Expert Systems Based on Object Model - Systematic Knowledge Modeling and Model-based Reasoning-*

Haruki UENO** and Yasumasa OOMORI**

Abstract : In this paper we propose the concept and methodology of object model-based expert systems. A systematic modeling method and a model-based reasoning scheme are shown. The object model is useful to represent the knowledge about structures and behaviors which are known as one of deep knowledge representations for complex structured objects. The key points of the object model-based knowledge system are that 1) representation of knowledge and interpretation of it are separated from each other so that the representation could be used for solving a variety of problems via associated interpretations, and 2) the knowledge model for a specific object can be smoothly constructed by means of a generalized scheme of the object. Rule-based systems, *i.e.*, surface knowledge systems, are convenient to represent expertise, but their performance and flexibility are limited. Advanced knowledge-based expert systems to solve complex problems need to handle both types of knowledge. The object model is represented by means of a two dimensional hierarchy where a whole-parts (PARTOF) relation is represented in a horizontal hierarchy and an abstract-specific (ISA) relation in a vertical hierarchy. A trouble-shooting expert system combining IF-THEN rules and an object model for electro-mechanical devices have demonstrated interesting characteristics. The long time goal of this work is to provide a useful technology for developing a multi-objective shared-use engineering knowledge base which will become a key technology in the 1990's. The system was developed using the frame-based knowledge engineering environment ZERO.

Key Words : Artificial intelligence, Knowledge base, Expert system, Object model, Deep knowledge, Reasoning, Trouble shooting

1. Introduction

Experiences of developing expert systems in the 80's tell us that in order to achieve high performance and flexibility in knowledge-based problem solvings features to represent and manipulate deep knowledge are required. This is understandable because human experts have knowledge about basic theories and principles extracted from education and research, in addition to expertise obtained through long time experience in a specific domain. In another case we know that design sheets and design specifications are very useful for engineers who are trying to find failures within systems, or are trying to understand texts describing complex devices. It should be noted that these types of knowledge which are referred to as "*deep knowledge*" are common and domain independent, and can be shared by a variety of problems.

Knowledge about principles and theories and knowledge about structures and behaviors are known as typical deep knowledge (*e.g.* Davis, 1982; Ueno,

1983). In this paper we deal with the second type of deep knowledge, *i.e.*, knowledge about structures and behaviors, which is useful to manipulate complex structured systems especially in engineering (Ueno, 1983 and 1986). We call such a representation model an "*object model*".

Knowledge-based systems which have the features to be able to manipulate these kinds of knowledge may be applicable to solving a variety of problems in a variety of domains in a shared use. Surface knowledge which has been applied to most currently available expert systems is dedicated to specific interpretation in specific problems within a narrow domain, and therefore results in limited flexibility and performance.

Because of its nature deep knowledge is objective and problem independent, while surface knowledge is subjective and problem dependent. Domain experts use both kinds of knowledge in actual problem solving. It should be stressed, however, that even if objective knowledge is used in solving problems, it is represented in a same scheme, *i.e.*, IF-THEN rule, as that of subjective one in traditional expert systems. In this scheme it is difficult to extract the flexibility. If we could maintain objective knowledge as a multi-use knowledge base, it will become easier to develop expert systems in the future.

* Orally presented at International Symposium on Mineral Exploration: -The Use of Artificial Intelligence, 1990 (ISME-AI '90, Tokyo and Tsukuba)

** Department of Systems Engineering, College of Science and Engineering, Tokyo Denki University, Hatoyama, Saitama 350-03, Japan

Problems to be solved by knowledge-based approaches are divided into two contrasting groups which are analysis-oriented problems such as trouble shooting and decision making, and synthesis-oriented problems such as system design and scheduling. Deep knowledge systems are able to solve both kinds of problems just like human experts do.

From the point of view of data structure for knowledge representations, a structured system includes both abstract-specific [ISA] and whole-part [PARTOF] relationships by its nature because it has many similar elements and consists of a number of sub-systems and subsub-systems. The ISA relation is used to represent an elements hierarchy, and is naturally represented by the frame-based knowledge representation model (e.g. Minsky, 1975). In the frame model, the inheritance control for attribute information is a key function in order to maintain the knowledge base and is naturally applicable in the ISA hierarchy. The PARTOF relation is used to represent the structure of the system in combination with a semantic network for representing local relationships of their elements, and is not appropriately represented in usual frames, because the inheritance is not applicable to this relationship by its nature.

Therefore we need a powerful and flexible knowledge representation framework for manipulating the object model. We have solved this problem by means of an extended frame formalism, where an object is represented in a two dimensional hierarchy and the inheritance is extended to apply to a set of slots, i.e., elements of frames, so that the knowledge about structures and behaviors are inherited vertically as a set (Ueno, 1986). In an ordinary frame system, inheritance is controlled on a slot-by-slot basis. By this extension, template schema of structured objects are inherited naturally to their subclasses and instances through an extended ISA hierarchy.

In this paper we discuss the concept of object model, the systematic modeling method, a way of model-based reasoning and an example of applications. The application systems are implemented in frame-based knowledge engineering environment, ZERO, which is extended to handle the extended frame representations for the object model-based system (Oomori *et al.*, 1990). The extension was accomplished so that the concept and methods of the object model can be realized smoothly. The ZERO system is written in the COMMON LISP language and is available on a variety of computer systems, including mainframe computer systems and workstations.

2. Motivations

The idea of knowledge modeling on a complex structured system is not new. Many advanced expert systems using object modeling have been developed and reported already. However, ways of modeling which have been proposed and applied are not systematic enough in terms of such requirements as acquisition, maintenance and multi-use of knowledge. Knowledge acquisition is a bottleneck through the history of Knowledge Engineering (Feigenbaum, 1978). Multiple use of knowledge will be an issue in the 1990's. We have been trying to develop a systematic method for modeling of objective engineering knowledge to meet these requirements. Providing advanced technologies to solve these problems is the major motivation and long-time goal of our efforts. Already we have proposed the idea and method of an object model-based approach and developed several experimental expert systems (Oomori and Ueno, 1990; Kato *et al.*, 1988).

Most currently available expert systems based on surface knowledge can work only for solving the specific problem in a narrow domain. Engineers who have the knowledge about principles and structures of a target system can solve a variety of problems related to the system using the same knowledge. One of the issues of the deep knowledge should be the multiple interpretability of the knowledge. Most of this type of knowledge is objective and can be shared. We need a knowledge modeling scheme which allow us to use a knowledge base for multiple purposes in shared use.

From observation of expert engineers' tasks, we can obtain useful hints. Some of the hints listed below affected in our way of knowledge modeling and model based problem solving.

- 1) A person who has the knowledge of one system can easily obtain the knowledge about similar systems.
- 2) Engineers can solve a variety of problems using the same knowledge.
- 3) Objective knowledge seems to be much more important than subjective knowledge, especially in engineering.
- 4) In problem solving complex structured systems, engineers seem to control a focus-of-attention along the structural and functional connections of the system.
- 5) Structures and functions seem to have strong relationships.
- 6) Objective knowledge and subjective knowledge are linked together and used in solving problems in actual situations.

We have been trying to develop a powerful and

flexible technology for future expert systems by keeping these observations in mind.

3. Concept of Object Model

By “object” we mean the objective of problem solving by means of knowledge-based approaches. It includes a physical object, a conceptual object and a structured system in general. We call a knowledge representation model of such objects an “object model” (Ueno, 1983; 1986). For example, the computer model of a computer system is the object model when the computer system is a target of problem solving. We believe that the features to manipulate the object model will become very important for advanced knowledge-based expert systems in the 1990’s.

Problems to be solved by knowledge-based approaches are divided into two groups; synthesis-oriented problems and analysis-oriented problems. The former includes medical diagnosis, trouble shooting and policy selection, and the latter includes CAD, scheduling and design. Constructing the model of an object by assembling suitable elements under the specific constraints to meet a set of given requirements is the issue in the synthesis-oriented problem. For analysis-oriented problems also, the object model should play an important role although the AND/OR tree-based approach is used in most traditional expert systems.

We know this from the following example. A person who has knowledge about the principles and the structures of one thing can solve a variety of problems related to it, using the same knowledge. For example, a person who knows the principles and the structures of a computer system can express what a computer is to another person, can understand a picture of a computer, can find some failures in the design specifications of a computer, and can do some trouble shooting for a computer. Of course, the quality and the performance of these abilities depend on the quality and quantity of the knowledge the person has about a computer.

On the other hand, currently available knowledge-based expert systems based on surface knowledge such as a Production System, where a unit of knowledge is represented in an IF-THEN rule, can work only for solving a specific problem on specific aspects in a narrow domain (Davis, 1982; Hart, 1981). This is because these expert systems have limited flexibility and therefore the ability of the system is much inferior to that of human beings. From the point of view of a problem solving method, it seems unlikely that a human being treats an analysis-oriented problem and a synthesis-oriented problem as different. Instead, it

Fig. 1. Two-layer knowledge base (a) and single layer knowledge base (b).

Fig. 2. Ideal view of a multi-use system based on the concepts of object model.

seems true that a human can use common knowledge for a variety of problems in various ways according to situations he is faced with.

The structure of a knowledge base to meet these circumstances should have a two-layer hierarchy as shown in Fig. 1(a) where the lower layer consists of the object model, i.e., deep knowledge, and the upper layer surface knowledge, i.e., a set of production rules. A usual knowledge base has a flat structure as shown in Fig. 1(b). The two-layer hierarchy model may be used as a multi-use knowledge base as shown in Fig. 2, where the knowledge organized as an object model is used for example as, trouble shooting, natural language interfaces, graphic interfaces and generating IF-THEN rules with attachment of associated sub-modules as shown in the figure. In this scheme the object model should maintain objective and problem-independent knowledge about an object. Problem-dependent knowledge, i.e., surface knowledge, should be maintained by sub-modules according to the specific problem. We believe that this scheme is a candidate to achieve a large scale shared-use

knowledge base.

What kind of data structures is the most suitable to represent the object model? From the point of view of representing the model, a semantic network seems to be widely applicable as a common representation model. In addition, a large complex object should be represented in a hierarchy, because the number of similar modules (parts) must be included in the model. We think that the object model can be appropriately represented in an extended frame-based representation formalism. The frame model has flexible and powerful functions which come from useful features such as hierarchical data structures with attribute inheritance functions, a combination of declarative and procedural knowledge, and availability of a variety of data types. Therefore, we think that a suitable extension of the frame model should meet our requirements.

In the following sections we describe how the object model can be represented in the extended frame-based formalism. We have proposed a systematic modeling procedure for describing complex structured objects (Ueno, 1986).

4. Systematic Modeling Method for Complex Structured Objects

4.1 Modeling Policy

A structural system has two aspects which are structures and behaviors. We have to properly combine these aspects in systematic knowledge modeling. The basic idea of our modeling is that each behavioral attribute is strictly connected to each associated substructure of an object. By structure we do not mean a physical structure but a functional structure of an object. For example, in the case of a computer system we are concentrating on the functional configuration of the system. We do not consider the physical attributes in basic modeling. As a consequence, our modeling policy is as follows: The skeleton of the knowledge structure is determined according to the functional structure of an object. The behavioral attributes are attached to associated components of the object. In this chapter we discuss the structural aspect of knowledge modeling within the frame-based knowledge representation schema with some extensions in inheritance control.

4.2 Modeling Method on Structure

In general, a complex system consists of a number of subsystems, which in turn consist of a number of sub-subsystems, and can be decomposed until every subsystem becomes a primitive component. We

need a method of representing such a system so that the complex system can be organized and manipulated systematically. By using a simplified automobile system, a concept of representing the object model in the extended frame formalism and an example of representations will be shown followed by a discussion about suitability.

The key point of knowledge modeling in the frame-based formalism is that the concept of abstract-specific (ISA) relations can be applied through the whole representation. This modeling method meets the requirements about the usage of the inheritance control which is the fundamental concept of the frame-based representation model. This means that knowledge models which are not based on the ISA relation have no benefit from using the frame-based model. It should be noted again that a complex structured system can only be represented by a combination of abstract-specific relations and whole-part relations. We need a suitable extension in order to handle complex structural objects within the framework of the frame-based formalism. This is the issue.

At first we discuss how the structural attributes of complex systems can be modeled. Fig. 3 shows an example of a representation of a simplified automobile system based on our modeling concept. For simplicity of discussion this model consists of four major subsystems, which are body, engine, steering and electronics subsystems. In addition, the body subsystem includes several sub-subsystems such as a floor, an engine room, a trunk and a door which consists further of a window and a handle sub-subsystems. Similar decompositions are possible for other subsystems such as the engine. By means of this decomposition the automobile system can be decomposed until each module becomes a primitive part.

Fig. 3. An example of a two-dimension three-layer representation of a complex structured object.

The modeling should be done as follows. On the left hand side the principal objects starting with the physical object are defined by means of ISA relations, *i.e.*, "Physical Object-Automobile-Legend-Legend-A". This vertical relation plays the role of the skeleton in the modeling, and is referred to as a skeleton hierarchy. On the right hand side the structures of the associated principal objects (defined in the skeleton hierarchy) are defined by means of HASPARTS relations from left to right in a horizontal hierarchy. Each node in the hierarchy represents a frame. The HASPARTS relations (and semantic network relations) are embedded within the parent system frame as discussed below.

It is very important to note that although the structures are defined around the principal objects at each layer in the ISA hierarchy, a set of structural definitions can be inherited from upper to lower layers. That is, the structural descriptions of Automobile are inherited by Legend, which are inherited by Legend-A. In this case, there is no need to re-define the structures at the second and third layers. If a different subsystem is used in Legend then redefinitions for that subsystem is needed. In this case, the information to be inherited about the structural descriptions from Automobile to Legend-A is partly replaced. It should be noted that the components are represented by means of ISA hierarchies, the structures are represented by means of HASPARTS hierarchies, and the structural information can be inherited naturally by the extended inheritance control.

4.3 Knowledge Modeling in Frames

The structured object can be represented in the ZERO formalism in the following way. Each frame has a unique frame name and consists of a set of slots as components which store attribute data associated to the frame. Each slot has a unique slot name within a frame and consists of five items, *i.e.*, a slot name, a data type specification, slot value(s), demon(s), and an option. In addition to three system slots, *i.e.*, the slots installed in the system (ISA, HASPARTS and RELATIONS slots), users can describe a variety of representations by means of user slots, *i.e.*, slots defined by users themselves. Data type specifications used in the following examples and their meanings are:

FRAME : the slot value is a frame name,
 i.e., a pointer to another frame,
 FLIST : a list of frame names,
 PREDICATE : a set of predicates,
 VALUE : a numerical value,
 TEXT : a text,

FRAME: Automobile		
ISA	FRAME	PhysicalObject
HASPARTS	FLIST	Body, Engine, Steering, Electro
RELATIONS	PREDICATES	(INCLUDE Body Engine), (TOGETHER Engine Electro), - - -
LENGTH	VALUE	NIL
- - -	- - -	- - -
- - -	- - -	- - -
FRAME: Legend		
ISA	FRAME	Automobile
LENGTH	VALUE	465 cm
- - -	- - -	- - -
- - -	- - -	- - -
FRAME: Body		
ISA	FRAME	PhysicalObject
PARTOF	FRAME	Automobile
HASPARTS	FLIST	Floor, E.Room, T.Room, ...
RELATIONS	PREDICATES	(FRONTOF E.Room Floor) - - -
COLOR	TEXT	NIL
- - -	- - -	- - -
- - -	- - -	- - -

Fig. 4. Frame description for three components in the extended frame schema.

LISP : a Lisp function name,
 i.e., an attached procedure's name.

An attached procedure is procedural knowledge associated with a specific frame.

Fig. 4 shows an example of frame-based representations for three objects from Fig. 3 in the ZERO formalism, which are Automobile, Legend and Body. The last frame is an example of non-principal object and includes a PARTOF relation to express that "Body is a part of Automobile". As shown in the figure, the Legend frame does not include the HASPARTS and RELATIONS slots because these attributes can be inherited from its parent frame Automobile by means of the extended inheritance control. This means that not only the attributes defined within the Automobile frame but also every attribute defined within the subsystems and sub-subsystems are automatically inherited through associated ISA links respectively.

4.4 Systematic Modeling Procedure

From the above discussion, we can obtain the modeling procedure for the systematic knowledge representation of complex structured objects which conform to the concept of the object model:

- 1) The principal objects, *i.e.*, top level objects, of a target system must be represented in a skeleton

hierarchy by means of ISA slots.

- 2) The components are organized in an abstract-specific hierarchy using ISA slots, while the structures are organized in whole-part hierarchies using a combination of HASPARTS, PARTOF and RELATIONS slots.
- 3) The HASPARTS slot has a list of subsystem element names as slot value. The semantic relations between the elements are described by a set of predicates which is stored as the value of the RELATIONS slot of the frame.
- 4) Additional attributes of each component are described by means of user definable slots of an associated frame.

This modeling method is applicable under the extended frame-based knowledge representation environment. It should be noted that this knowledge representation formalism does not include heuristics, that the modeling does not need expertise, and that the knowledge is application independent.

In conclusion a typical data structure to represent the object model should be a three-layer two-dimension hierarchy as shown in Fig. 5. Vertical hierarchy represents ISA relations. Horizontal hierarchy represents PARTOF relations from left to right. At the top layer, the generalized model of an object is represented. The object model for specific object is represented at the middle layer. It should be noted that this representation can be generated from the generalized model. Because most substructures and parts of the specific object are the same as, or similar to, those of the generalized model, definition of a specific object becomes easy. This is understandable from watching an engineer who is trying to obtain new knowledge using the knowledge which he already has, as discussed in the motivations section of this paper. For example, persons who have knowledge of computer systems will be able to easily obtain the knowledge about a new computer system. Instance knowledge at the bottom layer is generated at the time of specific problem solving which is shown in the example.

5. An Application: Trouble Shooting System Combining Surface and Deep Knowledge

In this chapter we show an example of object-model-based knowledge systems.

5.1 Outline of System

As shown in Fig. 6, this expert system consists of a production system, *i.e.*, surface knowledge system, and object-model system, *i.e.*, deep knowledge system (Oomori and Ueno, 1990). The production

Fig. 5. Three-layer two-dimension hierarchy to represent object model

Fig. 6. An overall view of a hybrid trouble shooting expert system combining a rule-based system and a model-based system.

Fig. 7. A function-based system structure of a focal plane shutter system and proper control signals.

system is used to represent expertise in rules, and rule-based reasoning is applied to choose the most likely hypothesis or to decide the unit to be focused on which the model-based reasoning will be applied. The reasoning manager controls the overall reasoning.

Fig. 7 shows a function-based system structure of a focal plane shutter system, a sub-system of a reflex camera system, and its proper control signals. As shown in the figure, the system receives two external input signals, one from a shutter dial unit and the other from a release switch unit, and gives an output to a mirror system. The signal from the release switch activates the action, that is, the control unit sends a signal to A-magnet which controls A-screen (front screen) by switching the signal from "on" to "off", and receives a signal from A-screen for confirmation. A similar signal is sent to B-magnet to control B-screen (rear screen). By the difference of timing of the two signals, the shutter interval is appropriately controlled according to the input value from the shutter dial unit. The mirror is moved just

before screen movement. Therefore, this is an example of a simple computer-based sequential control system.

5.2 Knowledge Representation

Fig. 8 shows the hierarchical knowledge representation of the system based on the concept of the object model. As shown in the figure, the hierarchy consists of three layers of representation. Camera, Camera 1 and Camera 1-A are principal objects, and they constitute the skeleton hierarchy. Their structure is represented by HASPARTS relations at each layer. At the top layer the general structure of the camera system is described. This figure shows that Camera consists of Body, Shutter Mechanism and Mirror, Shutter Mechanism consists of Shutter Unit, Release Switch and Shutter Dial, and Shutter Unit consists of Shutter, Shutter Control and Electronic Control, respectively. At the middle layer, the specific focal plane camera named "Camera 1" is described. As stated before, if the structure and behavior of the focal plane shutter system is different from those of

Fig. 8. The hierarchical representation of the focal plane shutter system shown in figure 7.

frame name : SHUTTER-UNIT frame type : CLASS		
A-kind-of	FRAME	ROOT
DDescendants	FLIST	(F-SHUTTER)
Description	STRING	*UNDEFINED*
Created-by	STRLIST	("KATOU" "19-Aug-87 11:36:28")
Modified-by	STRLIST	("OOMORI" "21-Aug-87 12:21:35")
HASPARTS	FLIST	(SHUTTER SHUTTER-CON ELECTRO-CON)
PARTOF	FRAME	SHUTTER-MECHANISM
RELATIONS	PREDICATES	((LEFTOF SHUTTER-CON SHUTTER) (LEFTOF ELECTRO-CON SHUTTER) (UNDER ELECTRO-CON SHUTTER-CON))
INPUT-FROM	FLIST	(RELEASE SHUTTER-DI)
OUTPUT-TO	FLIST	(MIRROR)
BEHAVIOR	BRULES	((IF (RELEASE SHUTTER-DI) NIL THEN NIL (MIRROR)))

frame name : F-SHUTTER frame type : CLASS		
A-kind-of	FRAME	SHUTTER-UNIT
DDescendants	FLIST	(F-SHUTTER-A)
Description	STRING	*UNDEFINED*
Created-by	STRLIST	("KATOU" "19-Aug-87 11:37:01")
Modified-by	STRLIST	("OOMORI" "24-Mar-88 17:11:52")
HASPARTS	FLIST	(A-SCREEN B-SCREEN A-MAGNET B-MAGNET CONTROL-UNIT)
PARTOF	FRAME	F-SHUTTER-MECHANISM
RELATIONS	PREDICATES	((LEFTOF CONTROL-UNIT A-SCREEN) (LEFTOF A-MAGNET A-SCREEN) (LEFTOF B-MAGNET B-SCREEN) (UNDER CONTROL-UNIT A-MAGNET) (FRONT A-SCREEN B-SCREEN) (FRONT A-MAGNET B-MAGNET))
INPUT-FROM	FLIST	(E-RELEASE SHUTTER-DI-1)
OUTPUT-TO	FLIST	(F-MIRROR)
BEHAVIOR	BRULES	((IF (E-RELEASE SHUTTER-DI-1) NIL THEN NIL (F-MIRROR)))
VALID-INPUT-VALUE	PREDICATES	((VOLT E-RELEASE 5) (VOLT SHUTTER-DI-1 (1 2 3 4 5 6 7 8 9 10)))
VALID-OUTPUT-VALUE	PREDICATES	((VOLT F-MIRROR 5))

frame name : F-SHUTTER-A frame type : INSTANCE		
A-kind-of	FRAME	F-SHUTTER
DDescendants	FLIST	NIL
Description	STRING	*UNDEFINED*
Created-by	STRLIST	("OOMORI" "24-Mar-88 17:14:46")
Modified-by	STRLIST	("OOMORI" "18-Jul-88 15:43:45")
HASPARTS	FLIST	(A-SCREEN-A B-SCREEN-A A-MAGNET-A B-MAGNET-A CONTROL-UNIT-A)
PARTOF	FRAME	F-SHUTTER-MECHANISM-A
INPUT-FROM	FLIST	(E-RELEASE-A SHUTTER-DI-1-A)
OUTPUT-TO	FLIST	(F-MIRROR-A)

Fig. 9. Detailed frame descriptions for same unit at different layers in the extended frame schema.

the general camera system, the sub-descriptions of the focal plane shutter mechanism named F.Shutter Mechanism are redefined as in the figure. At the bottom layer, the instance object of Camera 1, named Camera 1-A, is defined, and the related frames are defined also. Because the values of the instance frames are the same as those of the class frames between Camera 1 and Camera 1-A, all associated attribute data are inherited by means of the extended inheritance control features.

Fig. 9 shows three examples of detailed frame descriptions from Fig. 8. This figure shows two things; one is how the knowledge about structure and behavior are described within a frame, and the other is the relationship of the slot descriptions between frames in a hierarchy. The slots for structural description and their roles are:

A-kind-of to describe the abstract object of this

object in the ISA hierarchy,
HASPARTS to describe a set of directly connected components of this object,
PARTOF to describe the parent component of this object in the PARTOF hierarchy, and
RELATIONS to describe the relations between the components which are defined in the HASPARTS slot of this frame.

The slots for describing the knowledge about behaviors and their roles are:

BEHAVIOR to describe the behavior of this object,
INPUT-FROM to describe a set of elements from which the input is received,
OUTPUT-TO to describe a set of elements to which the output is sent,
VALID-INPUT-VALUE to describe a list of valid input value ranges for each input, and
VALID-OUTPUT-VALUE to describe a list of

Fig. 10. An example of displays of model-based trouble shooting session of the hybrid expert system.

valid output value ranges for each output.

The behavior of an object, *i.e.*, unit, is described by means of a behavior rule in an IF-THEN rule form as in the following.

IF Input-conditions, Operations
THEN Actions, Outputs

The semantics of the rule is "IF Input-conditions and Operations on the unit are satisfied THEN the unit takes Actions and produces Outputs to the connected units". A behavior rule gives the constraint on a target object as a black box. If a focused object satisfies the constraint, then we suppose that it is working well. In another case, we suppose that it may include faulty components. An example of behavior rules for the F-SHUTTER unit is as follows: "((IF (E-RELEASE SHUTTER-DI-1) NIL THEN NIL (F-MIRROR)))". The semantics of the rule is "IF the input signals from the E-RELEASE and SHUTTER-DI-1 units are included within the valid ranges THEN the proper output signal is sent to the F-MIRROR unit". Operations and actions are not defined. It is possible to define multiple behavior rules for a single unit.

The valid ranges for input and output are defined by means of a set of predicates as shown in the

figure. Although F-SHUTTER is the direct descendant, *i.e.*, child, of SHUTTER-UNIT in an ISA hierarchy, the substructure of F-SHUTTER is different from that of SHUTTER-UNIT. Redefinition of structure and behavior about F-SHUTTER is shown in the middle frame. No such redefinition is seen in the bottom frame, because the data can be inherited from F-SHUTTER.

5.3 Reasoning Method

Inference is accomplished by combining rule-based reasoning and model-based reasoning. The rule-based reasoning which is done using a set of IF-THEN rules stored in the rule base of the production system is used to identify the focusing unit of the camera system. AND/OR tree-based backward reasoning is applied for this purpose. Subsequently, the model-based reasoning starts from the focused unit which has been identified by the rule-based reasoning.

The strategy of the model-based reasoning is very simple. This is accomplished by repetitively checking the validity of input and output values of the focused unit by evaluating the behavior rules. The validity is checked by means of the behavior rules and

a set of valid range definitions which are attached to the frame associated to the unit. There are four cases; 1) both the input and output values are within the valid value ranges, 2) both the input and output values are outside the valid value ranges, 3) the input value is outside and the output value is inside, and 4) the input value is inside and the output is outside. In the first two cases the system concludes that the focused unit has no failures. The third case does not make sense. In the last case, the system assumes that the focused unit includes the failure. If the focused unit has no subunits, or nothing is described about the subunits, then the system concludes that the failure occurs in this unit and terminates the reasoning. If there are descriptions about the subunits of the focused unit in the last case then the system continues the reasoning by changing the focus of attention to one of the subunits using the information from the associated behavior rule of the associated frame. The reasoning continues so that the most primitive unit is detected by this repetition. We think that this way of trouble shooting is quite similar to that of human experts.

Fig. 10 shows an example of the model-based trouble shooting session. The system is working on an ELIS workstation. In the upper-right window the interaction by rule-based reasoning is shown. This reasoning concluded that "Failure occurred within Shutter Control Unit". According to this conclusion, the reasoning manager switched to the model-based reasoning as shown in the lower-right window. The model-based reasoning begins with the focused unit which is Shutter Control Unit as shown in the middle-right window. The current focused unit is "rear screen control magnet" as shown in the upper-left window. In the lower-left window the black box representation for the unit is shown, which is produced from the behavior rules and the definition of the valid ranges.

6. Other Applications

In addition to the trouble shooting system discussed in the prior chapter, several expert systems based on the object model have been developed at our laboratory.

ALPUS (Ueno, 1987; 1989a; 1989b; 1990), is a knowledge-based program understanding system. In this system, algorithm-oriented programming knowledge is organized as an object model in the HPG-graph formalism. The system detects local logical bugs in a program, reasons programmer's intentions from buggy code, and provides advice not only for fixing the code, but also for letting him

understanding the error. The ALPUS system is a subsystem of knowledge-based intelligent programming environment, INTELLITUTOR, for novice programmers.

In the trouble shooting system for a petroleum refinery plant (Kato *et al.*, 1988), a refining system for propane and butane is modeled. Although this is a chemical system with continuous and dynamic behavior the modeling method and reasoning strategy are similar to those of the focal plane shutter system. The difference of the modeling is that in this system the constraints, *i.e.*, behavior rules, are defined by means of a set of pressure, flow and temperature specifications of petroleum for each unit. The constraints are represented in mathematical equations.

In the trouble shooting system for micro-computer devices (Yamamoto *et al.*, 1990; Ueno *et al.*, 1991), multiple-level modeling is applied. That is, the system is modeled at an unit level, a functional module level, a logical gate level and a circuit level. At each level, the object model is represented by combining the logical configuration model and the physical configuration model. These representations are linked as an unified knowledge base. The behavior rules to express constraints of specific unit is represented in logical equations.

In an intelligent human interface system (Nagasaki and Ueno, 1988), to assist a person who is trying to find the most suitable CAD data from a CADAM database features for understanding graphical information have been achieved by means of model-based object comprehension.

7. Conclusion

In this paper we have presented the concept of the object model, the systematic modeling method based on the concept, and applications. The modeling is suitably achieved by means of the extended frame-based knowledge representation formalism which has been implemented in ZERO. By embedding the PARTOF relations within the ISA relations model generation (*i.e.*, knowledge acquisition), model-based problem solving procedures become simple.

The methodology shown here is useful in manipulating complex structured systems. There are several ways to combine rule-based knowledge and the object model. One of them was shown in the paper. It is also possible to attach other types of deep knowledge to the model.

The model-based approach will be applicable to solve a variety of knowledge-based problems such as trouble shooting, system designing, drawing understanding, text understanding, and so on. Some of the

problems have been examined at our laboratory as shown in the paper. Our long term goal of this research is to provide useful and powerful technologies for developing and maintaining a multi-objective shared-use knowledge base.

Object model-oriented knowledge acquisition seems to be quite useful to solve the bottleneck in the knowledge-based approach. Because the modeling procedure is systematic and needs less expertise, it may be possible to achieve automatic knowledge acquisition from design documents in the future. It is also possible to generate IF-THEN rules from the model. These topics should be our next targets.

We have obtained the idea from watching and considering engineers who are solving engineering problems. In this sense, the object model is a cognitive model of engineers.

Acknowledgement : The work shown in this paper was supported in part by Science Research Foundation of Ministry of Education, NTT Human Interface Institute and Research Institute for Technology, Tokyo Denki University. The author would like to thank members of Ueno laboratory of Tokyo Denki University for system implementations and discussions.

References

- Davis, R. (1982) Expert systems: Where are we and where do we go from here. *The AI Magazine*, 3
- Feigenbaum, E. (1978) The art of artificial intelligence -Themes and case studies of knowledge engineering. *Proc. NCC78*
- Hart, P.E. (1981) Direction for AI in the eighties. : *SIGART*, 79, 11-16
- Kato, H. Oomori, Y. and Ueno, H. (1988) Trouble shooting expert system for petroleum refinement plant based on object model. *JSAI SIG-KBS-8803-1* (in Japanese)
- Minsky, M. (1975) A framework for representing knowledge. : In *The Psychology of Computer Vision* (P.H.Winston, ed.), McGraw-Hill.
- Nagasaka, Y. and Ueno, H. (1988) Comprehension of mechanical drawings for CAD. *Proc. 2nd Annual Conference of JSAI*, 381-384, Tokyo (in Japanese)
- Oomori, Y., Sotou, A., Nakamura, K., Oohama, H. and Ueno, H. (1990) User extensible Frame-based knowledge representation language and its evaluations. *EIC AI90-75*, Kyoto (in Japanese)
- Oomori, Y. and Ueno, H. (1990) Hybrid troubleshooting system based on object model-object modelling and model-based reasoning-. *Journal of Japanese Society of Artificial Intelligence*, 5, 604-616 (in Japanese)
- Ueno, H. (1983) Concept of an object model. *Digest IPSJ, Fall*, 1105-1106, Tokyo (in Japanese)
- Ueno, H. (1986) Object model for a deep knowledge system. *Proc. 7th ICMCDM*, 792-800, Kyoto
- Ueno, H. (1987) Knowledge based intelligent programming environments-from the point of view of program understanding-. *Information Processing (Jyohoshori)*, 28 1280-1296, (in Japanese)
- Ueno, H. (1989) INTELLITUTOR: A knowledge based intelligent programming environment for novice programmers. *Proc. COMPCNO89 Spring*, 390-395, San Francisco
- Ueno, H. (1989) A knowledge based program understanding for intelligent programming environment for novice programmers. *Proc. bit'89*, Berlin
- Ueno, H. (1990) ALPUS: A program understanding system by means of algorithm-based programming knowledge. *Proc. PRICAL' 90*, 693-698, Nagoya
- Ueno, H., Yamamoto, Y. and Fukuda, H. (1991) Knowledge modeling and model-based problem solving-towards a multi-use engineering knowledge base -. *Proc. ASE 91*, Boston (in press)
- Yamamoto, M., Oomori, Y., Fukuda, H. and Ueno, H. (1990) An object model of keyboard unit and a prototype system for trouble shooting. *Proc. 4th Annual Conference of JSAI*, 507-510, Tokyo (in Japanese)

(Received 28 April 1991; accepted in revised form 24 May 1991)

要 旨

対象モデルに基づくエキスパート・システム
— 体系的モデリングとモデル推論 —

：上野晴樹・大森康正

本論文は、対象モデルに基づくエキスパート・システムの概念と方法論を提案している。また、体系的なモデリング法とモデル推論の枠組みについても述べている。対象モデルは、深い知識表現法の1つであり、複雑な構造をもつ対象の構造や機能に関する知識の表現に有用である。対象モデルに基づく知識システムのポイントは、1) 様々な問題の解決に用いるために、知識の表現とその解釈が互いに分離されていること、2) 対象に関する一般化された知識表現から特定対象の知識モデルを容易に生成できることである。浅い知識表現法であるルールシステムは、経験則の表現には便利であるが、能力と柔軟性に限界がある。複雑な問題の解決には、両方の知識が必要となる。対象モデルは2次元の階層構造によって表現されるが、全体一部分 (PARTOF) 関係が水平方向の階層で、抽象—具体 (ISA) 関係が垂直方向の階層で表現される。実験用に関与された IF-THEN ルールと対象モデルを組合せた簡単な電子機械システムを対象とする故障診断システムは、興味深い特性を示した。本研究の長期的目標は、90年代において重要となると思われる、工学知識に関する多目的共同利用型知識ベースの開発に有用な技術の開発である。なお、本論文で述べているシステムは、フレーム型知識工学環境 ZERO を用いて開発された。キーワード：人工知能、知識ベース、エキスパートシステム、対象モデル、深い知識、推論、故障診断。

(1991年4月28日受付, 1991年5月24日受理)