

**THE KNOWLEDGE DICTIONARY:  
STORING DIFFERENT KNOWLEDGE  
REPRESENTATIONS.**

Bob Jansen\* & Paul Compton†

TR-FD-89-02

**ABSTRACT**

This paper describes interim results of research in the area of the software engineering of knowledge based systems. It discusses several problems identified in the knowledge base work by the research group, especially in the area of design, documentation and maintenance. One of the major areas of concern is identified as

---

\* CSIRO Division of Information Technology, PO Box 1599, North Ryde, NSW, 2043, Australia

the lack of integration of the various 'standard' knowledge representation formalisms. The research aims to test the application of conventional software engineering methodologies to the software engineering of knowledge based systems. The methods employed include the entity-relationship analysis of various knowledge representations, and the documentation of these representations in a data dictionary augmented to enable the definition and description of knowledge in addition to information and data. This environment is called the Knowledge Dictionary. The results are discussed in the context of this environment which is used as a test environment for the software engineering of an expert system, Garvan-ES1. The paper describes conceptual models of some of the standard knowledge representations, how the Knowledge Dictionary may be used as an aid to integrate some of these representations, the benefits in doing so, and discusses the relationship that exists between knowledge representation and knowledge acquisition.

#### KEYWORDS

data dictionary, knowledge representation, production rules, semantic nets, frames, ripple down rules, knowledge dictionary, database software, knowledge engineering, relational calculus, knowledge acquisition

#### ELECTRONIC MAIL ADDRESSES

ACSnet: jansen@ditsyda.oz                      CSNET: jansen@ditsyda.oz  
JANET: ditsyda.oz!jansen@ukc                ARPA: jansen% ditsyda.oz@ seismo.css.gov  
UUCP: {enea, hplabs, mcvox, prlb2, seismo, ubcvision, ukc}! munnari !ditsyda.oz!jansen  
AUSPAC: jansen@au.csiro.ditsyda

## 1. INTRODUCTION.

In recent times, the field of knowledge based systems has seen an increase in the number of methods of representing knowledge. This increase is no doubt due to the complexity of knowledge and the perception that in tackling a new problem, existing representations may be only partially suitable. This then raises the question of whether to adopt an existing representation, possibly compromising the richness inherent in the knowledge domain, or to develop a new representation suitable to capture the domain. As with conventional system's design, the choice of either option is a compromise. However, the choice of the right representation is crucial in ensuring the robustness and usefulness of the software product, and usually leads to the translation of difficult problems into simpler ones (Winston, 1984).

Various 'standard' knowledge representations exist, namely *production rules* (Davis & King, 1984), *frames* (Minsky, 1974), *semantic nets* (Winston, 1984), *conceptual structures* (Sowa, 1984), *Prospector like inference networks* (Lister, 1987), and *ripple down rules* (Compton & Jansen, 1988). The common element between all of these representations is suitably summed up by Winston, "All representations must provide some way to denote objects and to describe the relations that hold among them" (Winston, 1984, page 253).

With the introduction of knowledge based systems into the commercial arena, problems arise regarding the access by the inference engine to data already stored in traditional databases. There is much work proceeding regarding solutions to this problem, (Al-Zobaidie & Grimson, 1984, Delcambre, 1988, Jansen & Compton, 1988a, Kaiser, 1988, Zaniolo, 1986) but as yet there appears to be no all encompassing solution. In addition, there are now several conferences devoted almost solely to the integration of knowledge based systems and databases (for example the Expert Database Systems workshop) showing the importance attached to this integration problem with the introduction of AI technology into the commercial computer area. The required integration introduces a major issue in the area of design and software engineering; how can a complex system, having both conventional and knowledge based components, be software engineered? There are as yet no design tools commercially available to aid the designer in the overall design process.

The work of maintaining GARVAN ES1 (Compton & Jansen, 1988) has led us to realize that knowledge is more complex than is perhaps envisaged, and may need computing solutions incorporating aspects of both holistic and reductionist strategies (Compton & Jansen, 1989). There appear to be philosophical aspects to the knowledge acquisition phase that should be taken into account to explain the problems that are being reported (eg. Shaw, 1988 on the problems of multiple experts) which need to be re-evaluated. This topic is not of major concern in this paper, but is discussed in more detail in Compton & Jansen, 1989. However, it appears that knowledge representation and knowledge acquisition go hand in hand as part of the analysis process, and not sequentially, that is, knowledge acquisition is not primarily a process of getting the expert to remember the knowledge but to acquire and represent the knowledge in a suitable integrated knowledge structure.

This paper describes preliminary results from on-going work that attempts to tackle some of these issues, to provide an environment where expert systems are maintainable for the long term, and can be integrated into large knowledge based systems. The work is centred around the use of existing software engineering techniques in the area of knowledge based systems, and developing models of software products incorporating both conventional and knowledge based components. This paper focuses on four knowledge representation formalisms, the *production rule*, *semantic nets*, *frames* and *ripple down rules*. Other knowledge representations will be the subject of future papers as the work proceeds. The environment in which the work is described is a development called the *Knowledge Dictionary* (Jansen & Compton, 1988b).

## 2. THE KNOWLEDGE DICTIONARY

Over the last two years, the CSIRO Division of Information Technology and the Garvan Institute of Medical research have been involved in a collaborative project aimed at the design and maintenance of expert systems, using as the target, GARVAN ES1 (Horn *et al*, 1985), an expert system for the clinical interpretation of laboratory reports. This expert system has grown more difficult to maintain over the four years of routine production use, due mainly to the refinement of the knowledge in the domain rather than an expansion of the domain (Compton & Jansen, 1988). In parallel with this project, the CSIRO Division of Information Technology has been redeveloping SIRATAC (Hearn *et al*, 1986), a cotton pest management decision support expert system, utilizing modern software engineering practises in the redevelopment.

A major development from these two projects is a tool called the *Knowledge Dictionary* (Jansen & Compton, 1988b). This tool was derived from techniques forming part of conventional software engineering, in particular the application of data dictionary technology to the area of knowledge based systems. To date, the Knowledge Dictionary may be used to record knowledge in the form of production rules, using the relational data model (Codd, 1970) supporting the underlying representation. This representation may be manipulated into conventional production rule of the form IF...THEN.... It should be noted that although the user interface representation is the production rule, the knowledge is not stored in production rule format, but is decomposed into constituent components and stored in relational tables.

Among the advantages of representing knowledge in the Knowledge Dictionary are:-

- the knowledge may be browsed using as a starting point any concept in the knowledge domain. For example, starting at a particular rule, the user can browse to other rules, facts, and rule actions related to the starting rule in any way; use of similar facts in the premise, reaching the same conclusions etc. (See Jansen & Compton (1988)c for more details).
- the knowledge is automatically documented and cross referenced with respect to other concepts in the domain. The documentation of the knowledge and knowledge base proceeds *in situ* with the knowledge acquisition, and not as an afterthought once the knowledge base has been built.

- the knowledge may be processed in various ways using the full power of relational calculus, thus opening up to the knowledge engineer and the expert, new and novel ways of manipulating the stored knowledge. One such process is inferencing. In fact the Knowledge Dictionary has an associated inference engine that accomplishes forward chaining inference using SQL type data manipulation rather than *resolution* (see Jansen & Compton, 1988c, Delcambre & Etheredge, 1988).
- access to the knowledge is independent of any computer programming language, thus allowing easier access to the knowledge by unskilled, in a computer sense, end user experts.
- the maintenance of the knowledge is aided by storing the knowledge in a normalized form (Howe, 1986), where each concept is fully defined and named **once only**. For example, assume knowledge acquisition discovered that expert A has a concept 'tsh high', representing the condition that the tsh level in a blood sample is above a certain threshold. Another expert has the identical concept, but labeled 'tsh is above normal'. From the view of the knowledge, this is un-normalized, in that there are two representations of the same concept and maintenance would require two changes to be effected, one to each representation. The knowledge engineer may recognize this condition, and either force one expert to use the other expert's label, or alternatively, store the definition of the concept once, and map each expert's label onto this node. In this way, the expert system has the concept in a normalized form (ie. defined once and a single label), whilst the experts are able to maintain their un-normalized form and are shown the knowledge using their own terminology. This maintains the experts' familiarity with the stored knowledge, stressed in Hayes-Roth *et al* (1983) as vital for the success of building expert systems, whilst incorporating the advantages of normalization in the maintenance process.
- as with some conventional data dictionary systems (eg. the ICL 2900 Data Dictionary System), the knowledge is related to the concepts in the business or user world, and all relationships are explicit. Thus the effects of any change in either the user world or the computer world, representing the implementation of the user world, may be ascertained prior to effecting the change.

To date the Knowledge Dictionary has been implemented in both Prolog and Hypercard on a Macintosh computer, running on the Mac Plus, Mac SE, and Mac II models, as well as in the Digital Equipment Corporation's relational database package RDB and the fourth generation software, RALLY, running on a MicroVax computer.

### **3. CAPTURING KNOWLEDGE REPRESENTATIONS IN THE KNOWLEDGE DICTIONARY**

As stated above, the Knowledge Dictionary currently enables the capturing of knowledge in the form of production rules, and the manipulation of the stored knowledge in various ways. Obviously, the other knowledge representation

formalisms must be able to be expressed as well in order for the Knowledge Dictionary to be useful in a wide variety of domains.



Work has been progressing in this area as described below.

### 3.1. Production Rules/The Existing Data Model.

As stated previously, the Knowledge Dictionary uses the Relational Data Model as the underlying storage representation for the concepts captured in the domain.

The conceptual data model is shown in figure 1<sup>1</sup>. This figure is drawn using the Entity-Relationship model formalism. It should be noted that this model is incomplete in respect to the concepts of the conventional computer system and is not the major concern here<sup>2</sup>. However, it does allow the capturing of the business information in the form of Entity-Relationship models (Chen, 1976) and Operation-Event models and shows the objects used in their implementation in the computer system. The shaded entities on this diagram are those entities added to the conventional conceptual model which allow the capturing of production rules in their constituent atomic form.

Conceptually, production rules are stored as follows. Each rule is decomposed into its constituent components, namely a *rule*, as set of *facts* and a set of *rule actions*. The constituent components are then stored in a table as a set of relationships between the *rule* and each *fact*, and the *rule* and each *rule action*. The complete rule can be built up by retrieving each tuple from the table and manipulating the names into the required formalism. This is demonstrated in figure 2 (see Jansen & Compton, 1988c for more details).

---

<sup>1</sup>Note that in this ER diagram, the arrows point from the owner object to the owning object, or the generic to the more specific.

Rule 44.

IF tsh > 7  
and tsh < 15  
THEN tsh is high

Object Name	Relationship Name	Obj
.	.	
.	.	
.	.	
Rule 44	presence	fac
Rule 44	presence	fac
Rule 44	actions	ass
.	.	
.	.	
.	.	

Figure 2. This shows the conceptual translation of a production rule into a table. The relationship names are those as shown on figure 1 between rule components.

### 3.2 Semantic Nets.

As described in Winston (1984), semantic nets represent objects, called *nodes*, and relationships between the objects, called *arcs* or *links*. Semantic nets must be differentiated from ordinary nets, in that semantic information, or knowledge, must be imparted when the net is viewed. This is demonstrated in figure 3.

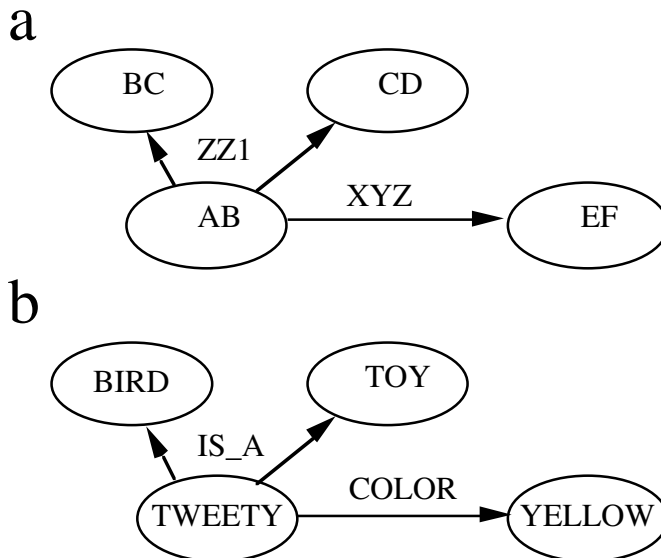
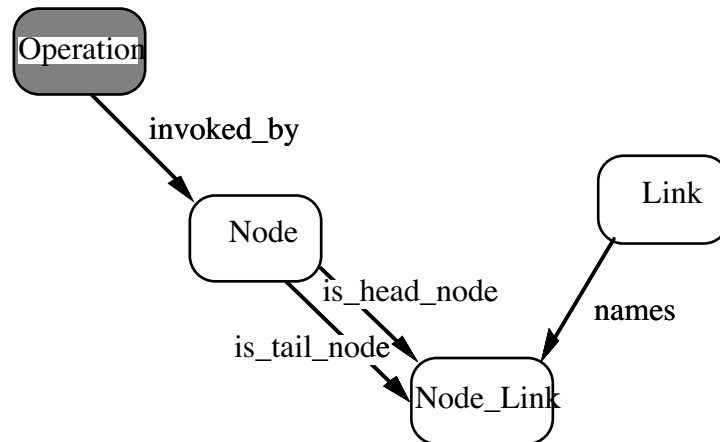


Figure 3. Two net diagrams, an ordinary net(a) and a semantic net(b). Note that the ordinary net imparts no semantic information, but may impart structural information, whereas the semantic net provides semantic knowledge about BIRDS, TOYS, and a bird/toy instance. TWEETY Note that in semantic nets, the arrows point from the specific to the generic node.

As the concepts to be captured are objects and relationships, in a similar configuration to an Entity-Relationship model, the Knowledge Dictionary can be used to capture this diagram, and hence the knowledge, by augmenting figure 1 to include the concepts as shown in the Entity-Relationship model, figure 4.



*Figure 4. Concepts added to Conceptual Model to allow the capturing of semantic nets. The figure shows the relationship between the semantic net and the shaded 'operation' entity, an existing entity on the conceptual model, figure 1. We have introduced a Node\_Link entity to represent the fact that a node may have multiple links with the same name, a situation not allowed in ER models. Two nodes may not be linked by multiple links of the same name however. Note this diagram is in Entity-Relationship form.*

Figure 4 shows one importance difference between our semantic net representation, and that espoused by Winston (Winston, 1984). This difference is important, in that it recognizes that the semantic net diagram should be in a normalized form. In figure 5a, the semantic link 'weight (if needed)' is not normalized in that it implicitly defines two concepts within the one link, namely a *weight* for a block, and the existence of the *weight calculation procedure* if the weight is needed. In our estimation, these two concepts should be explicitly defined, as shown in figure 5b, where we draw a node for the existence of a 'weight' for the block, and a separate node, attached to the weight node, representing the existence of the 'weight calculation procedure'. This difference is important when it comes to the representation of the semantic node as a frame, as in the Winston form, it is possible for a slot to have many facets, whereas in our representation, there is a single facet to a slot, although the facet may have more than one value.

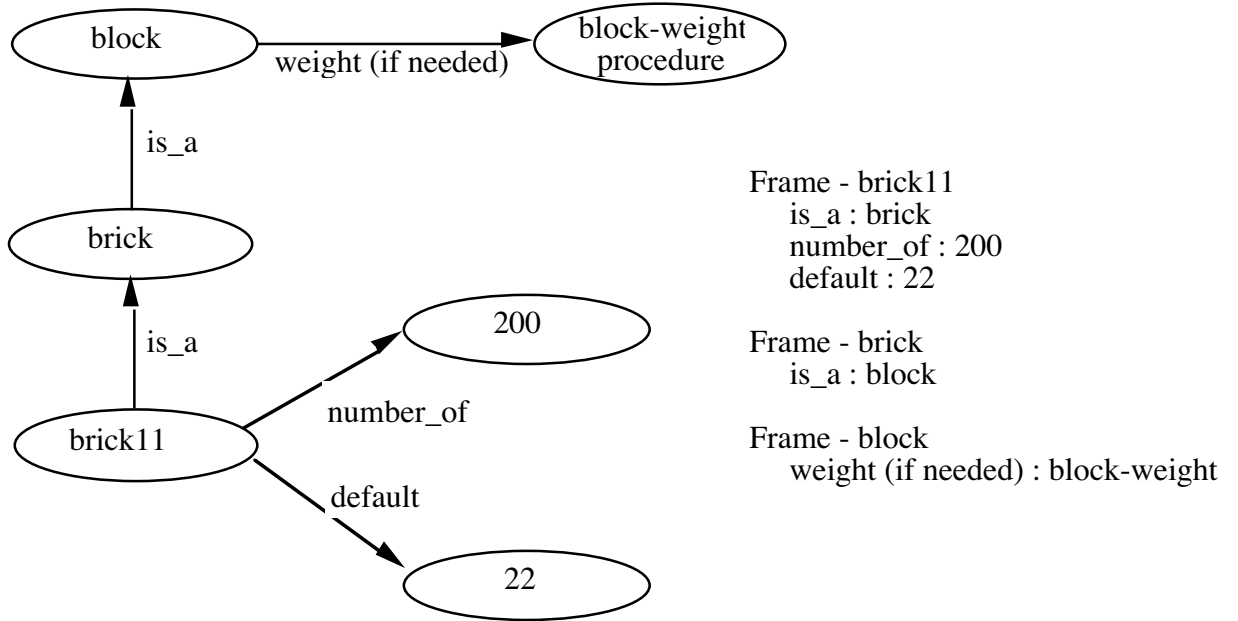


Figure 5a - this diagram shows a semantic net of the form described in Winston 84. It details a non-normalized link, 'weight (if needed)' between the nodes 'block' and 'block-weight procedure'. The equivalent frame representation of the diagram is also shown.

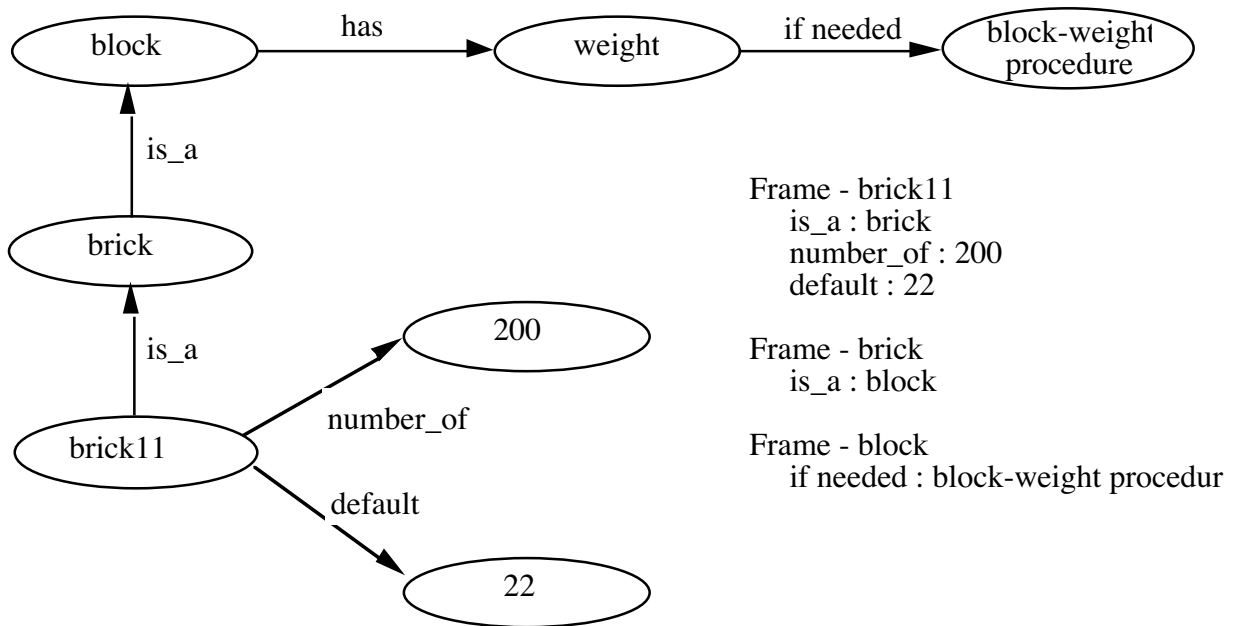


Figure 5b - this shows the 'normalized' representation of the 'weight (if needed)' link from figure 5a. This representation is preferred, in that it explicitly recognizes that there are two concepts, namely a weight and a procedure to calculate it if required. The equivalent frame representation of the diagram is also shown.

The semantic net in figure 3b would conceptually be represented in table entries as shown in table 1.

<u>Node Entries</u>	<u>Node Link Entries</u>	<u>Link Entries</u>
Tweety	Tweety:is_a:Bird	is_a
Bird	Tweety:is_a:Toy	colour
Toy	Tweety:colour:Yellow	
Yellow		

*Table 1. This table shows the conceptual representation of the simple semantic net in figure 3b.*

The above model can be shown capable of representing the mechanisms of *inheritance*, *demons*, *defaults*, and *perspectives*, described in Winston (1984), as detailed below.

*Inheritance* is the term used to describe the facility of passing down some or all of the properties of the more generic node to a more specific node. This is analogous to the database concept of subtyping. The inheritance mechanism is not an explicit part of the diagram itself, but a manipulation of the knowledge on the diagram, represented by the links between nodes. Thus inheritance is not visible either in the Knowledge Dictionary representation. In figure 3, the node TWEETY could inherit the properties of the nodes BIRD and TOY. This scenario makes sense of the statements that 'TWEETY is\_a BIRD', and 'TWEETY is\_a TOY'.

The *demon* mechanism is a mechanism for automatically performing an action in the event of a specific situation. For example, the weight calculation procedure, shown in figures 5a and 5b would be implemented as a demon if the weight of the block is required. The demon mechanism is represented and documented using the *invoked\_by* relationship between an *operation* and a *node* entity.

The *perspective* mechanism is a method of recognizing that the semantics of an object may vary depending on the current point of view of the observer. This allows us for example, to describe a person as happy when on holidays, but unhappy when at work. Thus the properties associated with the object may vary depending on the current perspective. This mechanism is represented by a semantic link between the perspective nodes and parent node, in the same way as other links.

*Figure 6 - Diagram showing how the perspective mechanism is handled in the semantic net representation by the 'has\_perspective' links. These links are treated as any other link by the model shown in figure 4.*



### 3.3. Frames.

Winston (1984) states that a *frame* is a collection of semantic net nodes and slots (or *link* in figure 4) that together describe a stereotyped object, act, or event. This statement implies that, given that a semantic net may be represented in the Knowledge Dictionary as shown in figure 4, then a frame can be considered a transformation of the information from the semantic net representation. This is analogous to the case of the IF...THEN... representation of a production rule, which is a transformation of the information stored within the Knowledge Dictionary, as shown by the shaded entities on figure 1.

Cuadrado & Cuadrado (1986) introduce frames in a simple and understandable way shown in figure 7 using Backus-Naur notation.

```
Frame      ::=  frame_name slot ...
Slot       ::=  slot_name facet...
Facet      ::=  (facet_name value) ...
           ::=  demon_name
Frame_name ::=  character string
Slot_name  ::=  character
Facet_name ::=  character
Demon_name ::=  module_name
Value     ::=  number
           ::=  character
```

*Figure 7. Backus-Naur representation of a frame as described in Cuadrado & Cuadrado 86*

Comparison of this representation with the semantic net representation shown in figure 4, leads to the conclusion that the frame representation can be generated from the semantic net representation, recognizing the *one-to-one* correspondence of 'node's to 'frame's, and the *one-to-many* correspondence between 'slot's and 'node\_link's.

Thus to present a frame based interface for a user, the Knowledge Dictionary would have to be front ended with appropriate graphic based translation code, a task independent of the major software engineering of the system. To allow frame based processing by a commercially available frame based knowledge based system would

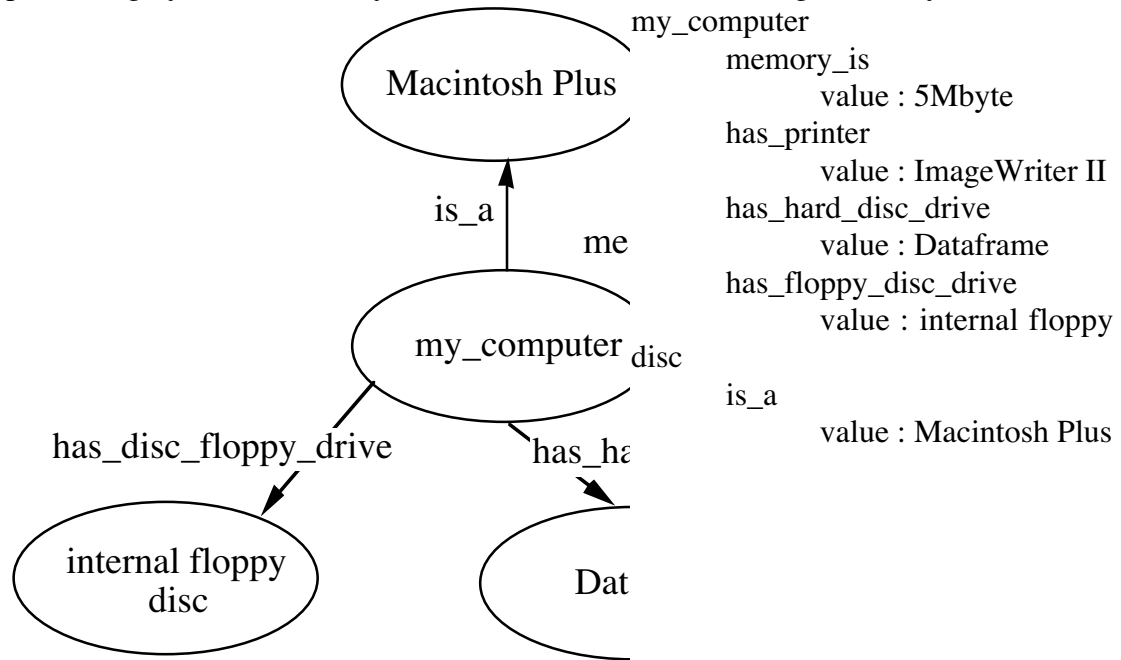


Figure 8. A diagram showing on the left a semantic net representation and on the right a frame based representation for equivalent information about 'my computer'.

require a generator to translate the semantic net representation within the Knowledge Dictionary to the required data structure in an external file<sup>3</sup>.

Figure 8 shows the semantic net representation represented in the Knowledge Dictionary and a generated frame based representation. Note that the semantic net representation is a conceptual representation of the actual underlying storage structure as implemented in the current Knowledge Dictionary. See Jansen & Compton (1988)b and Jansen (1989) for a details.

The similarity of frames and semantic nets is a useful feature when considering the generation of frame structures, especially if one classifies the semantic net as the user view of the knowledge, and the frame as the computer implementation of the knowledge. This classification allows the computer implementation of the knowledge to be hidden from the users or experts, but visible to the knowledge engineer. However, if the knowledge is represented naturally in a frame representation, then the expert should not be forced into manually translating to a semantic net representation first. If the semantic net representation is required, then this should be capturable automatically from the frame descriptions. On the other

<sup>3</sup>It is planned that the Knowledge Dictionary will have a generator capable of generating any defined representation and its own frame based knowledge based system. The translation tables, or grammar of the required representation would be stored as meta knowledge within the Knowledge Dictionary. Currently the generation

hand, if the expert's representation is in the semantic net form, then automatic tools should be available to generate the appropriate frame structure in the dictionary from the semantic net description. Thus the Knowledge Dictionary must be capable of

capturing semantic net descriptions directly, coupled with a 'semantic net to frame' translator facility.

and the capturing of frames directly, coupled with a 'frame to semantic net' translator facility.

This may be seen as analogous to the relationship between *entities* and *records* in conventional data structures. A record description is able to be generated from an entity description, but the dictionary still has a *record* entity to describe record structures separately. This is done to aid the maintenance task, where it may be important to know, what modules access a record type. This complies with the basic dictionary philosophy of representing the user world **and** its implementation in the computer world. Thus an *entity* is an object in the user world, whereas a *record* is an implementation in the computer world. Similarly, a *node* is a representation in the user world, whereas a *frame* is a representation in the computer world. Thus extra entities must be added to the conceptual diagram, figure 4. The amended entity diagram is shown in figure 9.

A major benefit in using the Knowledge Dictionary can be seen in this case, as the demon facility can now be linked to a 'rule' in a rule based system, thus invoking for example, a forward chaining operation starting from the related rule. Thus, a slot in a frame can cause the execution of part of a production rule expert system, instead of, as in most existing frame based systems, a module of executable code. This is a significant advance in the integration of two knowledge base representations.

#### 3.4. Ripple-Down Rules

The concept of *Ripple-Down Rules* (Compton & Jansen, 1988) was developed in an attempt to capture the context of a particular rule. This came about as a result of the realization that diagnostic rule-based systems undergo maintenance because the expert verbalizes a rule within the context of a currently failing case. The expert system then attempts to apply this new rule across the complete domain whenever possible, whereas we surmise that the new rule should only be used if the current area of investigation is

the rule that initially failed and the current profile matches that given by the expert as the reason for having a 'fix-up' rule.

Completely new rules have the context that none of the previous knowledge was applicable so they considered after all the earlier knowledge has been considered.

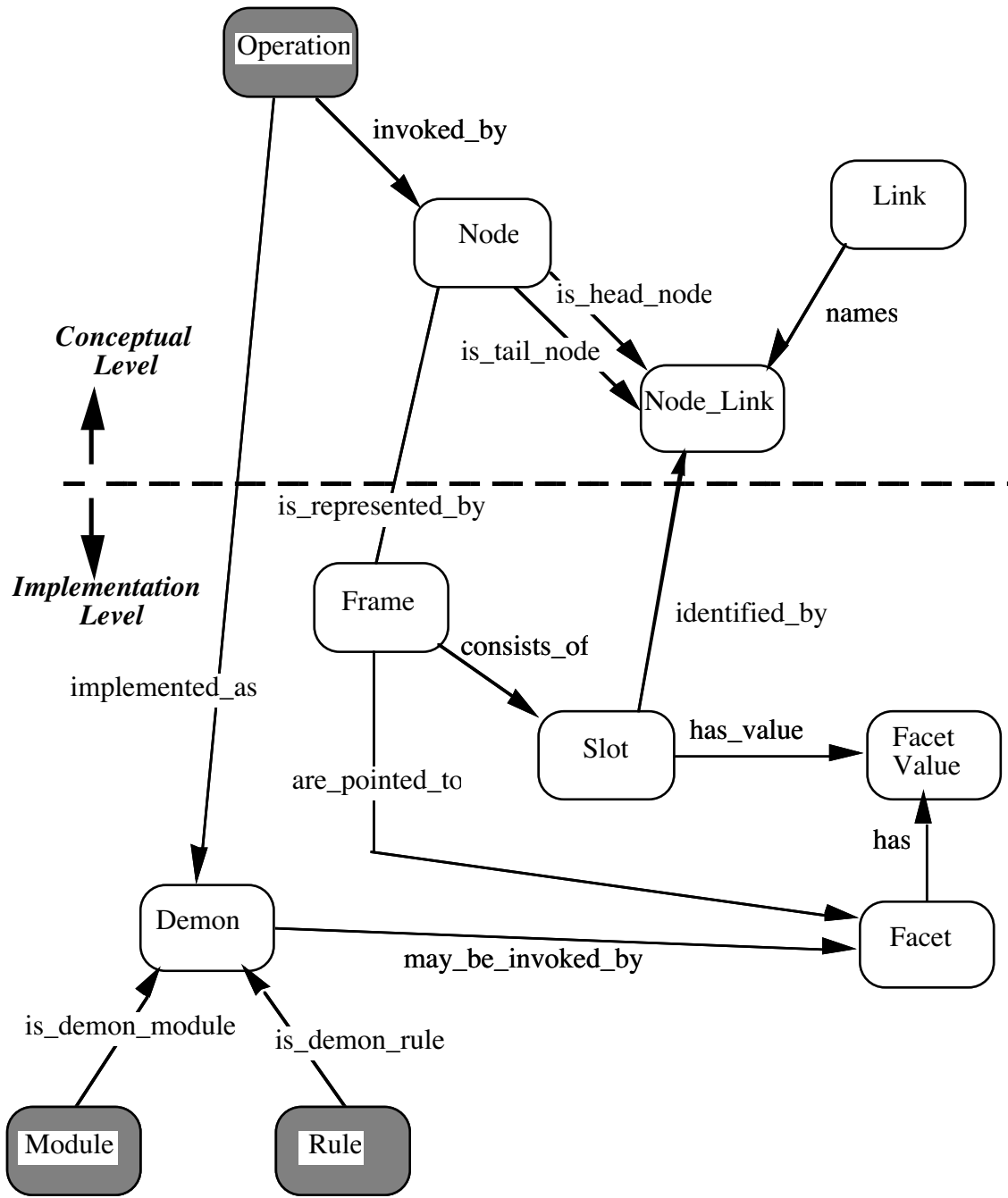


Figure 9. This figure shows the conceptual structures added to Figure 1 to detail the representation of a semantic net and its implementation as a frame in the computer world. Note the relationships to existing entities 'operation', 'module', and 'rule'. There is a 'mutual exclusivity' constraint between the 'are\_pointed\_to' and 'may\_be\_invoked\_by' relationships relating to a 'facet'. A 'facet' may either point to a frame, or invoke a demon, but not both.

```

RULE(1.46)
IF  LAST_FIRED(0)
   and T3 is high
   and TSH is high
   and fti_normal
   and antithyroid
THEN DIAGNOSIS(".... may occur after antithyroid medication")

RULE(2.32)
IF  LAST_FIRED(0)
   and fti_low
   and T3 is low
THEN DIAGNOSIS("..... consistent with the sick
euthyroid state.")

RULE(3.01)
IF  LAST_FIRED(0)
   and T3 is high
   and fti_high
   and hyperthyroid
THEN DIAGNOSIS(".... consistent with thyrotoxicosis")
.
.
.
.
RULE(25.11)
IF  LAST_FIRED(2.32)
   and TSH is high
THEN DIAGNOSIS("..... consistent with primary
hypothyroidism.")

RULE(27.01)
IF  LAST_FIRED(7.03)
   and antithyroid
THEN DIAGNOSIS("..... consistent with toxicosis")
.
.
.
.
RULE(40.19)
IF  LAST_FIRED(18.16)
   and TSH_BORD is high
   and T3 is low
THEN DIAGNOSIS(".... non-thyroidal illness.&
compensated hypothyroidism")

RULE(41.15)
IF  LAST_FIRED(25.11)
   and sick
THEN DIAGNOSIS(".... hypothyroid & concurrent
non-thyroidal illness.")
.
.
.
.
RULE(79.14)
IF  LAST_FIRED(2.32)
   and TSH is missing
   and hypothyroid
   and sick
THEN DIAGNOSIS(".... Need TSH to distinguish
hypothyroidism and sick-euthyroidism")

```

Figure 10. This shows some of the rules in the Garvan-ESI system. The links between rules indicate where rules were added to correct an interpretation given by an earlier rule. The rules are stored as a single long list, with new rules added to the

*end of the list. The gaps in the table indicate that rules have been omitted from this extract. The decimal rule numbering system encodes extra information not relevant here.*

*From Compton & Jansen (1988)*

Ripple\_down rules were so named because instead of fixing an existing rule, the knowledge engineer appends a 'fix up' rule to the knowledge base. This fix up rule contains the condition that the last rule to fire was a particular rule. Figure 10 demonstrates the principle.

For a more detailed description of the Ripple\_Down Rule mechanism, see Compton & Jansen (1988).

The implementation of the ripple down rule mechanism in the Knowledge Dictionary is very simple. As can be seen by studying figure 10, the ripple-down rules contain a condition LAST\_FIRED(..), where the number in brackets indicates the rule number of the last rule that must have fired. This gives the rule the context, that unless the last rule that fired has the rule number identical to that in the LAST\_FIRED (..) condition, the new rule can not fire. Thus the LAST\_FIRED(..) condition is treated as an extra fact in the premise of the rule.

Facts are represented in the Knowledge Dictionary by the existence of a *fact* object type. The premise of a rule then is a set of instantiations of *fact* object types, linked to the owning *rule* by either the *presence* or *absence* relationships.

Thus one method of representing the LAST\_FIRED(..) condition is to instantiate an extra fact object type, and tie it into the truth maintenance system. This approach, although workable, is more complex than that required.

The simple solution is to define an extra relationship called *last\_fired* from a *rule* object type to another *rule* object type. This is shown in figure 11.

The only other change required is to the generate function, generating IF...THEN production rules for use by the Garvan-ES1 expert system. This function was enhanced to test for the *last\_fired* relationship, and if it existed, to include a LAST\_FIRED(..) condition in the rule's premise.

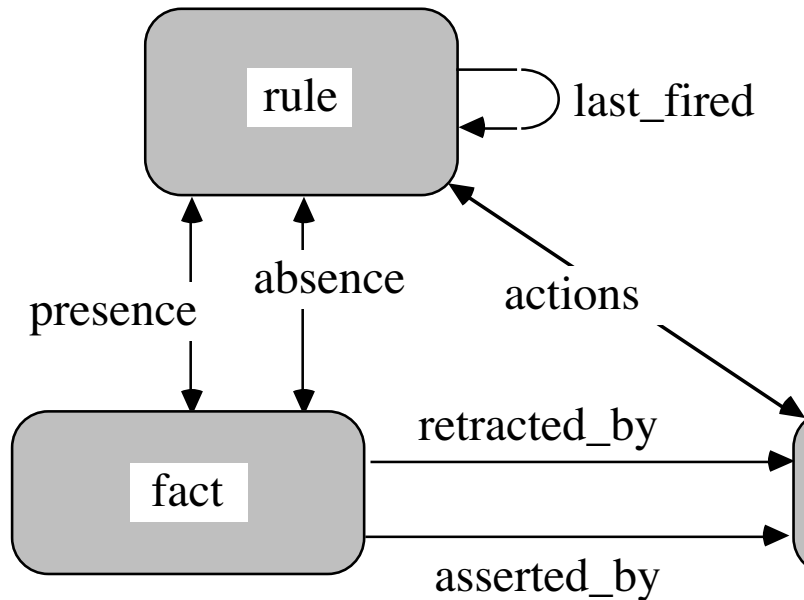


Figure 11. A subset of the conceptual model of the Knowledge Dictionary, shown in figure 1, for a production rule utilizing the Ripple-Down rule approach. The addition is the LAST\_FIRED relationship between rule object types.

#### 4. KNOWLEDGE ACQUISITION AND KNOWLEDGE REPRESENTATION

The above discussions have focussed on the area of different knowledge representations, and suggesting underlying data structures enabling them to be implemented in the Knowledge Dictionary environment. However, it appears that knowledge representation plays an important part in the knowledge acquisition process as well.

Compton & Jansen (1989) discusses the problems of knowledge acquisition, and their relation to philosophical questions on the nature of knowledge. These views, in conjunction with the problems being discovered in the knowledge acquisition process, highlight that knowledge should be seen as a complex resource whose structure and meaning depend on the current context of the observer. Shaw (1988) points out that the knowledge acquisition process involving multiple experts is complicated by the fact that experts use of the same labels for different concepts or different labels for the same concepts. We suggest that this infers that they utilize different structures for the representation of their knowledge, and in particular we maintain that the expert's knowledge structure at any time depends on acquisition and representation context. Thus, given the complexity of knowledge, how does a knowledge engineer begin and guide the knowledge acquisition process to a successful conclusion?

From experience gained with the redevelopment of SIRATAC and GARVAN ES1, it appears that the visualization of the knowledge in suitable representations (Rappaport & Gaines, 1988) is a suitable vehicle for this guidance process, in a similar way that a system's analyst will draw several versions of entity-relationship models for a domain being implemented in a database regime as part of the analysis

process. These modelling formalisms only act to draw the analyst's attention to logical subsets and relationships (or focal points) in the knowledge. These focal points may subsequently be used to aid and manage the systematic extraction of the knowledge from the experts.

The production of representation schemes as part of knowledge acquisition should not be seen as a holy grail, but rather as working documents that are liable to change as the knowledge acquisition proceeds. Further knowledge will alter focii and the possible relationships established in the already captured knowledge. At any point in time, the knowledge we have is the best currently available, but its properties will change with the introduction of new knowledge.

In order for representation to guide acquisition, the knowledge engineer requires flexible tools to visualize the knowledge in various levels of detail; to be able to alter the focus of the current view of the knowledge; and to be able to simply change between representations to suit the current focus. To accomplish these tasks, models of the different knowledge representation formalisms are required, supported by tools capable of performing the translation and visualization tasks. The Knowledge Dictionary is a tool aimed at supporting these types of operations.

## 5. SUMMARY

In this paper, we have discussed some of the problems that exist for the knowledge engineer and expert in implementing the domain knowledge for a knowledge based system. These problems included documentation of the expert system, variety of knowledge representation formalisms and the lack of one suitable formalism. We presented the results of work being carried out in the area of the software engineering of knowledge based systems, especially the application of data dictionary concepts the knowledge base area. We described an environment called the *Knowledge Dictionary* which consists of an augmented data dictionary able to represent, document and cross reference knowledge in the form of production rules. Having entered the knowledge in the *Knowledge Dictionary*, the knowledge is accessible by various techniques not normally catered for in current expert systems, techniques analogous to those that have proved invaluable in the long time support of conventional computer systems. We then went on to demonstrate the power of the *Knowledge Dictionary* environment and the relational data model supporting the underlying representation, by showing how three other knowledge representation formalisms could be included in the *Knowledge Dictionary* environment. In doing so, the knowledge engineer could chose an appropriate representation for either the whole domain knowledge or a part of the domain. In the latter case, each representation was linked into the complete domain, so an overall picture could be obtained.

This paper ends up in a discussion regarding the relationships between knowledge acquisition and knowledge representation, and proposes that knowledge representation, in a way analogous to the drawing of ER models as part of conventional analysis, is a technique that may be used to guide and focus the acquisition process. To be successful in this task, the tools used must be capable of

representing knowledge in various formalisms, and be capable of translating between them. This paper treats only four formalisms, production rules, semantic nets, frames, and ripple down rules. The modelling of other formalisms is the subject of further research. The tasks of translation between knowledge representations, and the generation of the run time knowledge base have not been treated in any great details, and although recognized as significant, present purely technical challenges.

## 6. ACKNOWLEDGEMENTS

The research results described in this paper are part of a collaboration project between the CSIRO Division of Information Technology and the Garvan Institute of Medical Research. We acknowledge the advice and guidance of the following people.

Bob Colomb, Neil Ashburner and Craig Lindley of the CSIRO Division of Information Technology.

Leslie Lazarus and other domain experts of the Garvan Institute of Medical Research.

## 7. REFERENCES

Al-Zobaidie A & Grimson J B, *Expert Systems and Database Systems: How can They Serve Each Other?*, Expert Systems, February 1987, Vol. 4. no. 1

Chen P. P, *The Entity-Relationship Model: Towards a Unified View of Data*, ACM TODS 8, 1 March 1976, pp 7-42

Codd E F, *A Relational Model for Large Shared Data Banks*, CACM, Vol. 13, No. 6, 1970.

Compton Paul, Horn Kim, Quinlan J R, Lazarus L, Ho K, *Maintaining an Expert System*, Proceedings of the Fourth Australian Conference on the Applications of Expert Systems, Sydney, 1988, pp 110-129

Compton Paul & Jansen Bob, *Knowledge in Context: A Strategy for Expert Systems Maintenance*, proceedings of AI'88 Conference, Adelaide, November 1988

Compton Paul & Jansen Bob, *A Philosophical Basis for Knowledge Acquisition*, CSIRO Division of Information Technology, Technical Report TR-FD-89-01, January 1989.

Cuadrado John L & Cuadrado Clara Y, *AI in Computer Vision*, Byte January 1986, pp 237-258

Davis Randall & King Jonathan J, *The Origin of Rule-Based Systems in AI*, in *Rule-Based Expert Systems, the MYCIN Experiments of the Stanford Heuristic*

*Programming Project*, Bruce G. Buchanan & Edward H. Shortliffe (eds), Addison-Wesley, 1984, pp20-52

Delcambre Lois M. L. & Etheredge James N., *The Relational Production Language: A Production Language for Relational Databases*, Expert Database Systems. Proceedings of the Second International Conference on Expert Database Systems, Kerschberg (ed) April 1988

Dolk Daniel R & Kirsch II Robert A , *A Relational Information Resource Dictionary System*, Communications of the ACM, January 1987, Volume 30, Number 1

Dolk Daniel R, *Model Management and Structured Modelling: The Role of an Information Resource Dictionary System*, Communications of the ACM, Volume 31, Number 6, June 1988

Hayes-Roth Frederick , Waterman Donald A & Lenat Douglas B (editors), *Building Expert Systems*, Advanced Book Program, Addison-Wesley Publishing Company Inc. 1983

Hearn B., Brook K., Ashburner N., Colomb R., *SIRATAC A Cotton Crop Management Expert System*, Proceedings of the 1st Australian Artificial Intelligence Congress, Melbourne, 1986

Horn K A, Compton P, Lazarus L, Quinlan R, *An Expert System for the Interpretation of Thyroid Assays in a Clinical Laboratory*, The Australian Computer Journal, Volume 17, No 1, February 1985

Howe D R , *Data Analysis for Data Base design*, Edward Arnold (Publishers) ISBN 0-7131-3481-X, 1986

Jansen Bob, *Applying Software Engineering Concepts to Rule Based Expert Systems*, in AI & Software Engineering, Derek Partridge (ed), Ablex, 1989, in press

Jansen Bob & Compton Paul, *The Knowledge Dictionary: An Application of Software Engineering Techniques to the Design and Maintenance of Expert Systems*, Proceedings of the AAAI-88 Workshop on Integration of Knowledge Acquisition and Performance Systems, August 1988, Minnesota USA

Jansen Bob & Compton Paul, *The Knowledge Dictionary: A Relational Tool for the Maintenance of Expert Systems*, Proceedings of the International Conference on Fifth Generation Computing Systems (FGCS'88), Tokyo, November 1988

Jansen Bob & Compton Paul, *The Knowledge Dictionary : A Data Dictionary Approach to the Maintenance of Expert Systems*, Knowledge- Based Systems, John Gero (ed), Vol 2, No. 1, March 1989, in press

Kaiser Gail E, Barghouti Naser S, Feiler Peter H & Schwanke Robert W , *Database Support for Knowledge-Based Engineering Environments*, IEEE Expert, Summer 1988

- Lister R , Asli K , Buda R, Horsfall C, Buntine W, *GOLD - an Expert System for Mineral Identification from Reflectance Spectra*, AI'87 Conference Proceedings, Sydney 1987
- Minsky Marvin, *A Framework for Representing Knowledge*, Cambridge MA: MIT AI Memo #306, 1974.
- Rappaport A T, Gaines B R, *Integrating Knowledge Acquisition and Performance Systems*, Proceedings of the Australian Joint Artificial Intelligence Conference (AI'88), Adelaide, 1988, pp317-336.
- Shaw Mildred L G, *Validation in a Knowledge Acquisition System with Multiple Experts*, Proceedings of the International Conference on Fifth Generation Computer Systems 1988, Tokyo, 1988, pp 1259-1266
- Sowa John F, *Conceptual Structures*, Addison Wesley (pub), 1984
- Winston Patrick Henry, *Artificial Intelligence*, Addison-Wesley (pub), 1984.
- Zaniolo Carlo, Ait-Kaci Hassan, Beech David, Cammerata Stephanie, Kerschberg Larry & Maier David, *Object Oriented Database Systems and Knowledge Systems*, Expert Database Systems Proceedings from the first International workshop, Larry Kerschberg (ed), 1986