Watson, I.D., Shave, M.J., & Moralee, D.S. (1989). A Knowledge Analysis Methodology Using an Intermediate Representation Based on Conceptual Graphs. In: Proc. 9th. Int. Workshop on Expert Systems & their Applications, Vol. 1: pp.183-198.

A Knowledge Analysis Methodology Using an Intermediate Representation Based on Conceptual Graphs

Ian Watson, Mike Shave & Stuart Moralee

University of Liverpool
Dept. of Computer Science
Chadwick Building
Liverpool

Abstract

A high-level intermediate representation of the knowledge of a task domain is recognised as a useful step in the design of knowledge-based systems. It can serve as a common medium for discussion between domain experts and knowledge engineers, thereby facilitating knowledge enhancement. It can also form an unambiguous statement from which to implement a knowledge-based system, and it can represent a permanent record of the knowledge included in the system. This paper describes a methodology for guiding the knowledge analysis of a domain, and for constructing a model of the domain knowledge in the Conceptual Graph Knowledge Representation Language. The domain model can then be easily implemented in Prolog or in a knowledge representation language that directly supports Conceptual Graphs

KEY WORDS: Conceptual Graphs expert systems, knowledge elicitation, knowledge acquisition, knowledge analysis, intermediate representation

introduction

Currently, the task of designing knowledge-based systems frequently proceeds in an ad hoc manner. The process of knowledge engineering can be hampered by the lack of formal specifications for the desired system and by the difficulty of knowing how to start organising the relevant knowledge of the task domain [Alexander et al., 1986; Partridge, 1989]. This paper will address this by examining the use of a methodology for knowledge analysis resulting in an intermediate knowledge representation in the Conceptual Graph Knowledge Representation Language (CGKRL).

The design and implementation of expert systems typically involves four interacting functional components: domain experts, knowledge engineers, a prototype program, and verification of the prototype [Buchanan, 1979]. The knowledge engineers elicit knowledge from the experts and implement a prototype, which is then reviewed by the domain experts.

However, there are several advantages in creating an intermediate representation of domain knowledge at a level more abstract than that of the program:

- the intermediate representation can serve as a record of the elicited domain knowledge, independent of any implementation [Alexander et al., 1986; Plant, 1987; Wielinga & Breuker, 1986; Young, 1987].
- the intermediate representation enables the domain experts and knowledge engineers to discuss the domain in a common medium. This enhances the knowledge analysis process by serving as a model of the domain [Alexander et al., 1986; Edwards, 1987; Butler & Chamberlin, 1987; Recogzei & Plantinga, 1987].
- the intermediate representation can facilitate the identification of errors and omissions, and reduce misunderstandings about the knowledge structures of the domain [Alexander et al., 1986]. Thus an intermediate representation can reduce the likelihood of costly, and complex rewrites of a prototype.
- the intermediate representation can serve as a means of deciding upon the most suitable implementation for the system [Wielinga & Breuker, 1986; Plant, 1987], without allowing current implementational trends to affect the knowledge elicitation/analysis processes.
- the intermediate representation can be used as a basis for explanation, training, and system documentation after implementation of the system [Young, 1987].

The use of “paper models” or graphical representations of knowledge is a powerful and intuitive way of explicitly representing knowledge [Johnson-Laird, 1983; Pylyshyn, 1984]. Graphical domain models combined with textual statements of knowledge have been used successfully in KEATS [Motta et al., 1986; Rajan et al., 1989] and more recently in KADS [Breuker & Wielinga, 1985] to facilitate the processes of knowledge elicitation and acquisition. The CGKRL described here provides a statement of domain knowledge, which is independent of any specific implementation. Thus, the resulting model could eventually be used to implement an expert system in any desired AI environment or language.

This paper will first introduce the concept of describing expert domains at the knowledge level, and will then illustrate the knowledge analysis methodology with a test case.

knowledge representation at the knowledge level

Firstly, it must be stated that this paper makes a distinction between a representational language and an implementational language, which is not always apparent in AI literature [Bobrow, & Winograd, 1977; Brachman, 1979; Brachman, et al., 1983; Hines & Unger, 1987]. Our knowledge representation is intended to be completely independent of any specific computational implementation, as for example, the theories of frames, production rules, or semantic nets are independent of specific implementations. This distinction has, however, frequently been blurred, and knowledge representations are often implementational languages containing knowledge pertaining to the control and execution of the program, in addition to knowledge about the domain.

Alexander et al., [1986] posited a knowledge level analysis producing a formal specification of domain knowledge independent of implementational details. Their ontological analysis provides a description of a task domain at the *knowledge level* [Newell, 1982; Pylyshyn, 1984]. The knowledge level is a higher (more abstract) level lying above the symbol, or program level. That is, the function of knowledge is independent of any symbolic representation required to express it. At the knowledge level only the behaviour of a system and the set of operations required to achieve the behaviour need be specified; the structures used to implement the behaviour need not be specified [Levesque, 1984]. For example, the abstract data type is described in a high-level manner, in terms of its behaviour and its operators, rather than in terms of any specific computational implementation [Johnson-Laird, 1983].

High-level representations of knowledge can therefore facilitate understanding by enabling domains to be described independently of the implementations which might support any resulting system. Such descriptions at the knowledge level constitute “functional descriptions.” For example, a clock may be described by the behaviour of its mechanism, whereas its function is to tell the time to an observer [Bobrow, 1984].

How then are we to describe knowledge functionally? One way is to construct an ontology defining the concepts within the domain of expertise. Regoczei and Plantinga [1987] proposed a methodology for natural language based knowledge acquisition which restricts itself to compiling an ontology and an inventory list for a domain. Our ontology will not only model entities and instances, but will also represent the relationships between them. Our basic ontology according to this criterion must contain entities and instances, their properties, functions, and relations between them. It should also admit properties of properties, relations of relations, and so on, in a system that has no *a priori* limit to the construction of higher-order properties and relations.

The idea that meaning may be expressed by a set of concepts has been used in a variety of differing domains within AI. A well known application is that of Schank [1973] in natural language understanding. The conceptual dependency concepts of Schank were specifically crafted to describe the underlying meaning in natural language. However, there is no *a priori* reason why a finite set of semantic concepts could not describe meaning or function in other AI domains. Sowa [1984] has proposed a more complex set - the Conceptual Graph. This representation is based on an extension of the entity-relationship diagrams of Chen [1983], which have proved useful for data description.

Entity-relationship diagrams are not, however, detailed enough to represent the full meaning of natural languages. Conceptual graphs were therefore developed as an intermediate semantic representation [Sowa et al., 1985] which may be used to describe any domain by a process termed *conceptual analysis*; this results in a precise catalogue of concepts, relations, facts, and principles. Thus, in the domain of knowledge-based systems, conceptual analysis is synonymous with *knowledge analysis*.

Conceptual Graphs are a way of representing both declarative and procedural knowledge in a uniform and disciplined manner. Moreover, their simple syntax can be combined with a visually expressive representation, enabling people who are not familiar with the knowledge representation to understand the graphs quickly. Conceptual graphs support a number of knowledge structures, enabling most knowledge types to be adequately represented. A full account can be found in Sowa [1984].

A particular point of interest to our application is that a *type hierarchy* is created in the CGKRL which enables domain knowledge to be considered at differing levels of abstraction. Thus, for example, if the domain of interest were disease diagnosis, it would be possible to treat the domain at the general level of infectious organisms, and also at the more specific levels of bacteria, and viruses. Hence, knowledge common to all infectious organisms would be associated with the more general concept, whilst only knowledge pertaining to bacteria would be associated with that concept.

Moreover, since a concept may itself form a conceptual graph, a description of a domain may be composed of a set of nested individual graphs (cf. Figures 1, 2, & 3), rather than a single large graph, as is the case with other semantic networks. This facility proves particularly useful during iterative knowledge elicitation/analysis cycles, permitting the details of a concept to be considered initially at a very abstract level. Subsequent cycles then add detail to general concepts, facilitating top-down knowledge elicitation and analysis. Conceptual graphs thereby allow several levels of knowledge abstraction to co-exist in the domain model, a desirable attribute of knowledge representations [Gladun & Rabinovich, 1979].

knowledge elicitation and analysis

It is often convenient to divide the knowledge of a task domain broadly into declarative knowledge, and procedural knowledge. The division between each category in most domains will be imprecise, but nonetheless it has been recognised that such divisions can assist knowledge elicitation [Alexander, et al., 1986; Edwards, 1987], since it is sometimes necessary to differentiate between the “facts” of a domain and the procedures which “operate” upon them. Alexander, et al., [1986] in their ontological analysis proposed treating knowledge in three steps: first the facts of a domain are defined, then the operations which act upon them, and finally the heuristics which control the operations. Interestingly, Edwards [1987] in his detailed account of a knowledge elicitation/acquisition process states that, the knowledge engineer should attempt to describe the procedural knowledge prior to describing the declarative knowledge. This apparent contradiction is mitigated because both agree that the distinctions between categories of knowledge are actually far from rigid, and that some declarative knowledge must be described to make sense of the procedural knowledge, and vice versa.

A strength of Conceptual Graphs is that they support a uniform representation for both procedural and declarative knowledge [Hines & Unger, 1987], and therefore it is not necessary to differentiate between knowledge types during analysis. This results in a more natural grouping of the knowledge in a domain, allowing the domain facts to be grouped with the procedures and heuristics which operate upon them. In our view Edwards was correct in emphasising that the procedural knowledge makes *sense* of a domain by imposing a structure on the relationships between the declarative knowledge. This implies that top-down knowledge elicitation is preferable; it enables the domain to be treated initially at its most abstract level.

overview of the methodology

The knowledge analysis methodology may be summarised by the steps described below. Source material for analysis may take the form of recorded (audio or visual) transcripts from interviews or textual sources.

1. The transcript is processed by a series of simple steps which remove noise and repetition, standardise terminology and tenses, and resolve ambiguity.
2. The transcript is then divided into modules, each referring to a particular subject. The transcript is now in a form which is suitable for translation into the representation. This is known as the *source document*.
3. The source document, is translated by a number of steps into the CGKRL to produce an intermediate representation. Translation proceeds statement by statement from the most general statements to the most specific. It is at this stage that the domain knowledge takes on a structured form, with the relations between domain concepts being explicitly represented. This becomes the *domain model*.
4. The domain model is examined jointly by the knowledge engineers and domain experts resulting in further knowledge elicitation/analysis cycles, until such time as the model is judged adequate.
5. When knowledge analysis is completed the resulting domain model can then be used as a document from which implementation can proceed.

the factory test case

The methodology has been refined by the use of several test cases. The source material for the first case study was a University admissions handbook. This required very little processing in order to make a suitable source document from which translation could take place. Indeed, it was the well-structured, precise, and unambiguous form of this handbook, which highlighted the desirability of such a form for future source documents. A second case study used elicitation transcripts dealing with soil erosion. This material had been used by another researcher to develop an expert system using the mathematical modelling language “Z” [Plant, 1987].. This gave us the opportunity to compare the two approaches.

Our third case study dealt with the running of a small manufacturing plant. This factory mixes two liquids to manufacture a detergent of a precise specification. One of the liquids is delivered by lorries and loaded into a storage tank, whilst the second liquid is obtained from a reservoir tank on site. The liquids flow through pipes controlled by valves into a mixing vessel, and finally the detergent flows to a packaging unit. The specification of the detergent can be altered by controlling the flow of liquids into the mixing vessel.

An interview with an expert familiar with the running of the factory was tape recorded, providing a transcript as a basis for the knowledge analysis. This transcript required considerable processing to render it in a suitable form. Work on case studies has found the following characteristics to be desirable in the source document:

- terminology and usage must be standardised.
- tenses should be standardised to avoid temporal ambiguity.
- text should be as unambiguous as possible, subjects should be referred to directly, not indirectly.
- the document should be structured into modules such that each module contains an identifiable subject grouping.
- modules should be logically ordered, such that their grouping reflects the grouping of subject groups within the domain.
- modules should be divided into units, such that each unit has an identifiable subject; units may be sub-divided into sub-units describing aspects of the unit's subject.
- units should be logically ordered as for module.

As an example, the following is a section from the elicitation transcript describing the factory:

Expert: *Right, the object of this system ummm...is to produce detergent within a certain specification, to maximise profits....ummm...in this particular system I’m allowed a certain time or a maximum of five lorries arriving at one time, or rather waiting at one time.....Ummm I can open or close the valves....aahh....lorries arrive randomly, I don't have any control of the lorries arriving, but I have control on holding the lorry or unloading.*

The above fragment is not yet suitable for translation into the intermediate representation; it is poorly structured, it deals with a number of separate subjects, and it is ambiguous. The initial transcript was processed through the stages described above to produce concise unambiguous text.

The decision as to what are the primary subjects of the domain should be taken jointly by domain experts and knowledge engineers, but at most 20-30 subjects should be decided upon. This is similar to the practise in traditional data analysis in which a manageable number of key entities are chosen initially to describe the domain. Thus, in the factory example the following set of subjects were selected: Strategy, Controls, Costing, Specification, Tanks, Valves, Material, Product, Lorries, Overflow, Pipes, Flow, and Extras. These constitute a range of knowledge types from heuristic, through procedural to declarative. However, the subject groupings, or modules, are not exclusive, and a statement could be assigned to more than one module. Consequently, statements are also given a qualitative measure (high, medium, or low) of their relevance to a module. By this stage the elicited knowledge was in the form of the source document from which translation into the intermediate representation could take place.

It is desirable for the knowledge engineers to be able to follow a disciplined translation methodology, which would produce a correct set of conceptual graphs when applied to any set of statements. However, since English can be ambiguous this task is not simple. First, it must be emphasised that it is not our intention to translate natural language into the CGKRL since, as is described above, the elicited material undergoes pre-processing to produce a source document in a form suitable for translation. The objective then, was to design a series of steps which, when applied to a suitable piece of processed text (the source document) will facilitate accurate translation into the CGKRL.

Translation occurs in a top-down manner: a statement from the source document is selected (preferably by the domain experts) as being a representative general statement of the goals or function of the domain, or a summary of the problem area. The concepts within this statement are then assigned places in the type hierarchy. Since the first statement is very general, these concepts should occupy places at the top of the hierarchy. For example, it was decided that the following statement (from the above extract of the transcript) was the most general statement of the function of the domain:

[1/#9.1: (HIGH) The object of this system is to produce detergent within a certain specification,
[1/#9.2: (HIGH) and to maximise profits.]]

This statement was translated into a type definition for SYSTEM as in Figure 1. This gave us the first conceptual graph of the domain model; henceforth any use of the concept SYSTEM had to conform to this definition.

Figure 1.

Translation then proceeds by describing the concepts in this first graph by adding more knowledge obtained from appropriate statements in the source document. This additional knowledge may be represented by any of the knowledge structures supported by conceptual graphs. At all times, when new concepts are added to the model, they are placed in the type hierarchy, and when they are subsequently used they must conform to type. New concepts which are used to elaborate initial concepts are then in turn described in increasing detail. In this manner translation proceeds from the most general statements of the domain, to general descriptions of the objects in the domain, and finally to more detailed descriptions of processes and objects.

For example, MAXIMISE is described by the definition in Figure 2, and MAX_STRATEGY is described by the schema in Figures 3a, 3b, and 3c. MAXIMISE is defined as a type of ACT using the method MAX_STRATEGY. This concept is in turn described by three schema each of which describes an alternative method to ensure the factory is running efficiently. These schema are not exclusive, and each description of MAX_STRATEGY may be applied at the same time.

Figure 2.

Figure 3a.

Translation from the source document into the CGKRL is continually guided by the developing model itself. New graphs added to the model must not contradict any graphs already contained in the model; thus the meaning or usage of concepts must remain constant throughout knowledge analysis. It is eventually intended that knowledge analysis will be carried out with the aid of a case tool, which will constantly check that concepts are conforming to type, and satisfying the requirements of associated definitions, prototypes and schema as the model is built up. However, these tasks are currently performed by hand during the translation process.

Figure 3b.

Figure 3c.

knowledge enhancement

We have found that the simple syntax and expressive representation of the CGKRL can enable domain experts to identify ambiguities and omissions easily. In view of the importance of this information, we can say that in this sense knowledge about the domain is *enhanced*. The concept of *knowledge enhancement* is central to the use of an intermediate knowledge representation, and to the choice of the CGKRL as the knowledge representation language. We believe, that the representation of domain knowledge in a high-level intermediate representation, free from implementational constraints, assists knowledge engineers and domain experts in checking the accuracy of the knowledge necessary for representing the problem task.

In general, any procedures which encourage critical examination of the knowledge of a task domain will result in some enhancement of that knowledge [Wielinga et. al.,., ?? Chadreskaran

??; Steels, ??]. However, the following features of the CGKRL make it particularly suitable to this task:

- it allows a domain to be considered initially at a general or abstract level, which enables the general outline of the domain to be considered prior to the detail.
- it has a simple syntax, and an expressive graphical representation which enables the domain experts to examine the model directly, a feature notably lacking if either first order predicate calculus, or high-level languages such as LISP or Prolog were used.
- it supports consistency during knowledge analysis.
- it can highlight imprecise or ambiguous definitions of domain concepts and relations, it enables knowledge which is unrelated to the domain to be identified, and it can facilitate the identification of significant omissions.

During knowledge analysis, translation is constrained principally by the type hierarchy - a concept must *always* conform to type. Thus, if a concept were intended to replace a default concept during translation, but it did not conform to the type of the default concept, a mismatch would occur. Type mismatches are not permitted, and either the type of the intended replacement, or the type of the default concept, would need to be changed. If neither of these actions could be taken the knowledge engineers and domain experts would need to review the type hierarchy in order to resolve the mismatch. In addition, the use of previously defined definitions, prototypes, and schema ensures that the usage, or meaning, of concepts remains constant during analysis, enabling the CGKRL to maintain consistency during knowledge analysis, regardless of the size of a domain, and regardless of the number of people performing the analysis.

It is possible to present the resulting domain model in a variety of forms: a concise textual form, and the more expressive graphical form (cf. Figures 1 through 3c). However, unlike certain database analysis tools the graphical representation is a complete translation of the textual form. Moreover, it is possible to provide a number of useful overviews of the domain model. Figure 4 shows a schematic diagram of the relations between certain concepts within the domain. The numbers under the concept names refer to graphs in the model describing the concepts. For instance, the concept MAXIMISE is described by the definition in graph #11 (cf. Figure 2).

Figure 4. Schematic diagram showing relationships of concepts from graph #1

This form of diagram has proven useful by enabling the domain expert to check quickly the relationships between concepts. The diagram also demonstrates the similarity of conceptual graphs to entity-relationship diagrams. Experience has shown us that a domain expert who had received only a brief introduction to the representation was able to use the various facilities provided by the CGKRL to review the domain model, and to identify errors, ambiguities, and omissions, thus leading to a further knowledge elicitation/analysis cycle.

conclusion

We believe that the use of an intermediate representation of domain knowledge lying between the elicitation of the knowledge, and the implementation of a prototype can assist knowledge analysis, and thus the design of expert systems. The description of a domain at the knowledge level permits the knowledge to be considered free from all implementational constraints and details; this allows the domain experts and knowledge engineers to construct a domain model before implementing a prototype. The CGKRL is particularly suited to describing a domain at the knowledge level. It can enable experts to define the concepts and relations of the domain in a natural form, whilst permitting the knowledge engineers to structure them. Furthermore, the CGKRL can actively maintain consistency of usage and meaning during knowledge analysis. Finally, by enabling domain experts to examine the model themselves the CGKRL can highlight omissions and ambiguities, thus facilitating knowledge enhancement.

The use of the CGKRL permits knowledge engineers to follow a disciplined methodology during knowledge analysis since the central task of translating the source document into the CGKRL is guided both by a set of clearly defined steps, and by the constraints imposed by the CGKRL itself. Although knowledge elicitation still remains a complex task, the use of the CGKRL as an intermediate representation enables the cycles of elicitation and analysis to be focused and directed by the developing domain model itself.

We believe that the advantage of this methodology over the prototyping method of knowledge-based system design, lies in the ability of the CGKRL to support consistency and enable domain experts to check the model personally, so that a resulting implementation is likely to be more complete than if it were constructed in an ad hoc manner. In support of this we have found in a case study that errors, ambiguities, and omissions in the elicited knowledge were identified by a domain expert who had received only a brief introduction to the representation. This therefore leads us to believe that a well-designed knowledge analysis tool utilising an intermediate representation in the CGKRL could significantly enhance knowledge analysis.

References

Alexander, J.H. et al. (1986). Knowledge level Engineering: Ontological Analysis. AAAI-86 vol.2, pp.963-968.

Bobrow ,D.G. & Winograd, T. (1977). An Overview of KRL, A Knowledge Representation Language. Cognitive Science, 1(i).

Bobrow, D.G. (1984). Qualitative Reasoning about Physical Systems: An Introduction. AI 24: pp.1-5.

Brachman, R.J. (1979). On the Epistemological Status of Semantic Networks. Associative Networks: Representation and the Use of Knowledge by Computers, pp.3-50.

Brachman, R.J., Fikes, R.E. & Levesque, H.J. (1983). KRYPTON: A Functional Approach to Knowledge Representation. IEEE Computer 16(10): pp.67-73.

Breuker, J., & Wielinga, B. (1985). KADS: Structured Knowledge Acquisition for Expert Systems. Proc. of Expert Systems and their Applications, Avignon.

Buchanan, B.G. (1979). Issues of Representation in Conveying the Scope and Limitations of Intelligent Assistant Programs. In: Machine Intelligence 9, pp.407-425. Hayes, J.E., Michie, D. & Mikulich, L.I. (Eds.). Ellis Horwood Ltd., Chichester.

Butler, A. & Chamberlin, G. (1987). The Aries Club - Experience of Expert Systems In Insurance & Investment. In: Research and Development in Expert Systems IV, pp.246-257. Moralee, D.S. (Ed.). Cambridge University Press.

Chen, P.P. (1983). English sentence structure and Entity-Relationship diagram. Information Science, 29: pp.127-150.

Edwards, A. (1987). Knowledge Elicitation Warts and All: A Case Study. Proc. The British Accounting Association Conference, Glasgow - April 1987.

Gladun, V.P. & Rabinovich, Z.L. (1979). Formation of the World Model in Artificial Intelligence Systems.In: Machine Intelligence 9, pp.299-309. Hayes, J.E., Michie, D. & Mikulich, L.I. (Eds.). Ellis Horwood Ltd., Chichester.

Hines, T.R. & Unger, E.A. (1987). A Variation of Conceptual Graphs: An Object-Orientated Approach. Proc. 1987 Fall Joint Computer Conf.-Exploring Technology: Today and Tomorrow. IEEE Comp. Soc. Press, pp.57-64.

Jackman, M.K. (1987). Inference and the Conceptual Graph Knowledge Representation Language In: Research and Development in Expert Systems IV, pp.66-75. Moralee, D.S. (Ed.). Cambridge University Press.

Johnson-Laird, P.N. (1983).Mental Models. Cambridge University Press, Cambridge.

Levesque, H.J. (1984). Foundations of a Functional Approach to Knowledge Representation. AI 23: pp.155-212.

Motta, E. et al. (1986). KEATS: The Knowledge Engineer's Assistant. Human Cognition Research Laboratory, Milton Keynes, Technical Report No.20.

Newell, A. (1982). The Knowledge Level. AI 18: pp.87-127.

Partridge, D. (1989). To add AI, or not to add AI? In: Research and Development in Expert Systems V, pp.3-13. Kelly, B., & Rector, A. (Eds.). Cambridge University Press, Cambridge, UK.

Plant, R. (1987). A Methodology For Knowledge Acquisition In The Development Of Expert Systems. Ph.D. Thesis, University of Liverpool, UK.

Pylyshyn, Z.W. (1984). Computation and Cognition. MIT Press, Cambridge, Massachusetts.

Regoczei, S. & Plantinga, E.P.O. (1987). Creating the domain of discourse: ontology and inventory Int. J. Man-Machine Studies, 27: pp.235-50.

Rajan, T., Motta, E., & Eisenstadt, M. (1989). Acquist: A Tool for Knowledge Acquisition In: Research and Development in Expert Systems V, pp.113-125. Kelly, B., & Rector, A. (Eds.). Cambridge University Press, Cambridge, UK.

Schank, R.C. (1973). Identifications of Conceptualizations Underlying Natural Language. In: Computer Models of Thought and Language pp.187-247. Schank, R.C. & Colby, K.M. (Eds.). W.H. Freeman & Co., San Francisco.

Sowa, J,F. (1984). Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley Publishing Co. Inc., Reading, Massachusetts.

Sowa, J.F., Chen, P.P., Freeman, P., Salveter, S.C., & Schank, R.C. (1985). Mapping Specifications to Formalisms. Proc. 4th. Int. Conf. on Entity-Relationship Approach, pp.100-1.

Wielinga, B.J. & Breuker, J.A. (1986). Models of Expertise. Proc. of the European Conference on AI, pp.306-318.

Young, R.M. (1987). The Role of Intermediate Representations in Knowledge Elicitation. In: Research and Development in Expert Systems IV, pp.287-288. Moralee, D.S. (Ed.). Cambridge University Press.