

Knowledge Acquisition Based on Representation (KAR) for Design Model Development

Jhyfang Hu and Jerzy W. Rozenblit

Abstract

In recent years, knowledge-based systems have become one of the most popular approaches for solving engineering problems. Along with the rising complexity of the problem, knowledge management, acquisition, representation, inferencing, and refinement become increasingly difficult. Conventionally, tasks of knowledge acquisition and representation are accomplished separately and sequentially. Knowledge engineers are responsible for preparing question patterns and setting up personnel interviews with domain experts for knowledge acquisition. All acquired knowledge is then manually interpreted, verified, and transformed into a predefined representation scheme. In this chapter, a new methodology for knowledge acquisition, termed Knowledge Acquisition based on Representation (KAR) is presented. Instead of treating acquisition and representation separately and sequentially, KAR deals with acquisition and representation in an integrated manner. All knowledge acquired with KAR is verified and transformed into a representation scheme which is ready for inferencing. The major objectives of KAR are to facilitate the knowledge acquisition process, to increase its reliability, and to reduce the development cost of the model bases.

1 Introduction

Knowledge acquisition is the main bottleneck in the design of expert systems. Several methodologies have been introduced to help eliciting knowledge from experts. The most common method for knowledge acquisition is an interview [Hart 1985]. In face-to-face interviews, experts are asked questions and are expected to give informative answers. All details of the interviews must be recorded for further manual analysis and conversion so that essential knowledge can be extracted and translated into a representation scheme. Problems have been discovered with the interview approach. For example, experts are often unaware of specific details

of the particular problem; experts are unable to spell out their knowledge; knowledge engineers are unable to query all the essential knowledge; experts misunderstand knowledge engineer's questions because of different interpretations of the terminology. All the above situations may result in unnecessary, duplicate, and conflicting knowledge.

Some of the interview technique problems are resolved by a more structured approach termed the protocol analysis [Waterman and Newell 1971]. With the protocol analysis, experts comment on specific examples from a problem domain. For example, experts may look at a specific design example and comment on the question: "Why is the design good or bad?". This is different from the interviewing approach which may tackle the same problem by asking question patterns such as "What made the design good or bad?". Often, it is easier to comment on a specific example in the protocol analysis rather than to answer the general questions in the interviewing process. In protocol analysis, knowledge is extracted from detecting general patterns, e.g., experts may always examine one particular characteristic first.

Computer induction [Ritchie 1984] is another technique to deal with knowledge acquisition. With induction, experts provide a set of examples of cases (called training sets) together with attributes considered in design decision making. Then a program is applied to induce rules from those training sets. The quality of the induced knowledge depends on the selection of training sets, attributes, and the use of induction algorithms.

Much of the difficulty in knowledge elicitation lies in the fact that experts cannot easily describe how they view a problem. This is essentially a psychological problem. Kelly viewed a human being as a scientist categorizing experiences and classifying his own environment. Such a description is very suitable for an expert in his knowledge domain. Based on Kelly's personal construct theory [Kelly 1955], the repertory grid technique [Boose 1988] was developed for knowledge acquisition. Given a problem domain, experts build up a model consisting of elements and constructs which are considered relevant and important. The constructs are similar to attributes except that they must be bipolar (e.g., good/bad, true/false, strong/weak). Elements are analogous to examples in induction. The grid is a cross-reference table between elements and constructs. For example, in acquiring knowledge for programming evaluation, experts may build up the grid table with a number of typical programs (elements) and attributes (constructs) such as modularity, testability, portability, meaningful variables, and readable layout. Each square of the grid table is then filled with a quality value or index. Finally, the quality of programs is determined by experts based on the summing quality index.

Although methodologies such as interviewing, protocol analysis, observation, induction, clustering, prototyping [Waterman and Newell 1971, Ritchie 1984, Hart 1985, Kessel 1986, Gaines 1987, Gaines 1988, Olson

and Rueter 1987] have been proposed to help in eliciting knowledge from experts, none of them is commonly accepted. Different problem domains may require different acquisition strategies.

To reduce errors (e.g., misinterpretation of the interviewed data) caused by the human intervention, a more efficient and reliable approach for acquiring knowledge is to automate the elicitation process based on a certain representation scheme which will completely and efficiently denote all the domain traits and encompass all essential knowledge. Knowledge acquisition guided by the structure of a knowledge representation is termed Knowledge Acquisition based on Representation (KAR) [Hu et al. 1989]. To distinguish it from conventional approaches, KAR exploits the structural nature of a representation scheme to motivate acquisition activities.

2 Requirements for Design Modelling Representation Schemes

Before selecting a good representation scheme to serve as the basis of KAR, let us first examine requirements for a good design knowledge representation scheme. Reviewing the common traits in system design, structured techniques are used to reduce the complexity of the design process. Common design traits found in modern design approaches are:

Hierarchy: The use of hierarchy involves dividing a system into subsystems and then repeating this operation on the subsystems until the complexity of subsystems is at an appropriate, or desired, abstraction level.

Modularity: Modularity helps designers to reduce the complexity of system models and clarify an approach to a problem. A modular design facilitates flexibility and future modifications. The modular design approach also aids in team design.

Regularity: Employing regular structure to simplify the design process is gaining its popularity. Regularity can exist at all levels of the design hierarchy. As seen in computer system design, uniform transistors are used at the circuit level; identical gate structures are employed at the logic level. At a higher level, a multi-processor system is designed with identical processors [Weste and Eshraghian 1985].

In general, a good knowledge representation scheme for system design applications must be able to denote the above properties within its representation structure. Furthermore, the knowledge representation scheme must be able to capture both static and dynamic knowledge of the system such as:

Static knowledge:

General properties of objects.

Taxonomy/decomposition of objects.

Rules for design synthesis.

Dynamic knowledge:

Model descriptions of objects.

Procedures for generating design alternatives.

Procedures for design validation and evaluation.

A good knowledge representation scheme should also facilitate knowledge management, efficient knowledge acquisition, inferencing, and refinement. Finally, knowledge reflected by the representation scheme must be transparent to domain experts, knowledge engineers, and system users.

To qualify for KAR, a knowledge representation scheme must satisfy the following requirements:

1. The scheme is able to query (or generate question patterns automatically) knowledge about the domain.
2. The knowledge acquired from domain experts can be automatically and directly translated into an internal representation which is ready for inferencing. The internal representation of knowledge is transparent to domain experts and system users.
3. The structure of the knowledge representation scheme provides efficient and systematic mechanisms for knowledge organization, inferencing, and refinement.
4. The knowledge representation scheme must possess axioms or operations to detect conflicting information, to remind that essential knowledge may be missing, and to eliminate repeated or unnecessary knowledge.

The increasing demand on the quality of knowledge-based systems has resulted in knowledge representation becoming a major topic in AI research. Although various schemes [Quillian 1968, Minsky 1975, Schank and Abelson 1977, Nilsson 1981, Zeigler 1984, Shastri 1988] have been introduced to help representing and managing knowledge, none of these conventional representation schemes satisfies all requirements that qualifies a representation scheme for the KAR approach. We have augmented the system entity structure [Zeigler 1984] into an integrated and entity-oriented knowledge representation scheme, termed the Frame and Rule-Associated System Entity Structure (FRASES) [Hu et al. 1989]. FRASES is a scheme that combines concepts of the system entity structure, frame [Minsky 1975], and production rules [Newell and Simon 1972]. By exploiting the reasoning flexibility provided by production rules, the efficiency in representing declarative knowledge offered by frames, and the visibility and hierarchy supported by the system entity structure, FRASES is a powerful and efficient scheme for managing domain knowledge supporting design model development.

3 Structure of FRASES

FRASES is a superclass of the system entity structures that encompasses the boundaries, decompositions, and taxonomic relationships of the system components being modelled. All axioms and operations defined originally for managing system entity structures are also present in FRASES representation [Hu et al. 1989, Rozenblit et al. 1989].

Each entity of FRASES signifies a conceptual part of the system which has been identified as a component in one or more decompositions. Each such decomposition is called an aspect. In addition to decompositions, there is a relation called specialization. It facilitates representation of variants for an entity. Each specialization variant inherits properties and substructures from the parent entity to which it is related.

A typical example of FRASES for representing a tightly-coupled multi-processor system is shown in Figure 1. As shown in the figure, each entity of FRASES is associated with an Entity Information Frame (EIF). An EIF is a generalized property list [Winston 1984] which can be divided into discrete elements called "slots". Each slot describes an attribute which may, in turn, contain one or more facets such as "value", "default", "if-needed", or "if-accessed". Every occurrence of an entity has the same Entity Information Frame (EIF) and an isomorphic substructure. During application, knowledge contained in the EIF is extracted and interpreted by the inference engine for design reasoning [Rozenblit et al. 1989]. An Entity Information Frame (EIF) is a structure:

$$\langle M, ATTs, DSF, ESF, CRS, CH \rangle$$

where

M: is the name of the associated model.

ATTs: are attributes of the entity.

DSF: is the design specification form.

ESF: is the experiment specification form.

CRS: are constraint rules for design synthesis.

CH: are children entities of M.

With FRASES representation, behavioral knowledge about objects is described by simulation models stored in the model base. M represents the key to access a model of the entity to which EIF is attached. ATTs are attributes used to characterize the associated object. Attributes of an entity are partitioned into two groups, i.e., static and dynamic [Rozenblit and Hu 1988]. Static attributes are variables used to describe properties of an object that do not change over time. Dynamic attributes are related to the dynamic behavior of the models represented by entity objects.

Design Specification Form (DSF) accepts the specification of design objectives, constraints, and criteria weighting schemes. The contents of DSF define the system requirements that must be satisfied by the system being

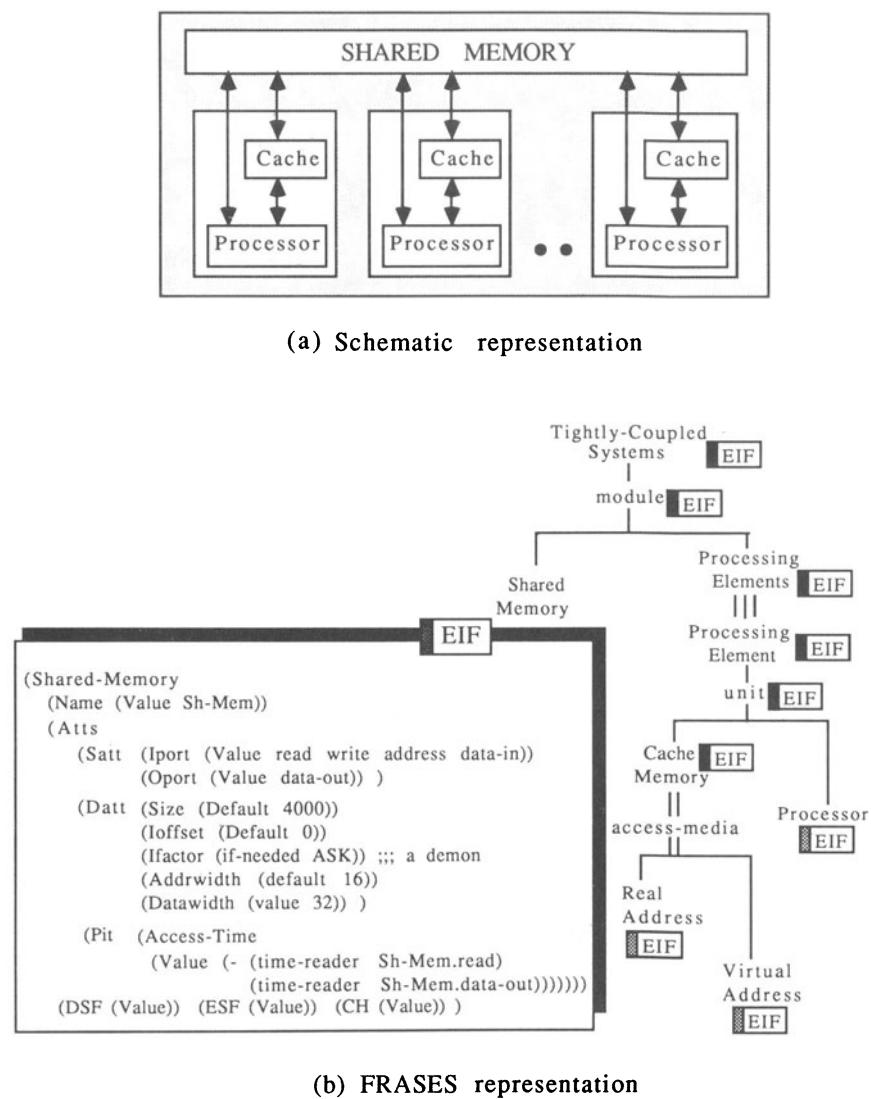


FIGURE 1. A tightly-coupled multi-processor system.

designed. The DSF information is used to guide the synthesis of design model structures. Each entity of FRASES has its own DSF. Once composition trees (i.e., a decomposition tree with information about the coupling schemes among model components) are generated based on the knowledge provided in the CRS (see below) slot, users are requested to define the simulation experiment in the Experiment Specification Form (ESF).

Finally, simulation is activated via automatic extraction and coupling of simulation models [Hu 1989].

Experiment Specification Form (ESF) is applied to accept the specification of simulation requirements such as an arrival process, event structure, and simulation control scheme. ESF provides information to direct the automatic generation of experimental frames [Hu 1989]. Again, ESF is placed along with entity nodes of a composition tree. For illustration, typical DSF for design of a processor of a tightly-coupled multi-processor system is shown in Figure 2. With the specified DSF, the processor should be capable of executing 10 Million Instructions Per Second (MIPS), the cost of the processor is less than 300 dollars, and the power consumption of the processor must be less than 0.5 μ watts. Figure 3 shows a simplified form of a simulation experiment for the processor. The simulation specification indicates that the input arrival rate for the first 100 events will follow the Poisson distribution with the mean value of 10. For subsequent events, the normal distribution with mean of 1 will be employed. Each event is composed of a symbolic identification and numerical workload. Simulation will be executed for 300 events. For each 50 system time units, the measurement of performance indices must be reported.

Constraint Rules for Synthesis (CRS) contains heuristic rules for configuring design model structures. Formally, selection constraint rules for pruning alternative are associated with specialization nodes, and constraint rules for synthesizing components are associated with aspect nodes. Rozenblit and Huang have defined model development driven by production rules in [Rozenblit and Huang 1987].

FRASES is a generative knowledge representation scheme which organizes the knowledge represented into an entity-based, hierarchical structure

Design Constraints ((> MIPS 10) (< cost 300) (< power-consumption 0.5))
Design Objectives ((max MIPS) (min cost power-consumption))
Criteria Weighting (:rw cost MIPS power-consumption)

FIGURE 2. Design specification form.

Arrival Process <pre>(cond ((< events 100) (Poisson 10)) (t (normal 1)))</pre>
Event Format <pre>(cond (t (list (symbol) (number 10.0))))</pre>
Simulation Control <pre>(cond ((> event 300) (stop)) ((= (div clock 50) 0) (report processor)))</pre>

FIGURE 3. Experiment specification form.

and allows the represented knowledge to be refined and inferred efficiently [Rozenblit et al. 1989].

4 KAR Based on FRASES

FRASES is a complete knowledge representation scheme for system design application, which conveys the declarative knowledge (structure and attributes) as well as procedural knowledge (production rules for design synthesis, verification, and evaluation). With the well- [defined axioms and operations, FRASES satisfies all the requirements of the KAR approach as follows:

1. The entity-oriented hierarchical structure of FRASES can be easily employed to represent design structures characterized by modularity, hierarchy, and regularity. By employing query rules based on the structural nature of FRASES, question patterns are generated automatically to acquire knowledge from domain experts.
2. The acquired knowledge is directly translated into an internal representation of Entity Information Frame (EIF) in the format ready for inferencing. This reduces human intervention to the minimum. Due to the graphic interface of FRASES, knowledge represented with FRASES is transparent to domain experts and system users.
3. FRASES provides an efficient scheme for knowledge representation, refinement, and inferencing by partitioning the global knowledge base into hierarchical and entity-oriented frame objects.

4. Contradiction, duplication, missing or incompleteness of essential knowledge can be detected by checking verification rules associated with each query rule. In other words, axioms and operations of FRASES are checked with verification rules to assure the consistency and validity of knowledge.

Selecting FRASES to conduct KAR, the query process can be depicted by state transitions as shown in Figure 4. The knowledge acquisition starts from querying knowledge about the problem domain. Once the problem domain is specified, the acquisition process falls into the Entity/Aspect/Specialization (EAS) loop. Component information and knowledge about determining design parameters are acquired at the "Entity" phase. Knowledge about functional decomposition, coupling, and synthesis constraints is elicited in the "Aspect" phase. Knowledge about alternatives and selection rules is acquired in the "Specialization" phase. Experts may choose the most appropriate design approach (e.g., top-down, bottom-up, and mixed) based on the characteristics of design problems by indicating processing priority between aspect and specialization nodes [Hu 1989]. At each design abstraction, the most crucial component is indicated by assigning the highest processing priority. The EAS loop will continue until desired level of abstraction is reached. After the FRASES structure is acquired, system identifies entities that require a corresponding behavior model and starts acquiring behavioral features for each entity. At each acquisition state, verification rules are automatically applied for checking knowledge inconsistency.

In Figure 5, a simple comparison between the conventional knowledge acquisition and KAR with FRASES is given. The diagram in the middle column delineates the acquisition process for developing a knowledge base. Several advantages are expected from using KAR with FRASES for knowledge acquisition:

Efficiency: Question patterns necessary to acquire design knowledge for decomposition, taxonomy, pruning, and synthesis of systems are generated automatically. Knowledge provided by users is verified automatically by applying appropriate verification rules to assure consistency of knowledge. With FRASES, verified knowledge is translated directly into a ready-for-inferencing Entity Information Frame (EIF) by a number of macros defined for managing frame objects [Hu 1989]. Once the knowledge acquisition process is completed, experts may practice a number of design applications by providing information required for Design Specification Forms (DSF) and Experiment Specification Form (ESF). By examining generated designs, experts can justify if knowledge refinement is required or if the design reasoning is performed correctly.

Flexibility: One of the problems found in conventional knowledge acquisition methods is that they only appear efficient in a specific problem

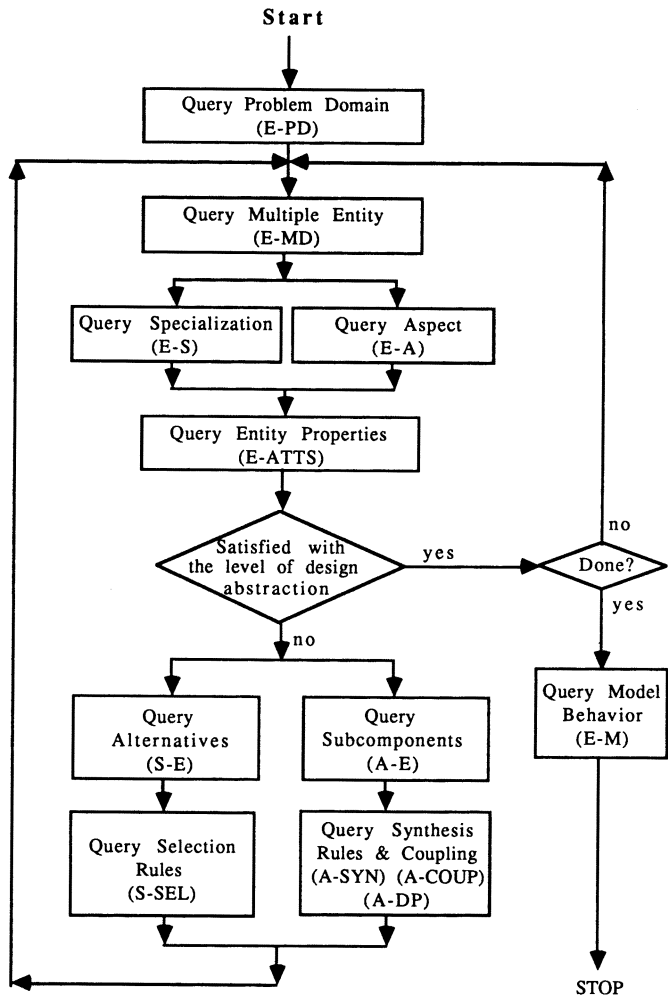


FIGURE 4. Query transition of KAR with FRASES.

domain. With the flexibility of FRASES, KAR can be used for a large class of systems exhibiting hierarchical and modular structure.

Manageability: The entity-based, hierarchical structure allows the knowledge represented to be examined and modified easily. During the knowledge acquisition process, users are allowed to modify both the structure and EIF contents of the FRASES tree.

Cost-Effectiveness: Unlike conventional acquisition approaches which require human intervention and labor-intensive preparation, interviewing, verification, translation, and organization, the complex process of knowl-

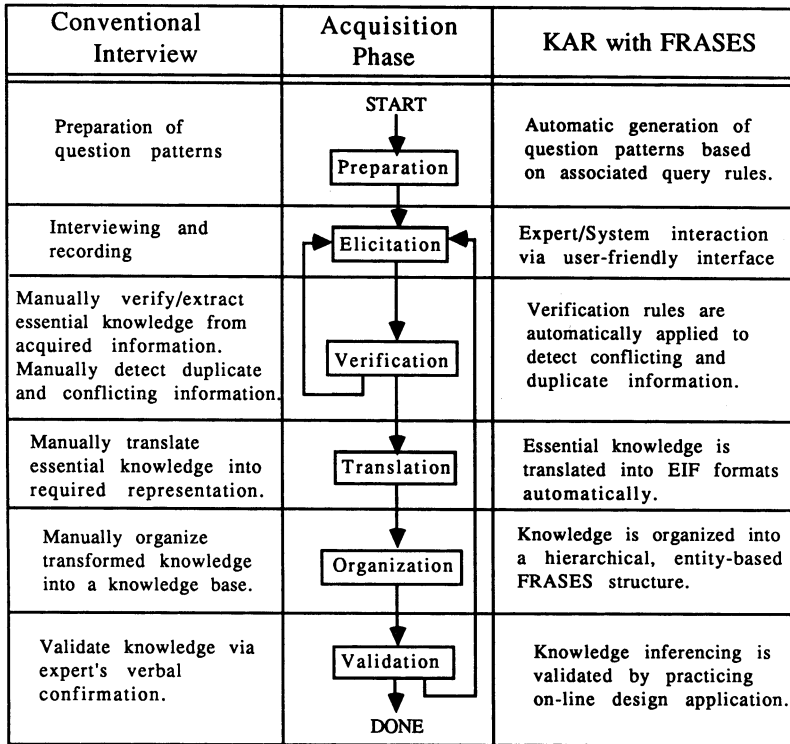


FIGURE 5. Interviewing vs. KAR with FRASES.

edge base development is automated and efficiently handled by KAR approach. The fast turnaround of knowledge base development highly reduces the cost of knowledge-based systems.

5 Query Rules of FRASES

In order to automate the generation of question patterns for constructing a FRASES tree, query rules are first defined in a database and linked to the acquisition tool. On each acquisition iteration, appropriate question patterns will be generated based on interpreting associated query rules. In general, query rules of FRASES are classified into four types:

I. Entity Query:

- A. Querying Problem Domain (E-PD).
- B. Querying Multiple Decomposition (E-MD).
- C. Querying Design Attributes (E-ATTS).
 1. Static Attributes (E-SATTS).

- 2. I/O Ports (E-PORTS).
- 3. Design Parameters (E-DPARA).
- 4. Performance Indices (E-PIX).
- 5. Processing Priority (E-PP).
- D. Querying Specializations (E-S).
- E. Querying Aspects (E-A).
- II. Specialization Query:
 - A. Querying Specialization Variants (S-E).
 - B. Querying Selection Rules (S-SEL).
- III. Aspect Query:
 - A. Querying Subcomponents (A-E).
 - B. Querying Synthesis Rules (A-SYN).
 - C. Querying Coupling Information (A-COUP).
 - D. Querying Design Priority (A-DP).
- IV. Model Query:
 - A. Querying Model Behavior (E-M).

Each query rule is associated with three explanation patterns (e.g., WHY, WHAT, and HOW) as follows:

WHY is the question asked?
 WHAT does this question mean?
 HOW to answer this question?

For example, the S-SEL rule of a specialization node has three explanation rules, termed S-SEL.WHY, S-SEL.WHAT, and S-SEL.HOW, to explain “Why selection rules are required for a specialization node?”, “What a selection rule means?”, and “How to specify a selection rule?”.

6 Verification Rules of FRASES

In order to assure the consistency of FRASES representation, verification rules based on axioms and operations of FRASES must be checked after each query rule is applied. In other words, to assure the consistency of knowledge provided by users, one or more verification rules will be applied on each query iteration. Whenever conflicting or invalid information is detected, error messages will be signaled to users so that an appropriate correction or modification can be made. For logical errors (e.g., conflicting information), the system will explain which axiom or operation of FRASES has been violated and how these problems can be solved. On the other hand, if a physical error (e.g., typos such as a missing token “IF” in the rule specification) is found, the system will correct the error by referring to the explanation rules associated with the entity (e.g., S-SEL.HOW explains

how to specify selection rules for a specialization node). If the user is not satisfied with the modification made by the system, he will be requested to give a correct input.

To verify logic errors, each query has verification rules which are employed to assure the consistency of the knowledge being represented. They are:

I. Query Problem Domain (E-PD)

Verifying Domain Existence (VDE): If an old domain existed, users are allowed to use, to discard, or to save the old domain knowledge.

II. Query Multiple Decomposition (E-MD)

Verifying Variation Range (VVR): What is the maximum and minimum number of entities allowed in the system.

III. Query Design Attributes (E-SATTS, E-DPARA, E-PIX):

Verifying Attached Variables (VAV): No two variables have the same name.

Verify Inherited Knowledge (VIK): All properties and substructures inherited through a specialization (i.e., taxonomy) must be validated for the current entity. This assures knowledge consistency for future pruning operations.

IV. Query Selection Rules (E-SEL), Query Synthesis Rules (A-SYN), and Query Coupling Information (A-COUP):

Verifying Knowledge Hierarchy (VKH): Knowledge contained in an Entity Information Frame is related only to itself and its substructures.

Verifying Conflicting Knowledge (VCK): Dead-ends (e.g., none of the specialization variants is selected) and inconsistency (e.g., coupling two output ports together) should be avoided.

Verifying Concise Form (VCF): Rules which are logically related should be combined into a concise form to reduce the complexity of knowledge bases.

V. Query Specialization (E-S) and Query Aspects (E-A):

Verifying Uniformity (VUF): Any two nodes with the same labels have identical Entity Information Frames and isomorphic substructures.

Verifying Structure Hierarchy (VSH): No labels appear more than once down any path of the FRASES tree.

Verifying Valid Siblings (VVS): No two entities under the same node have the same labels.

VI. Query Subcomponents (A-E):

Verifying Uniformity (VUF)

Verifying Structure Hierarchy (VSH)

Verifying Valid Siblings (VVS)

VII. Query Specialization Variants (S-E):

Verifying Uniformity (VUF)

Verifying Structure Hierarchy (VSH)

KAR/EXPERT INTERACTION	FRASES CONVERSION
KAR> What is your problem domain (E-PD) ? => Tightly-Coupled-System	Tightly-Coupled Systems
KAR> What kind of static attributes that you want to add to the system (E-SATTS)? => (designer Jeff) (date 4/18/89) (place UA) KAR> What kind of design parameters that you want to consider at this design level (E-DPARA)? => cost size KAR> What kind of performance indices are considered for evaluating a Tightly-Coupled-System (E-PIX)? => response-time thruput KAR> Can you classify a Tightly-Coupled-System based on certain specialization (E-S)? =>why KAR> -> This question is used to query how experts -> classify variants of the asked entity. For -> example, a computer network can be classfied -> into RING, BUS, and TREE based on Topology. KAR> Can you classify a Tightly-Coupled-System based on certain specialization? => nil KAR> Can you decompose a Tightly-Coupled-System based on certain aspect (E-A)? => module	Tightly-Coupled System EIF module
• • • KAR> What are these subcomponents when you decompose a Tightly-Coupled-System based on module (A-E)? => Processing-Elements Shared-Memory • • •	Distributed Systems EIF module / \ Porcessing Elements Shared Memory
KAR> Does the number of Processing-Elements vary with design requirements (E-MD)? => yes KAR> Specify the range for the number of Processing-Elements? => 0 64 <i>;;; multiple decomposition</i> • • •	Tightly-Coupled System EIF module EIF / \ EIF Processing Elements Shared Memory Processing Element

FIGURE 6. KAR with FRASES for tightly-coupled multi-processor system.

KAR> Can you classify a Cache based on certain specialization (E-S)?

=> access-media

KAR> What are these alternatives when you classify a Cache based on access-media (S-E)?

=> Real-Address Virtual-Address

KAR> Can you specify selection rules for determining access-media (S-SEL)?

=> r1: if desired system expandability = high then select Virtual-Address.

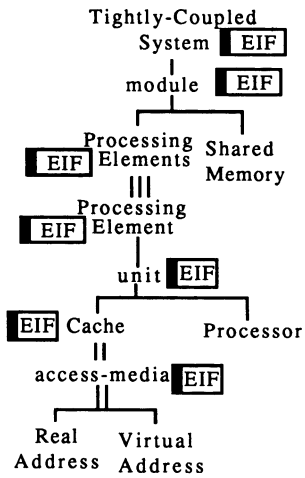
r2: if desired system expandability = low then select Real-Address.

• • •

KAR> Can each Processing-Element be decomposed into (Processors Real-Address-Caches) or (Processors Virtual-Address-Caches)?

=> yes

;;; verifying pruned knowledge (VPK)



• • •

KAR> Can you classify a Real-Address cache based on certain specialization (E-S)?

=> nil

KAR> Can you decompose a Real-Address cache based on certain aspect (E-A)?

=> nil

;;; satisfied with the level of design abstraction

• • •

KAR> Can you describe behavior model for Real-Address-Cache (E-M)?

• • •

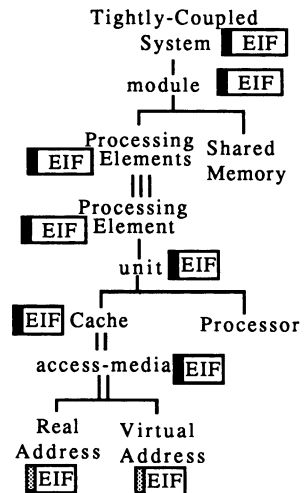


FIGURE 6. Continued

Verifying Valid Siblings (VVS)

Verifying Combinations of Knowledge (VCK): New knowledge obtained from combinations of multiple aspects and/or specializations of FRASES must be confirmed by the user. Constraints will be acquired to indicate invalid combinations.

Verifying Pruned Knowledge (VPK): Specialization variants should be valid for future pruning operations.

Verifying Transformed Knowledge (VTK): New knowledge transformed from the current structure of FRASES must be confirmed by the user.

With the rule base approach, verification rules defined for each query pattern can be flexibly refined by updating rules instead of reprogramming the system.

7 Example

To illustrate the operation of KAR with FRASES for knowledge acquisition, part of the query process for design of a tightly-coupled multi-processor system is illustrated in Figure 6. The acquisition process begins with generating a query about the problem domain. This query is activated by the E-PD structure incorporated in FRASES. After the answer is generated, FRASES converts it to its internal frame representation as shown in the right column of Figure 6.

The next step is to substantiate the Entity Information Frame. Relevant information is elicited through questions about the design parameters and performance indices. The Tightly-Coupled System is then decomposed into Processing Elements and Shared Memory by answering decomposition-oriented queries (E-A). Notice that there was a negative response (nil) to a query eliciting information about a possible taxonomy of the system.

The acquisition process is carried out until a level of hierarchical specification that satisfies the modeller is reached as shown in the last section of the left column in Figure 6.

8 Conclusions and Future Research

Conventional knowledge acquisition methods require a significant human effort to prepare, verify, translate, organize, and validate knowledge. When the complexity of systems grows, conventional methods become inefficient. To improve the efficiency of knowledge acquisition and to reduce the cost of knowledge-based systems, a new methodology called the Knowledge Acquisition based on Representation (KAR) is proposed. To realize KAR we have developed a knowledge representation scheme, termed Frame and Rule-Associated System Entity Structure process by using the automatic approach via exploiting the structural nature of a representation scheme.

With the KAR approach, both the cost and time for developing a knowledge base from which a design model construction rules are derived are drastically reduced.

Topics for future research include:

1. To incorporate psychological methods in the KAR methodology for a.) developing the best query rules with which domain experts are more willing to share their knowledge; b.) determining the reason a user has asked a question so that a customized response can be generated; and c.) verify the degree of validity of knowledge with respect to the psychological factors.
2. Upgrading the KAR interface with a natural language processor for design specification and answering system queries.

References

- Boose, J.H. 1988, "Uses of Repertory Grid-Centered Knowledge Acquisition Tools for Knowledge-Based Systems," *Int. J. Man-Machine Studies*, vol. 19, no. 3, p. 287-310.
- Gaines, B.R. 1987, "An Overview of Knowledge Acquisition and Transfer," *Int. J. Man-Machine Studies*, no. 26, p. 453-472.
- Gaines, B.R. 1988, "Knowledge Acquisition-Systems for Rapid Prototyping of Expert Systems," *Inform.*, vol. 26, no. 4, p. 256-285.
- Hart, Anna 1985, "Knowledge Elicitation: Issues and Methods, Computer-Aided Design, vol. 17, No. 9, p. 455-462. Hart, Anna 1985, "The Role of Induction in Knowledge Elicitation," *Expert Systems*, 2, p. 24-28, January 1985.
- Hu, Jhyfang and J.W. Rozenblit 1988, "Towards Automatic Generation of Experimental Frame in Simulation-Based System Design," *SCS AI Papers*, vol. 20, No. 1, p. 1-6.
- Hu, Jhyfang, Y.M. Huang, and J.W. Rozenblit 1988, "FRASES—A Knowledge Representation Scheme for Engineering Design," *Advance in AI and Simulation*, SCS Simulation Series, vol. 20, no. 4, p. 141-146.
- Hu, Jhyfang 1989, "Towards an Integrated Knowledge-Based Design Support Environment for Design Automation and Performance Evaluation," Ph.D. Dissertation, University of Arizona, Tucson.
- Kelly, G.A. 1955, *The Psychology of Personal Constructs*, New York: Norton.
- Kessel, K.L. 1986, "Methodological Tools for Knowledge Acquisition and Transfer," Proceeding of the 1986 IEEE International Conference on System, Man, and Cybernetics, Atlanta, GA.
- Minsky, M. 1975, "A Framework for Representing Knowledge," in Winston, P.H. (ed.), *The Psychology of Computer Vision*, New York: McGraw-Hill, p. 211-277.
- Newell, A. and H.A. Simon 1972, *Human Problem Solving*, Englewood Cliffs, NJ: Prentice-Hill.
- Nilsson, N.J. 1981, *Principles of Artificial Intelligence*, Palo alto, CA: Tioga Press.

- Olson, J.R. and Henry H. Rueter 1987, "Extracting Expertise for Experts: Methods for Knowledge Acquisition," *Expert System*, August 1987, Vol. 4, No. 3, p. 152-168.
- Quillian, M.R. 1968, "Semantic Memory," in Minsky, M. (ed.), *Semantic Information Processing*, Cambridge, MA: MIT press, p. 27-70.
- Ritchie, I.C. 1984, "Knowledge Acquisition by Computer Induction," Proceedings of UNICOM Seminar, London, England.
- Rozenblit, J.W. and Jhyfang Hu 1988, "Experimental Frame Generation in a Knowledge-Based System Design and Simulation Environment," to appear in: *Modeling and Simulation Methodology: Knowledge System Paradigms*, (M. Elzas et al., eds.), North Holland, Amsterdam.
- Rozenblit, J.W., Jhyfang Hu, and Y.M. Huang 1989, "An Integrated, Entity-Based Knowledge Representation Scheme for System Design," The 1989 NSF Engineering Design Research Conference, Amherst, Mass., June 1989.
- Schank, R.C. and Abelson, R.P. 1977, *Scripts, Plans, Goals, and Understanding*, Hillsdale, New York: Erlbaum.
- Shastri, L. 1988, "Semantic Networks: an Evidential Formalization and its Connectionist Realization," *Research Notes in Artificial Intelligence*, Morgan Kaufman Publishers, San Mateo, CA.
- Waterman, D.A. and A. Newell 1971, "Protocol Analysis as a Task for Artificial Intelligence," *Artificial Intelligence*, 2, p. 285.
- Weste, Neil and Kamran Eshraghian 1985, *Principles of CMOS VLSI Design—A System Perspective*, Addison-Wesley Publishing Company.
- Winston, Patrick H. and Berthold Klaus Paul Horn 1984, *LISP* (Second Edition), Addison-Wesley, Reading, Massachusetts.
- Zeigler, B.P. 1984, *Multifaceted Modelling and Discrete event Simulation*, Academic Press.
- Zeigler, B.P. (1987a), "Knowledge Representation from Newton to Minsky and Beyond," *Applied Artificial Intelligence*, 1: 87-107.