Toward A Reusable Knowledge Based System

Young-Dong Yoo(Visiting Professor Vanderbilt University)

Abstract

Knowledge acquisition, maintenance of knowledge base, and validation and verification of knowledge are the addressed bottlenecks of building successful knowledge based systems. Along with the increment of interesting in the knowledge based systems, the organization needs to develop a new one although it has a similar one. This causes several serious problems including knowledge redundancy and maintenance of knowledge base.

This paper presents three models of the reusable knowledge base which might be the solution to the above problem. Three models are: 1) multiple knowledge bases for a single AI application, 2) multiple knowledge bases for multiple AI applications, 3) a single knowledge base for multiple AI applications. A new approach to build such a reusable knowledge base in a homogeneous environment is presented. Our model combines the essential object-oriented techniques with rules in a consistent manner.

Important aspects of applying object-oriented techniques to AI are discussed (inheritance, encapsulation, message passing), and some potential problems in building an AI application (decomposition technique of knowledge, search time, and heterogeneous environment) are pointed out. The models of a reusable knowledge base provide several amenities: 1) reduce the knowledge redundancy, 2) reduce the effort of maintenance of the knowledge base, 3) reuse the resource of the multiple domain knowledge bases, 4) reduce the development time.

1. INTRODUCTION

Knowledge base techniques have been used in various fields. There has been an explosion of interest in applying its thnologies to a variety of fields in a real world. Many organizations have an experience with building their own knowledge based system. Some organizations have built more than one knowledge based systems. Handling of numerous information requires the knowledge based systems to have appropriate techniques.

The increment of knowledge addresses several issues such as knowledge

acquisition: update, addition and deletion of the knowledge ; and reuse of knowledge, etc. Knowledge acquisition is one of the most important tasks of the knowledge based system development. This knowledge acquisition will serve as a basis for representing knowledge in the form of the codification in a knowledge base. However, it is yet one of the primary bottlenecks of building a successful knowledge based system[1,14]. There are various techniques of knowledge acquisition, and they can be categorized into four groups: text analysis, interview analysis, behavior analysis, and machine induction.

The machine induction among the techniques could remove the bottleneck of knowledge acquisition, although full implementation is not yet feasible. The advantage of this technique is to offer the possibility of deducting new knowledge, which will be specifically useful for solving well-defined classification problems and where numerous test data is involved. However, the most efficient method of knowledge acquisition is the use of appropriately combined techniques. Yet, no single technique is perfect. In addition, knowledge acquisition requires an iterative process in order to avoid missing any important facts or rules.

In addition to the above problem, maintenance of the knowledge of knowledge based systems is an issue critical to a successful design. The maintenance of the knowledge depends on how a large volume of data or information can be managed efficiently. The current method of building a production knowledge based system is straight forward. In other words, two components of a knowledge base, facts and rules, are stored together in a single knowledge base.

However, the knowledge based systems frequently involve a large amount of data that needs periodic updating. Whenever a small update is necessary, a whole knowledge based system must be reopened, and a small update will be made. When information in a knowledge base is increased, the size of the knowledge base also should be increased in order to manage the information. It may not be a significant matter when the size of the knowledge base is small, but, the management of a knowledge base can become a serious issue when their sizes are increased[13]. In this paper, the unified term: AI application includes expert systems and knowledge based systems. The unified term is used for the applicable to both expert systems and knowledge based systems.

Another issue is the reuse of knowledge. Some of knowledge contained in different expert systems could be overlapped if the expert systems are developed for a similar purpose of works in a same organization. This decreases productivity [4,7,11]. Although most applications of expert systems consist of a single domain in a single knowledge base, the knowledge base of some expert systems is often composed of several subknowledge bases due to several reasons. One of the reasons is to facilitate the maintenance of the knowledge base. Besides, the various information and growth of its size

require expert systems to have more than one specialized domain whose contents are related to each other. That is, an expert system could be composed of several knowledge bases.

For the issues addressed in the above, we must consider a knowledge base to be reused and shared by other applications. This means that a knowledge base developed for one application can be used for other application totally or partially. In this way, we can prevent the overlapped development of knowledge bases for AI applications. The research shows that the reusable knowledge provides several amenities for an organization. The organization could save development time, reduce development expenses and manpower, etc. in terms of business viewpoints. The technical advantages are discussed in the following sections. Here, a question arose how to build such an AI application which consists of several knowledge bases in an adequate way so that the resource in the knowledge bases can be reused by other applications. This suggests the following definition for such a knowledge base:

A Reusable knowledge base: A reusable knowledge base is the knowledge base containing the knowledge which can be reused for other AI applications.

The entire knowledge in the knowledge base could be reused by others in a particular situation, while a certain part of the knowledge in the knowledge base is reused by others normally. This notion is applicable for the applications which are composed of several subknowledge bases.

For the above question, the key issue is the paradigm of knowledge representation and its programming. There are two situations that could be considered. The first is a homogeneous environment. A homogeneous environment is the system environment, whose several knowledge bases are constructed in a same paradigm. Knowledge of each knowledge base is represented in a same paradigm, and is also stored in a same structure. Several knowledge bases also are implemented by using a same language paradigm. Although the knowledge could be represented in one form and implemented in another paradigm, a single paradigm is preferred to represent knowledge and implement it. This is probably an ideal method regarding performance, consistency, knowledge base.

The second is a heterogeneous environment. Most of artificial intelligence (AI) applications have been developed for the different purpose and in a different environment. Accordingly, different representation and programming paradigm have been used. For example, one application employs the object-oriented paradigm while others are constructed using the frames. Thus, the integration of these applications is pretty complex. This method should have a mechanism for integrating multiple programming paradigms[7].

This paper considers only the homogeneous environment. The study has found that an object-oriented paradigm might be one of the solution for the above problem [2,6,7,8,9,10,12]. This paper describes how to represent knowledge and how to structure such a reusable knowledge base using an object-oriented paradigm. In addition, accompanied problems are discussed. The chief advantage of this approach, compared to other paradigms, are that objects provide a natural and flexible representation of the structural, compositional, and conceptual aspects of a variety of application knowledge domains; an object-oriented paradigm provides benefits of sharability, reusability, modularity, and extensibility of applications; and substantially improve productivity and has a good balance between power and generality[7,10].

The next section describes the basic features of an object- oriented paradigm and discusses the paradigms of object-oriented knowledge representation.

2. OBJECT-ORIENTED KNOWLEDGE REPRESENTATION

2.1 Frame

The building blocks of the representation called "conceptual objects" has imported a number of principle concepts from frame[11]. Frame represents related knowledge about a narrow subject which has much default knowledge. A frame is basically a group of the slots and slot fillers that define a stereotypical object. Frames are used frequently as a simple mechanism for guiding the process of pattern recognition. A number of special purpose languages have been designed for frames, such as FRL,KRL, KEE, HP-RL, and LOOPS, FLAVORS[5]. Frame representations can be used with rule-based programming. This concepts of a frame has been extended to object-oriented knowledge representation which provide more powerful conceptual modeling power[12].

2.2 Fundamentals of Object-Oriented Paradigm

Object-oriented languages are essentially based on the concepts of objects, classes, messages, methods and inheritance. Accordingly, an object-oriented knowledge representation should object is an entity consists of attributes and actions. The attributes describe the state of a real world object and the actions are associated with the real world object. This notion of an object enables that an object-oriented paradigm is more suitable than rules (production rules) for describing the portion of the real world of interest. In terms of an object-oriented language, an object captures both the data and the procedures associated with the real world object. The procedures or actions are referred to as methods and the

data or variables characterized the state of an object are referred to slots or attributes. The attributes of an object can only be directly accessed by the methods of the object itself (Figure 1).

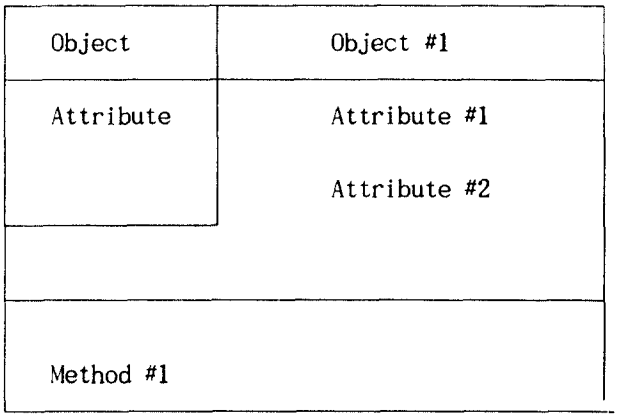

Figure 1. Structure of an Object

Methods are the procedures defined within an object. Each method has a name and a body which performs the action associated with the method name, the methods in an object can only directly manipulate data by messages. That is a method within an object is activated by a message that is sent by another object to the object containing the method. In addition, a method could be invoked by another method in the same object by a local message

Message passing provides the means for objects to communicate with each other. The protocol of message passing involves two parties: sender and receiver. When the receiver receives the message, it attempts to match the message name selector with the messages that it understands. It respond to the message by invoking the associated method if matched.

A class is a group of several objects that have similar characteristics. The attributes and methods associated with a class define the states and behavior associated with the class. Three different kinds of attributes are considered in a class: class attributes, shared instance attributes, default instance attributes[8]. The class attributes describe the characteristics of the class as a whole. Shared instance attributes hold for each individual instance of object of a class. These are the true attribute of an instance object of the class. The default instance attributes are attributes that are assigned an individual value for each instance object.

Inheitance is an important techniques in object-oriented programming which allows new object classes to be created based on existing classes. Classes with common properties are grouped into a superclass, which results in an

organized hierarchy that support generalization. The class hierarchy allows the subclasses share the attributes and methods of corresponding superclass. These attributes are shared attributes at the superclass level, and are inherited by the subclasses when their values are needed.
Methods also can be inherited by the subclasses.

Encapsulation is another important features of an object. Encapsulation is similar to the concept of information hiding. It hide the details of its internal implementation from the users of an object.

The next section discusses the features of a hybrid object-oriented knowledge representation and three models of a reusable knowledge base.

3 The REUSABLE KNOWLEDGE BASE

3.1 Hybrid Object-Oriented Knowledge Representation

Rules and objects can be used together effectively as the frame does. Facts are assertions about the state of the system and are used to fire the rules. Objects are used as storage for the facts or state of the system, and to trigger rules. It is noted that the objects represent the real world (i.e., conceptual model) while the rules correspond to the model of the problem sovler's heuristic knowledge.

There is an issue of building a system that can make common sense deductions from information that is stored in objects. This can be achieved by the technique of modularization. In addition, as the number of rules becomes very large, it is difficult to understand the rules and determine if they are consistent and correct.

However, when the rules with similar conditions are stored together in the same ruleset, two problems could occur as:

1) All the rules in the knowledge base should be considered, although some of the rules are not relevant.
2) Accordingly, maintenance of rules is difficult and require relatively more time. In a large number of rules, when a new rule is added to or deleted from the knowledge base, there could be a conflict with the existed rules.

The above problems could be solved by the concept of ruleset. Grouping of rules into smaller and understandable chunk is a ruleset. A ruleset is based on the notion of implementing function in procedural abstractions. Rules of a similar purpose can be group into a ruleset (Figure 2).

A large knowledge base can be composed of several smaller ruleset. Figure 2 shows the hierarchical structure of rulesets. A certain mechanism is needed

to control the rulesets in a knowledge base. Making a plicable rules attached to a class object in a certain order is quite useful but difficult. The strategy for the inference engine is necessary to access appropriate rules automatically when an object of a given type comes up for consideration.

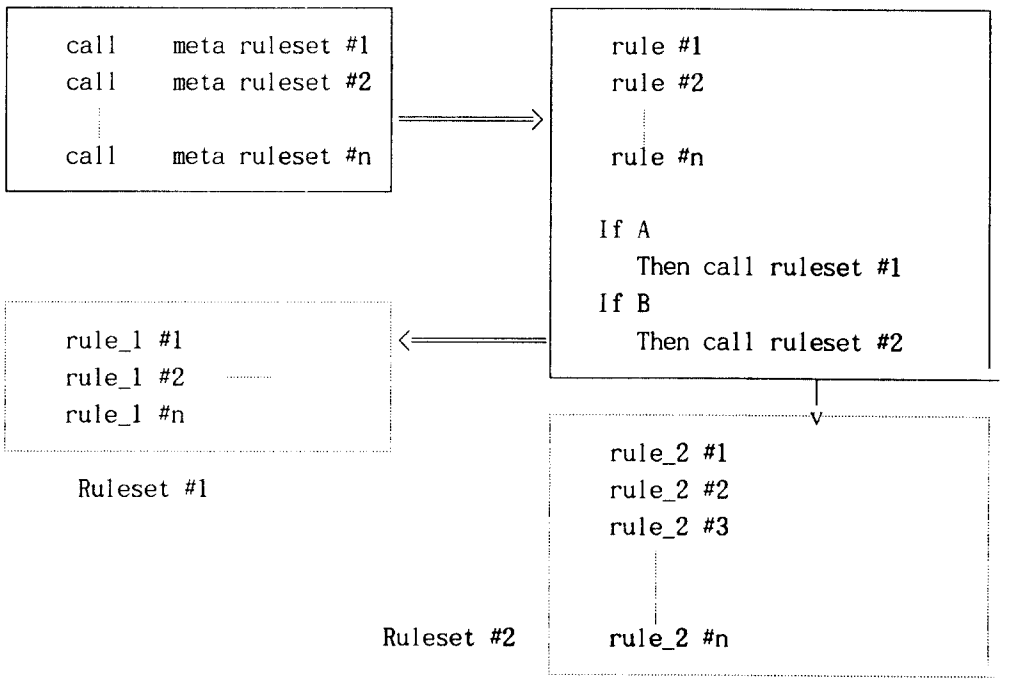

Figure 2. Hierarchical Structure of Rulesets

To select an appropriate ruleset correspond to the high level rule or the previously fired rule/functions, meta-rulesets, sets of the rulesets, are used as an intermediate controller. Meta-rulesets consist of the rulesets which have a similar purpose. Using the notion of the meta-ruleset, it is possible to avoid blind search and combinatorial explosion for a specific rule in a large knowledge base.

The steps for finding a specific rule are as follows:

1) Diagnose the problem in the mainset, and find which meta-ruleset should be linked (e.g., meta ruleset #1 in Figure 2).
2) Look up the meta-ruleset (e.g., meta ruleset #1), and decide which ruleset (e.g., ruleset #1) should be

selected to infer further.

3) Search the ruleset (e.g., ruleset #1) which is selected in the meta level of its ruleset.

In an object-oriented knowledge base, the rulesets provides advantages for the users as:

- easier consistency verification.
- can view the system at a higher level of abstraction, rules.

3.2 Models of The Reusable Knowledge Base

As discussed previously, we have found that an object-oriented paradigm could be a basis for a proper knowledge representation for building a reusable knowledge base. Three types of the reusable knowledge base are shown in Figure 3, 4, 5.
They are 1) a single knowledge base for multiple applications, 2) multiple knowledge bases for a single application, 3) multiple knowledge bases for multiple applications. The three models of a reusable knowledge base are discussed in turn.

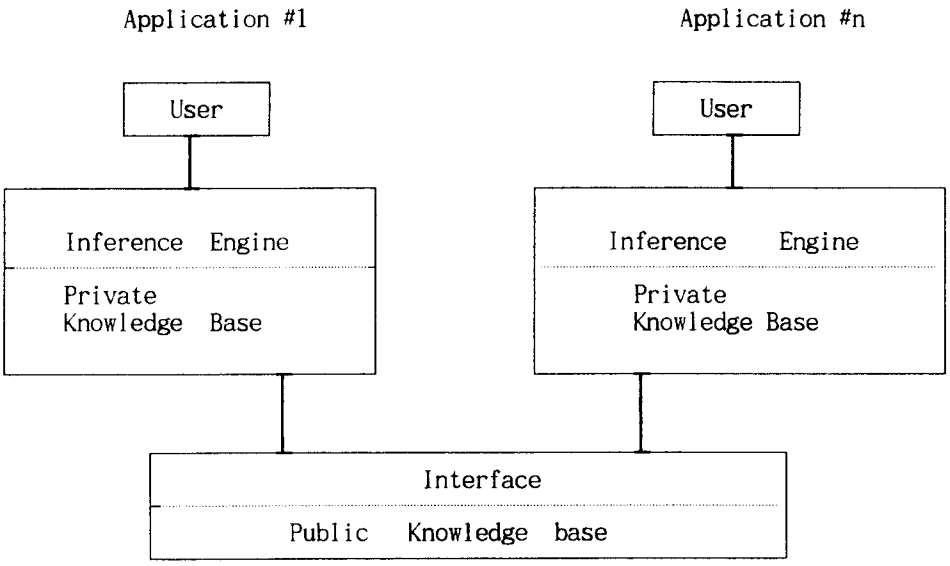

Figure 3. A Single Knowledge Base for Multiple Applications

Figure 3 shows the first model of the reusable knowledge base. The several applications existed in a same environment to share knowledge. In such an environment, each application is composed of its own inference

engine, private knowledge base and public knowledge base. The public knowledge base includes a set of knowledge which is common to and reused by several applications. Each application communicates with the public knowledge base via an interface in the public knowledge base.

The private and public knowledge base include rules and objects, and have a consistent knowledge representation paradigm: object-oriented paradigm in order to facilitate the communication among them easily. The implementation also should be done using an object-oriented programming language to satisfy the purpose of the original design.

This model is more suitable for the system environment which this case, the public knowledge base becomes a common knowledge base which includes a collection of common knowledge. The private knowledge base contains its specific domain knowledge sets. Each application, the private knowledge base in more detail, communicates with the public knowledge base to reuse knowledge through the provided interface. Although two different knowledge bases are located separately, the communication would have no difficulty since both knowledge bases are on the same paradigm.

The second model of a reusable knowledge base is shown in Figure 4. The structure of this model is fairly straightforward. This model is structured with a main knowledge base and several subknowledge base. This model is more suitable for a single application which needs a large nowledge base.

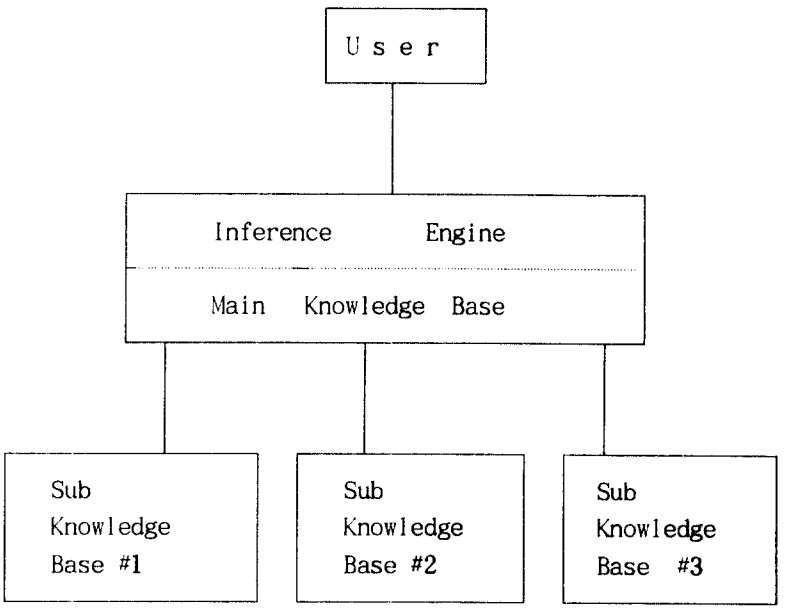

Figure 4. Multiple Knowledge Bases for A Sing e Application

The philosophy of this model is the notion of decomposition. By the

decomposition of a large knowledge base, several advantages could be taken. The major advantages are:

- The knowledge base can be maintained easily and effectively.
- The knowledge base can be reused to share its resource by other applications which might be built later.
- The Knowledge redundancy can be reduced.

Unlike the first model, the decomposed subknowledge bases act as the private knowledge bases while the main knowledge base correspond to the public knowledge base. Inference engine, first, access to the main knowledge base to find a subknowledge base contained an adequate knowledge, then the subknowledge base is linked to the main knowledge base to solve the problem. This model also is based on the use of an object-oriented paradigm. Although this model has several amenities, there is an issue in of knowledge is not simple, so it should be carefully done. The building a successful application is completely dependent on this work.

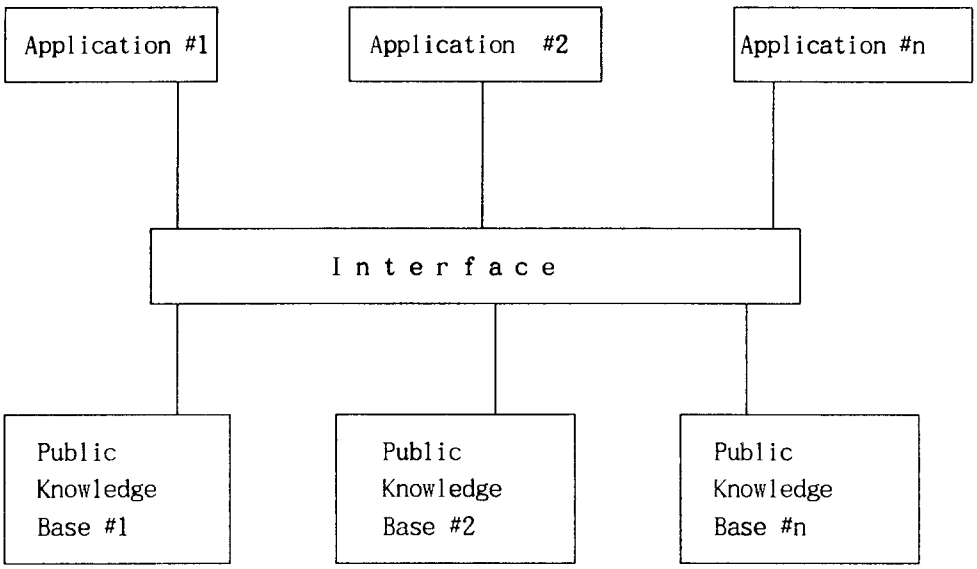

Figure 5. Multiple Knowled e Bases for Multiple Applications

The third model shown in Figure 5 is quite complex. It consists of several applications, multiple domain knowledge bases and interface. This model is a composite of the first and second model. Each application has its own particular knowledge base. In this model, each of multiple domain knowledge

base so called public knowledge base are developed in correspondence with the application. In other words, a public knowledge base is built when a corresponding application is developed. Some public knowledge base could be composed of several subknowledge bases. This makes that the resource in the public knowledge base can be reused entirely or partially by other applications.

The role of the interface in this model is important. Its main role is to search an appropriate public knowledge base which contains the required knowledge, and then, link it to the application. The search time is an issue to be considered.

Exhaustive search often causes a combinatorial explosion, which must be considered when designing an expert system. Heuristic search has been recognized as one of the possible solutions to prevent a combinatorial explosion. Heuristic searches generate elements of the search space that depend on information obtained in previous tests and on the characteristics of the goal of the search. Search space elements can be modularized based upon the precedence of the characteristics of these elements. The elements are the rules and objects . If a number of the rules and objects are modularized into several parts from one large knowledge base, the size of search space elements become smaller. By splitting a large set of the rules and objects into submodules, the size of the searching space would be reduced significantly.

every applications and public knowledge bases are based on the same paradigm(i.e., object-oriented paradigm). The structure of the applications in this model is organized hierarchically, and the object-oriented knowledge representation paradigm used for the application is based on the notion of the hierarchy. Thus, the object-oriented knowledge representation paradigm elevates the burden of the search time.

The structure of three models take advantage of a homogeneous environment as discussed in the previous sections. That is an object-oriented paradigm that provide an ability to keep track of all the objects currently in the system according to their places in the hierarchy.

4 CONCLUSIONS

The goal of this study has been to develop a model of the knowledge base that allows multiple applications to reuse the knowledge in its knowledge base. We have proposed a new approach that use an object-oriented paradigm to build a reusable knowledge base in a homogeneous environment. The primary contribution of this research is the development of three models of a reusable knowledge base using an object-oriented paradigm.

Object-oriented techniques are discussed in the context of knowledge representation paradigm. The notion of the hierarchy embedded in object

becomes the basis of building a reusableknowledge base. Some potential problems, decomposition of knowledge and search time, are pointed out and possible solutions are presented.

Our model combines the essential object-oriented techniques with rules in a consistent manner. The models of a reusable knowledge base provide several amenities: 1) reduce the development time, 2) reduce the effort of maintenance of the knowledge base, 3) reduce the knowledge redundancy, 4) reuse the resource of the multiple domain knowledge bases.

- REFERENCES -

1. C.M. Cleal and N.O. Heaten, Knowledge-Based Systems: Implication for Human-computer Interfaces. NY: John Willy and Sons, 1988.
2. H. Dai, J.G. Huges and D.A. Bell. "Knowledge Representation and Problem-Solving using Object-Oriented Paradigm," Proceedings of IEEE TENCON'93, pp. 275-279, 1993. Prentice-Hall, 1993.
4. C.F. Erick and C.K. Mao, "Modular Structures for Better Encapsulation and Reuse of Rule-based Sustems," The second World Congress on Expert Systems, 1994.
5. T. Finin, "Understanding Frame Languages, Part I," AI Expert, pp. 44-50, Nov. 1986.
6. P. Jackson, Introduction to Expert Systems. Addison-Wesely, 1990.
7. M.H. Ibrahim and S.W. Woyak, "An Object-Oriented Environment for Multiple AI paradigms," Proceedings of TAI'90, pp. 77-83, 1990.
8. W. Kim, Introduction to Object-Oriented Databases. MIT press, Cambridge, 1990.
9. T. Koyama, "Building a Common Knowledge BAse for Internal Medicine," Proceedings of KJCES'93 pp. 876-888, 1993.
10. J.A. Lewis, S.M. Henry and D.G. Kafura, "An Empirical Study of the Object-oriented Paradigm and Software Reuse," OOPSLA'91, pp. 184-185, 1991.
11. Y. Takaoka, M. Sakamoto and et. al, "Towards Re-use of knowledge: A Study of Methods Regarding Substation Fault Recovery Operation Support System," The second World Congress on Expert Systems, 1994
12. E.R. Tello, Object-Oriented Programming for Artificial Intelligence. Addison-Wesely, 1989.
13. Y.D. Yoo, "An Expert Training System Loosely Coupled to External Database Systems," Proceedings of Developing and Managing Expert System Programs'91, pp. 24-28, 1991.
14. D.A. Waterman, A Guide to Expert Systems. Reading, MA: Addision-Wesley, 1986.