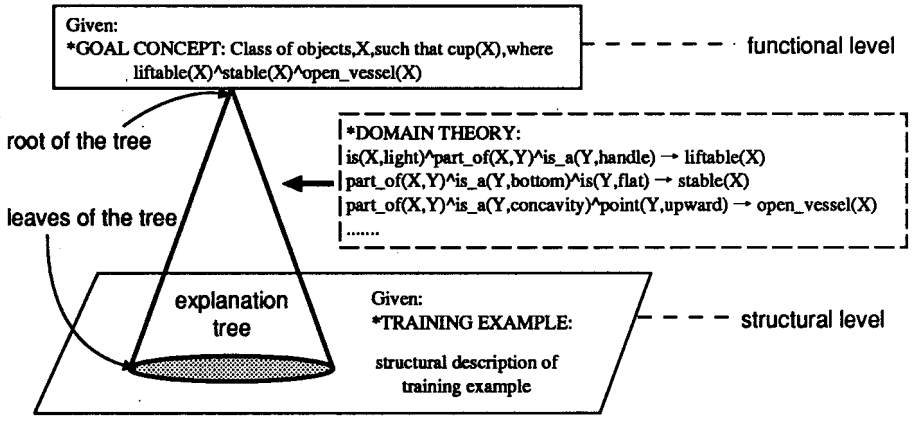
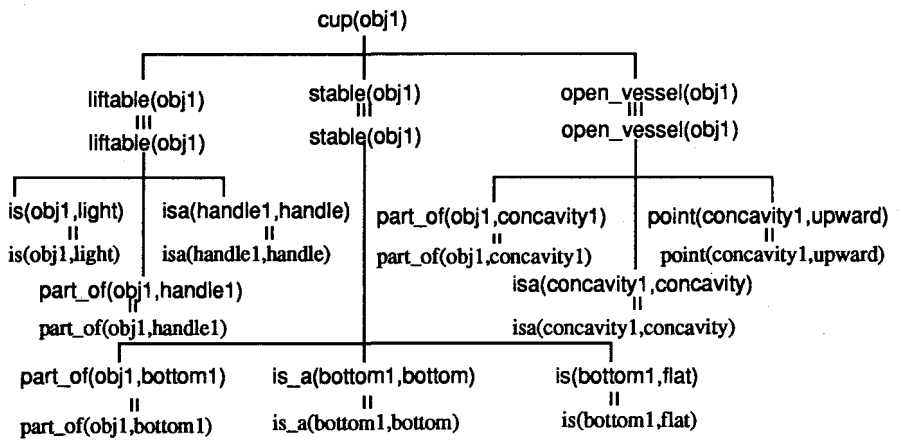
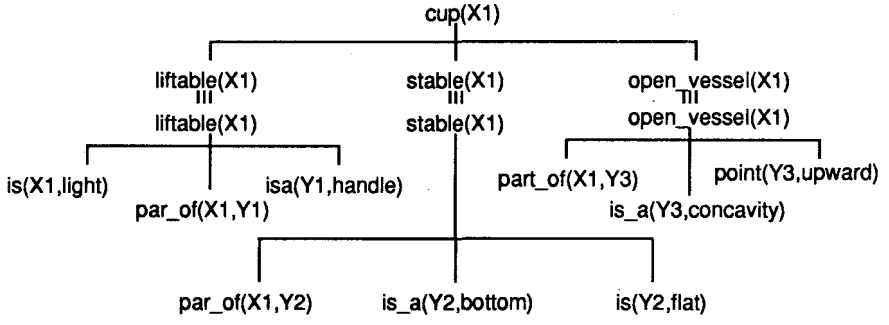


Figure 6 An Example of Explanation Based Learning

First, the system constructs an explanation in terms of the domain theory that shows how the training example satisfies the goal concept definition. Each leaf of the explanation tree (fig.7a) terminates in an expression that satisfies the operationality criterion. Assuming that the resultant concept definition is used by machining robots that have only visual sensors, they can not understand the functional command such as "make a stable object". This command is not "operational", although the structurally expressed command such as "make its bottom flat" is operational. In this case, the operationality criterion is "to be expressed structurally".



(a) The explanation tree of "cup".



(b) The generalized explanation tree

Figure 7 Explanation Tree

Next, the explanation structure is generalized by altering some constants into variables which depend only on training example. In the case of "cup", constants $handle, flat, light, \cdots$ are retained and $obj1, handle1, \cdots$ are altered to variables $X1, Y1, \cdots$ (fig.7(b)).

The system determines a set of sufficient conditions under which the generalized explanation structure holds, in terms that satisfy the operational criterion. This process results in the description of goal concept expressed in operational terms. The operational "cup" is defined (fig.7b) as an object which is light and has a handle(X1), flat bottom(Y2) and an upward concavity part(Y3).

$$cup(X1) \leftarrow is(X1, light) \wedge part\_of(X1, Y1) \wedge is\_a(Y1, handle) \wedge part\_of(X1, Y2) \wedge is\_a(Y2, bottom) \wedge is\_a(Y2, flat) \wedge part\_of(X1, Y3) \wedge is\_a(Y3, conavity) \wedge point(Y3, upward)$$

The training example contains both relevant and irrelevant information about goal concept. The EBL method eliminates the irrelevant information such as *owner*, *color* and *pattern* from

acquired concept definition.

3.3 Some Problems of EBL and Its Solutions

3.3.1 Problem of Acquired Knowledge

Generally, the deductive inference needs consistent domain theory. EBL method, based on deductive inference, also requires preparation of domain theory for processing the knowledge acquisition. Here, it faces the paradox i.e. the knowledge (domain theory) must be acquired before the knowledge acquisition processes. In other words, if consistent domain theory has been prepared beforehand, the knowledge acquisition method is essentially not necessary. On the other hand, it is impossible practically to prepare consistent domain knowledge for a vast and variant domain such as engineering design. Hence, domain theory should:

1. establish sufficiently the truth of the inference;
2. be insufficient in order to construct the explanation tree without the information about the training example, and
3. be collected easily.

For example, the following representation to define a function "transmit" was employed.

> A physical quantity P of an entity X will be transmitted to a physical quantity Q of a device Y, when there is a causal law joining X with P to Y with Q.

$$transmit(X,P,Y,Q) \longleftarrow p\_law(X,P,Y,Q)$$

But this definition is obviously over-generalized, in that we can infer that any physical law supports the function "transmit". Hence this inference needs to be guided by information about the training example. The deductive inference yields the knowledge that defines what sort of physical law supports the function "transmit". The general domain knowledge is instantiated to be acquired knowledge. It is true that the applicability of the knowledge is reduced, but it no longer needs to be guided by the information of an existing object. It thus establishes the truth of the deductive inference alone.

3.3.2 Problem for Setting the Operationality Criterion and Goal Concept

The operator of EBL system has a right to set the operationality criterion arbitrarily, but has a duty to set it before use by assuming the character of a system which will use the acquired knowledge. Strictly, these criterion should not be fixed previously, and they should also be altered according to the occasion.

The problem of the original EBL method is that it acquires a single knowledge by a single explanation tree. Explanation tree has a hierarchical structure which naturally reflects the hierarchy of the domain knowledge which corresponds to the boundaries of operationality [5]. Our system can set the goal concept to any node of explanation tree and can set the operationality criterion to any boundary. This facility enables us to extract various knowledge from a single explanation tree.

Considering the aim of this work is to automatically construct an IDEA BANK, this system should acquire the knowledge about the relationship between function and structure. This acquisition can be done by setting the operationality criterion to a part of explanation tree which corresponds to the structural domain theory, and setting the goal concept to a node of the tree which corresponds to the function. But the structural part of explanation tree is not flat. We employ the strategy to select the uppermost or lowermost level of the structural part. When the operationality criterion is set to the uppermost level of the structural part (fig.8a), the system yields the most general (abstract) structure for attaining a function. Conversely, the lowermost level (fig.8b) yields the most detailed (concrete) structure which reflects strictly the original structure of the training example.

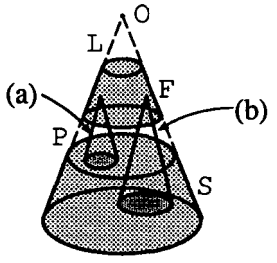


Figure 8 Two Types of Extraction Modes

4 IMPLEMENTATION OF THE SYSTEM

4.1 EBL Engine

Generation process of explanation tree is a deductive inference which searches the sufficient condition for attaining the goal concept from general theories ($\subset$ domain theory), general facts ($\subset$ domain theory) and particular facts ($\subset$ training example). It shows that the EBL method can be implemented as an augmentation of SLD-resolution theorem proving for Horn Clause Logic [6] which has been investigated and can be made using PROLOG. The EBL method slightly modifies the SLD-resolution, in performing generalizations. With the exception of leaving the leaf nodes ununified, the ordinary way of theorem proving constructs the generalized explanation tree.

The EBL needs the proof tree. In retaining the proof tree as it solves a goal, the 3rd and 4th argument of *ebg* in fig.9 are augmented for unifying the proof tree.

The next augmentation constructs a generalized proof tree. The generalized proof tree and specific one are constructed in parallel. When the specific proof has successfully terminated by unifying a leaf of proof tree with a fact, a copy of the ununified leaf is retained. The 1st and 3rd arguments of *ebg* in fig.9 are unified with a fact, but 2nd and 4th arguments remain ununified.

From the process described above, the EBL engine as a PROLOG meta-interpreter was obtained. In addition, the following facility is included in the EBL engine viz. the clauses *visu* and *v* in fig.9 which shows the generating process, and *ins* which maintain the proof tree from root to

```
ebl(A,Ga) :- gc(Ga,Gb),copy(gc(Ga,Gb),gc(A,B)),ebg(B,Gb,Bp,_,[A,x]),
             append([A],Bp,X),visu(X,_),nl.
ebg(A,_,_,_,_)              :- call(A).
ebg(A,Ga,[ex(A)],[Ga],_)    :- ex(A).
ebg(A,Ga,[A],[Ga],_)        :- dt(D,th(A)).
ebg((A,B),(Ga,Gb),P,Gp,Tr) :- ins([x,B],Tr,Tre),ebg(A,Ga,Ap,Gap,Tre),append(Ap,[x],W),
               ins(W,Tr,Tree),ebg(B,Gb,Bp,Gbp,Tree),append(Ap,Bp,P),append(Gap,Gbp,Gp).
ebg(A,Ga,[P],[Gp],Tr)  :- dt(D,th(Ga,Gb)),copy(dt(Ga,Gb),dt(A,B)),ins([[A,x]],Tr,Tre),
                          ebg(B,Gb,Bp,Gbp,Tre),append([A],Bp,P),append([Ga],Gbp,Gp).
ebg(A,Ga,[A],[Ga],_)    :- !,fail.
copy(Old,New) :- !,assert('$marker'(Old)),retract('$marker'(New)).

ins(A,[B,x],C) :- append([B],A,C).
ins(A,[x|B],C) :- append([A],B,C).
ins(A,[B|X],[C|X]) :- ins(A,B,C).
ins(A,[X|B],[X|C]) :- ins(A,B,C).
append([],Y,Y).
append([A|X],Y,[A|Z]) :- append(X,Y,Z).
visu(Tr,X) :- v(Tr,X,0),get0(_),!.
v([H|T],X,N) :- N1 is N+4,v(H,X,N),dev(T,X,N1).
v((H,T),X,N) :- v(H,X,N),v(T,X,N).
v(x,X,N)     :- assert(green),v(X,_,N),retract(green).
v(ex(H),_,N) :- tab(N),write(H),nl.
v(H,_,N)     :- tab(N),write(H),nl.
dev([],_,_)    :- !.
dev([H|L],X,N) :- v([H],X,N),dev(L,X,N).
```

Figure 9 EBL Engine represented by Horn Clause

the current part.

The 1st argument of v unifies the whole proof tree to be displayed. The 2nd argument unifies the current constructing part of proof tree. The 1st argument of *ins* unifies the part of proof tree to be inserted, 2nd argument unifies the part which has already been constructed and 3rd argument unifies the resultant insertion.

The first three *ebg*'s are clauses that unifies a leaf of proof tree with a fact. The 1st and 2nd arguments are current branch, and each clause unifies the 1st argument with built-in predicates, descriptions of a training example (ex) and the facts of domain theory (dt). In each case, only the 1st argument is instantiated and the 2nd argument remains ununified. The 3rd and 4th arguments maintain the proof tree, but only the 3rd argument is unified and the 4th argument maintains the generalized proof tree. The 4th and 5th *ebg*'s are clauses to unify branches with a rule. The 6th *ebg* will remove the self loop.

4.2 Overview of The System

Fig.10 illustrates an overview of the system which constructs the IDEA BANK automatically. The knowledge base initially contains the domain theory which is organized hierarchically into five knowledge spaces viz. Goal space, Planning knowledge, Function, physical causal Low and Structure and attributes, in which each domain theory forms most generally (upper right window in fig.11) for each space (G,P,F,L,S).

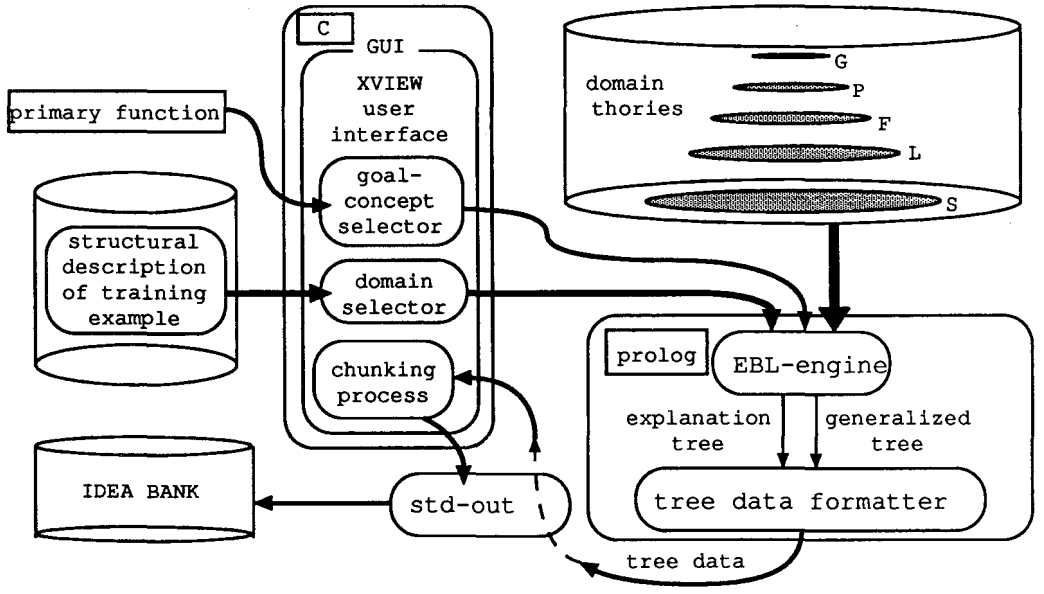


Figure 10 Overview of the IDEA BANK Acquisition System

The operator selects, from the menu, the primary function as a goal concept and the file name as a training example. The file contains the structural description of an existing article in the Clause form as shown in the right upper window of fig.11. The GUI informs the EBL engine of the file name, and forces it to load the contents of the file to the memory. Then the EBL engine starts to construct explanation tree and generalized version of it. The engine always outputs the construction process to the GUI, and the GUI displays it in the upper middle window of fig.11.

The left lower window in fig.11 shows the knowledge acquisition process. The user can set arbitrary nodes as an acquired function and as a level of structural description on the window. The system automatically collects the same level of structural nodes, and chunks the area below the selected function and above the selected structural description level into a "function-structure module". The area to be chunked is an inverted area in the left lower window. The acquired knowledge is added to the IDEA BANK.

5 CONCLUSION

This paper discussed the acquisition of "function-structure module", which forms a 2-tuple consisting of a function and a set of structures and its attributes that attains the function.

The process of the Functional Analysis method in the field of Value Engineering seems to resemble closely the internal process of an expert designer. It was supposed that the functional analysis may be one of the most hopeful alternatives to becoming a framework of systematic design process.

After examining the functional analysis, it was concluded that a computer should serve as

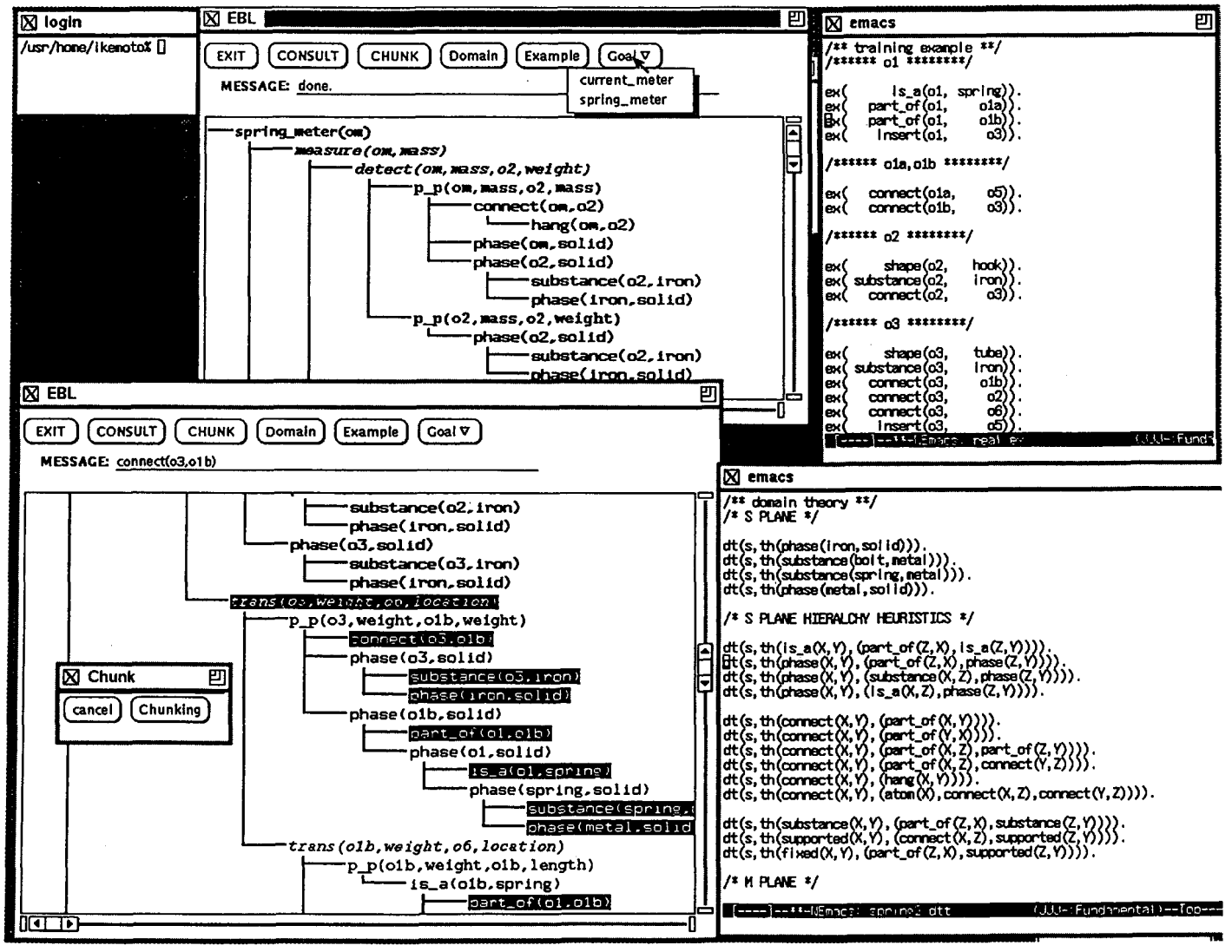


Figure 11 Graphic User Interface of The System

IDEA BANK which keeps the "function-structure module" in well organized standardized form. The designer can withdraw the idea to obtain some functions from the IDEA BANK when he/she is processing the upper reaches of design process.

Based on the EBL method which is a general knowledge acquisition method, we developed the "function-structure module" acquisition system. This system analyzes the structural feature and primary function of an existing article by domain theory, and extracts the "function-structure module".

The generating process of explanation tree is an inductive inference which involves the paradox, that is "knowledge acquisition for knowledge acquisition" . And practically, it is impossible to prepare consistent domain knowledge for a vast and variant domain such as engineering design. These problems lead the system to divide the domain theory into five spaces, and lead each domain theory to be set in the most general form. These knowledge are refined by the information of existing objects and some of them become the "function-structure module".

In the last part of this paper, the practical way to implement the knowledge acquisition system on the workstation as an application of X window system was introduced.

6 Acknowledgments

The authors are grateful for the support of the Okayama Foundation for Science and Technology. The authors also wish to thank Mr. Ikemoto and Mr. Yagi, graduate students of Okayama University, for their collaboration and programming.

References

[1] M.D.Miles: "Techniques of Value Analysis and Engineering", McGraw-Hill Inc. (1961).

[2] The Japanese Design Engineering Society: CAD Standardization and STEP, (1991).

[3] T.M.Mitchell & R.M.Keller & S.T.Keder-Cebelli : Explanation-based Generalization: A Unifying View, Machine Learning 1,(1986),47.

[4] The Japan Value Engineering Society: Selection and Classification of Functions, VE report No.49 (1981).

[5] M.S.Braverman and S.T.Russell: Boundary of Operationality, Proc. of the 5th Intl. Conf. on Machine Learning (1988), 221.

[6] S.T.Kedar-Cebelli & L.T.McCarty: Explanation-Based Generalization as Resolution Theorem Proving, Machine Learning 1(1987),47.