

Knowledge Acquisition without Analysis

European Knowledge Acquisition Workshop 1993. Springer Verlag

Paul Compton, Byeong Kang, Phillip Preston, Mary Mulholland[†]

School of Computer Science and Engineering and [†]School of Chemistry,
University of New South Wales,
PO Box 1, Kensington NSW, Australia 2033

Abstract. This paper suggests that a distinction between knowledge acquisition methods should be made. On the one hand there are methods which aim to help the expert and knowledge engineer analyse what knowledge is involved in solving a particular type of problem and how this problem solving is carried out. These methods are concerned with classifying the different types of problem solving and providing tools and methods to help the knowledge engineer identify the appropriate approach and ensure nothing is omitted.. A different approach to knowledge acquisition focuses on ensuring incremental addition of validated knowledge as mistakes are discovered (validated knowledge here means only that the earlier performance of the system is not degraded by the addition of new knowledge). The organisation of this knowledge is managed by the system rather than the expert and knowledge engineer. This would seem to correspond to human incremental development of expertise. From this perspective task analysis is a secondary activity related to explanation and justification not knowledge acquisition. Ripple Down Rules is a limited example of this approach. The paper considers the possibility of extending this approach to make it a more generally applicable.

1 Models and Tasks

Knowledge based systems is an area where analysis of what the discipline is concerned with has followed practical development. Expert systems were being built before there was a well developed theory of what they were about. The initial analyses were very simple, e.g. forward and backward chaining, model-based versus heuristic etc. Newell proposed that one could step back from the symbolic representation level for systems and see them at a higher level one: should ask "why" questions about the expert rather than "how" questions [50]. Clancey made clear what practitioners knew but had not made explicit: that the world divides into classification and configuration tasks [10]. He also developed a detailed analysis of a major type of problem solving, heuristic classification. Chandrasekaran enumerated a wider range of tasks [4, 5] in terms of which problem solving could be

described. In a different approach Clancey has attempted to elucidate all the control knowledge or model construction operators that may be implicit in an expert system [14]

The culture of early expert systems, and probably still the prevailing culture in every day practice is that in building an expert system one attempts to extract or mine the knowledge from the experts head. It has become very clear that this is not in fact the case. Clancey [8, 14] proposed that all knowledge based systems, whether they were based on heuristics or traditional ideas of a model, are really qualitative models of some aspect of the world. Hence knowledge engineering becomes not an attempt by the knowledge engineer to find what the expert was really doing in their mind but a cooperative effort between expert and knowledge engineer to build a model of part of the world using the expert's expert knowledge and understanding about what happens in the world and the knowledge engineer's skill in building and testing models.

This general outlook culminates the task based approach to building KBS. This is a far from homogenous field and it has a variety of ancestries, but all of these approaches can be characterised as second generation attempts to step back from building an expert system to understand more clearly what tasks one is involved in. This enables these tasks to be approached more rationally and one can check that all the relevant issues for an identified sub-task have been covered and so guide the selection of tools appropriate for the task. In Chandrasekaran's generic task approach it is claimed that there a group of tasks and hierarchy of sub-tasks in terms of which all problem solving can be understood (for an example see [1]). These tasks are specified at a fairly high level. The KADS approach is related, but the tasks are specified in more detail, so that there may or may not a task available by which a particular problem solving activity of an expert can be understood [59]. KADS however is more general as it explicitly considers all aspects of the problem, for example specifying the domain model[63]. It also specifies the application of a task method to the problem in domain independent terms. The details of how the different approaches vary is beyond the scope of this paper. However they are all attempts to set up a frame work in which one can approach knowledge acquisition in a systematic way, so that nothing gets left out and one understands where one is up to. The second stage of this approach is to provide tools to facilitate the development. Work includes[61, 49, 62, 55] amongst many others.

These methods are obviously useful and successful. There can be no doubt that a disciplined approach that helps you select the best approach and take everything into account, will provide benefits. It appears that the KADS approach has been used in perhaps hundreds of applications in Europe. The clearest demonstration of the virtue of the general approach is in the SPARK BURN FIREFIGHTER work of Digital. Here one has basic tools to build more specific tools to finally build the application system [44]. This approach has resulted in remarkable decreases in the time taken to develop systems. It is not clear whether such results will be found universally or whether the systems built were particularly suitable for the approach; however, it is obvious that large gains can be achieved.

The various task approaches emphasise a principled decomposition of expert problem solving. In practice the careful principled approach may not be necessary.

Rappaport suggests that it doesn't really matter much whether individual problem solving tools are ideally suited to a problem, the power will come from having a large enough range of user friendly tools fully embedded in systems with excellent communication between the tools[56].

The KSSn system of Gaines. [30] probably exemplifies Rappaport's claim. The system includes tools to carry out repertory grid analysis, induction, a subsumption based knowledge representation language, tools to enable the knowledge to be constructed graphically, an inference engine and an integrated wordprocessor for reports. One can move freely between these different tools to build and explore a knowledge base and related set of cases. Gaines partial claim in the Sisyphus experiments, in contrast to the more formal task approaches, is not that the KSSn tools provide a comprehensive way of assisting one to think about the domain correctly, but that they make things sufficiently easy that one develops a good generic solution to the problem anyway[28].

2 Knowledge as Justification

Implicit in the modelling approach to knowledge based systems is the realisation that the knowledge in expert systems does not represent the knowledge in an expert's head, but rather it represents the system in the world being modelled. Other work has reached the same conclusion that knowledge acquisition from experts was not based on experts reporting on what was going on inside their mind. Rather than explain how they reached a conclusion experts justify that their conclusion is correct [20, 23]. These justifications were tailored to the specific contexts in which they were given. The justification the expert gives is directed towards what the expert guesses are the concerns of his or her questioner and attempts to address those concerns. For example if the questioner appears to believe that a deep scientific explanation of reasoning is required the expert will explain his reasoning in those terms. If the questioner believes that judgements are founded on past experience, the explanation will be couched in these terms. Again the justification will vary depending on the expertise of the questioner. The knowledge extraction understanding of knowledge acquisition would assume that the more complete or complex explanation will be given to a naive questioner rather than to a person who is already fairly expert. In fact the converse is generally true. It is more easy for an expert to justify their judgement to a lay person with naive concerns than to another expert.

One conclusion from this analysis is that knowledge, justification in context, is constructed. This fits closely with the knowledge bases as qualitative models perspective discussed earlier. The philosophical implication of this position is that all knowledge is constructed, a created artefact and that no knowledge can ever be "true". Knowledge has to be taken in conjunction with insight [41]. We use knowledge to try and express our insights into reality and constantly require insight to recognise how our knowledge makes sense of reality[16]. This applies to all knowledge from sense knowledge to expertise [16, 23]. This analysis also relates to the philosophy of Popper in which it is impossible to prove an hypothesis true, rather one progresses in knowledge by disproving or falsifying hypotheses[53]. Popper is best known for applying this to scientific development but in fact applied his falsification approach to all knowledge. His work thus relates to that of Piaget who

analysed early childhood development in terms of children forming and testing hypotheses, often initially quite bizarre, in terms of an adult world view [52]. Other related ideas can be found in Wittgenstein's later work[65]

Clancey in other work has proposed related views, for example [6, 7, 11-14] Others such as Winograd and Flores have provided similar but more general critiques[64].

3 Tasks and philosophy

It has been argued elsewhere that the philosophical ancestry of the knowledge extraction approach to knowledge acquisition is Platonism[16, 23]. The reductionist assumption that one should be able to dig deep enough to find primitive concepts and the relationship between them on which knowledge is built finds its origins in Plato's concept of archetypes. That is, that there exist (literally) archetypes for all the things in the world and the concepts we use and the objects in the world are just faint reflections of the archetypes. Proposals such as the knowledge principle [40] and the physical symbol hypothesis [51] are essentially statements of belief that if the archetypes and relevant logical relationships can be found and manipulated intelligent thought can be reproduced. This is perhaps a simplification of the physical symbol hypothesis, but it seems fair to claim that it expresses how the physical symbol hypothesis is usually understood. Although this finds its origins in Plato, Plato would probably disown it as his concerns were the converse and his development of the archetypes was an attempt to explain insight: to explain how we understand what things are. His question of how concepts are linked to reality is of esoteric concern to knowledge based system researchers. Despite the partial recognition that knowledge bases are models and the philosophical views this implies, the prevailing view is that intelligence is only the manipulation of concepts.

A version of the physical symbol hypothesis which fits more easily with the analysis of knowledge engineering above is rather that symbols and their relationships can be used to express and manipulate anything that can be humanly expressed and manipulated. The physical symbol hypothesis is not concerned with intelligence per se but with the discourse that arises from intelligence. Clancey has emphasised that knowledge is not what we have in our heads but what we express as a result of intelligent activity [11], i.e. knowledge only exists in its expression. It has been claimed [11, 23] that the problems with building KBS are largely due to incorrect philosophical assumptions. Others have used the same approach to argue for the futility of trying to build expert systems [24]. The argument here is rather that different philosophical assumptions suggest other approaches to building expert systems.

The question arises of whether the task based approach as outlined above is just a new version of the Platonic approach to knowledge engineering or really represents a new approach to dealing with knowledge. Opinions vary with the KADS authors strongly disagreeing with the philosophical outlook above as presented by Clancey [58], although Clancey has made major contributions to the notion of tasks.[10, 14]. The conventional approach to knowledge engineering has been to emphasise that what the expert does in problem solving is important and go in search of this. As we have argued this search is always frustrated as the expert only justifies their

conclusions, not explains how they are reached. The emphasis in the task approach is that the expert's way of organising their knowledge, organising their problem solving is important. What the task approach does is to say that you need more framework to more easily identify what the expert is doing. The task people might prefer to put it that you help the expert to find the appropriate way to model the problem solving that her or she carries out. There are a number of problems with either version of this idea. Apart from the philosophical analysis already presented, the most obvious problem is that the number and nature of the tasks is perhaps even more indeterminate than the number of sub atomic particles. The tasks one identifies depend on one's starting point and philosophical framework and there appears to be no clear mapping between the tasks identified in the different approaches. Gruber is attempting to translate between different ontologies, a related problem, but readily acknowledges that there is no guarantee that this is universally possible [32].

It is clear that one cannot claim that an expert in his problem solving carries out specific types of tasks. Just as knowledge bases are models, so too task analysis is an attempt to model human problem solving. However, if the various task analyses are just ways of modelling human problem solving is it important to use them? They obviously have practical advantages over raw attempts at knowledge engineering, but they still have many of the difficulties of knowledge engineering. Allemang notes that generic task analysis is difficult and requires a knowledge engineer [1]. Marques et al [44] similarly note that experts are enmeshed in the details of using their skills and find it difficult to understand what they are doing in more abstract terms. Any reading of the KADS literature suggests that one requires a high level of skill to use the approach. Are we adopting such difficult approaches because of their superiority over unstructured methods, or is part of the motivation the philosophical baggage of the Platonic knowledge extraction approach to knowledge engineering? Perhaps we may say that we are no longer motivated by finding out how the expert solves a problem, rather we are concerned with choosing the right way to solve a particular problem. The emphasis seems to be still on the notion of archetypes. We are no longer trying to get down to the right symbols and relationships, just to the right task analysis (and the other related analyses).

Clearly this has a different emphasis from the justification approach above. The justification approach claims that experts cannot report on how they solve problems, but they will attempt to justify their problem solving in whatever frame work they are questioned. This allows them to work in the task framework, but if this is really just a post factum justification or explanation of the problem solving that the expert does, why should problem solving by the knowledge base system have to be developed and carried out in these terms? Surely the problem is how to handle the simple justifications the expert provides for individual decisions rather than to force the expert to create justifications in terms of some global problem solving framework.

4 Aim

The real long term goal of knowledge based system technology is systems that can learn to carry out problem solving from the justifications that humans provide and then explain what they are doing in terms of generic tasks or any other framework that is required. This is the conventional aim of knowledge based systems research:

systems that can perform at an expert level and explain themselves. Here however, it is part of the explanation facility of the system that it can explain itself in terms of tasks, or at other levels in terms of "deep" models, heuristics etc. There is no importance in having the system solve the problem in a particular way, the importance is in having the system able to justify itself in what ever framework is required. This is what people do - they solve problems and then justify (well or badly) what they have done from whatever angle the context requires. The goal of KBS should be to emulate this ability. Gruber makes a similar point in arguing for Ontolingua [32].

The initial focus of knowledge acquisition was to assist the expert elucidate how they solved a problem. Because of the difficulty of doing this and the result that too much problem solving was implicit [9, 14], we now try to get the expert to elucidate what they are doing within a particular framework. What we really want is for the expert to have a dialogue with the KBS where the KBS can ask the expert questions that require only the expert's problem solving expertise to answer, not his ability to understand problem solving, (a separate and difficult task [44]). Out of this should come a KBS that can reproduce the expert's behaviour and can in turn answer the expert's domain based questions, and for added benefit, answer our knowledge structuring questions.

The task approaches aim to enable the developer to work at the knowledge level rather than the symbol level; however, intensive analysis is still required. The aim here is to be able to handle knowledge without analysis. Tools based on Personal Construct Psychology such as KSS0 go some way towards this goal and are based on the related notion that human intelligence should be used for identifying differences rather than trying to create definitions[29]. These tools require the expert to make domain decisions about the differences and similarities between objects in the domain. However they also later require the expert to examine the knowledge and manipulate it more globally if it is unsatisfactory.

We do not have a fully developed plan of how this extravagant goal is to be achieved. What we have are some very limited instances of partially achieving this goal from which we are trying to extrapolate. The aim of this paper is to consider these instances to see how progress may be made towards the goal. Despite the critique of the task approaches above, the work here does not yet offer an alternative of the same scope and task methods will continue to be the basis of a major developments. The aim here is explore the possibility of methods more committed to the philosophical assumptions that underlie the task/modelling approaches.

5 Basic Ripple Down Rules

The limited system from which we hope to extrapolate is Ripple Down Rules. This approach was developed with the aim of using the knowledge an expert provided only in the context within which it was provided. For rule based systems it was assumed that the context was the sequence of rules which had been evaluated to give a certain conclusion. If the expert disagreed with this conclusion and wished to change the knowledge base so that a different conclusion was reached, knowledge was added in the form of a new rule of whatever generality the expert required, but

this rule was only evaluated if the same rules were evaluated with the same outcomes as previously. This corresponds to an expert guessing what led the trainee to the wrong conclusion and giving a justification for an alternate conclusion aimed at overcoming the specific deficiency in the trainee's knowledge. Fig 1 illustrates the structure of a ripple down rule knowledge base and Table 1 summarises the approach.

Initial experiments were based on rebuilding GARVAN-ES1 an early medical expert system [19, 33] and demonstrated rapid rule addition with low error rates[20, 23]. An extension to the method was to restrict the expert's choice of rule conditions to those that would ensure a valid rule. Firstly the case that prompts the addition of a new rule is stored in association with the rule as a "cornerstone case". In adding a rule to correct the classification of a case the expert is allowed to choose any conjunction of conditions that are true for the case as long as at least one these conditions differentiates the case from the only cornerstone case that could possibly reach the new rule, i.e. the cornerstone case associated with the rule that gave the wrong classification. [21, 60]. Gaines has proposed more thorough validation by storing all cases that are correctly classified by a rule and only allowing the expert to choose conditions for the new rule which exclude all these cases[27].

5.1 Ripples in relation to other methods

A key feature of ripple down rules is thus validation at acquisition time. The expert is only allowed to chose from conditions that will ensure a valid rule. However to do this all that is needed is domain knowledge. If an expert thinks the interpretation of a case is incorrect, then there must be some feature of the case which distinguishes it from a case which is correctly interpreted by the pathway. Thus nothing other than domain expertise is required to build a knowledge base.

Alain Rappaport (personal communication) has suggested that a good way to consider RDR is a strong method, with conventional knowledge engineering as a weak method. With RDR, the user has no control over the structuring of the knowledge and new knowledge is added to the system in a highly controlled fashion. However the knowledge is guaranteed to correctly interpret the case for which it has been added and not misinterpret any case which has been used to instigate knowledge acquisition. In contrast, conventional methods allow the knowledge engineer to structure the knowledge in any way he or she chooses. The method is weak and provides few constraints on how the knowledge is organised.

Task methods are in between in that they aim is to make the structure explicit by requiring the knowledge engineer to analyse problem solving in terms of tasks. Analysis is obviously required whereas RDR simply require the expert to use domain knowledge to select from a list of conditions. On the other hand, Clancey aims to ensure that there is no structure or control knowledge hidden in the knowledge base [14]. With RDR there can be no control knowledge hidden in the way the rules are organised; the expert (knowledge engineer) is not allowed to make knowledge structuring decisions. However, we have yet to argue whether the RDR approach can handle tasks other than classification. If it can we hypothesise that performance of

the RDR knowledge based system should be able to be modelled in any of the frameworks used to model human problem solving.

Knowledge base is a binary tree.
Each node is a rule with any number of conjunctions (only).
Rules are only added never modified (the tree is never reorganised and nothing higher up in tree is ever changed).
No probabilistic reasoning is used.
Conditions used in rules are boolean expressions which can include standard operators e.g. (sex = female, pregnant = true, age >= 45, (A*B) = (A-B) etc.
Each rule is linked to the case which prompted the addition of the rule and this case is stored when the rule is added (cornerstone case).
A rule can contain any conditions which are true for the case for which it is being added but must include a condition which is true for this case but not true for the case associated with the last satisfied rule in the sequence leading to the current rule.
Each rule has a conclusion associated with it. An RDR system normally gives a single conclusion, the conclusion from the last rule in the sequence evaluated which is satisfied by the data.
The key feature of ripple down rules is assisting and constraining the expert to only add valid knowledge
A new rule is valid if it will correctly deal with the case for which it is added, and will not result in the system mishandling at least the other cases which have been used for knowledge acquisition. This means that knowledge acquisition will be incremental, rather than, cyclic or haphazard.
The expert (and knowledge engineer) can only add rules at the bottom of a path, they thus cannot embed any control knowledge in the rules effecting the way inferencing proceeds

Table 1: features of an RDR system used to provide single classifications

5.2 Ripples experience

RDR have now been used to build a large medical expert system, PEIRS used to interpret chemical pathology reports at St.Vincent's Hospital Sydney [18, 25, 54]. This system now has over 1700 rules and covers much of chemical pathology, making it a very large medical expert system. However the knowledge has been added entirely by experts without any knowledge engineering assistance, nor any knowledge engineering or programming skills. The system was put into routine use with about 200 rules, with all later rules added with the system in actual routine use. The 200 initial rules were added off-line while interfacing problems were sorted out, but by the same process.

The laboratory data is initially processed into very simple categories of high, low, normal. To allow greater flexibility over these simple functions, rules can also use the standard mathematical operators found in procedural languages plus relational operators to produce boolean expressions. Numerical data is used in such

expressions rather than the simple high, low etc in one quarter of the conditions in rules. The rationale behind this approach is that although it is extremely difficult to obtain appropriate expressions to deal with temporal data from experts in a global sense, experts are highly skilled at providing justifications in particular contexts in terms of such expressions. This obviates the need for probabilistic reasoning and obtaining probabilistic information from an expert. Any oversimplification in a rule will be corrected in a later rule, with the use of the expressions allowing whatever type of refinement the expert requires.

An important overall result from these studies is that way the knowledge is added in small independent chunks allows the knowledge acquisition to be carried out and the system built while it is already in routine use. Experts can easily manage the acquisition task as a minor extension to their normal duties of checking reports. The experts have also come to the independent conclusion that the knowledge base will never be complete, but will evolve along with the evolution of laboratory technology and clinical interpretation knowledge. The speeds of the two developments seem reasonably matched. Recently one of the key analytes in thyroid testing has been changed with the new analyte providing somewhat different diagnostic information. This has resulted in some knowledge no longer being used and new knowledge having to be added. This required some effort but was not a significant task. The success of this type of system strongly supports Rappaport's proposal that one does not need a "better or "best" solution as long as a good enough solution can be implemented on the right time scale, with appropriate resource requirements[57].

6 Ripple Problems

Ripple Down Rules in the form described above have the limitations that they are restricted to classification and that only a single classification can be assigned to a case. The important question of whether the general ripple strategy can go beyond these limitations will be considered later. The immediate problem for ripple down rules even within this limited type of application is that the same knowledge may end up being repeated in many places throughout the tree. This may demand a great increase in the knowledge acquisition task. Although this is a real problem it does not vitiate the method. Standard inductive algorithms produce similar size knowledge bases to manual RDR [43]. Gaines' Induct algorithm modified to produce RDR produces a knowledge base one third the size of the manual version and one half the size in terms of rule conditions [27, 31] These results suggest that manual RDR are not as compact as inductive RDR, but a two fold difference is still a very good result for a manual method. An important finding is that the inductively built RDR knowledge bases, built for the standard induction data sets, are about one third the size of those built by the original method [31] . This seems to be simply a result of objects in the world being classified by a conjunction of conditions and they fail to be that object if any of the conditions are false. Catlett also has argued that RDR are a good mediating representation for dealing with the results of induction [3].

Gaines suggested an appropriate strategy overall, was to first build an RDR KB manually (assuming well classified data is not available for induction). As this develops it may eventually need compressing but by then it could be used to provide

consistent classifications for archived cases which could then be processed inductively to produce a more compact version of the KB.

7 Formalising Ripples

Catlett above described ripple down rules as ordered rules with exceptions. The exceptions to any rule is again a set of ordered rules with exceptions. This description probably more truly captures the nature of this representation than describing it as a tree particularly since the tree is so unbalanced, far more new rules are added than corrections.[17].

Gaines has developed a KL-ONE like graphical representation language for knowledge bases [27, 30]. In this language links are not labelled, only nodes, so that one cannot specify that a rule is attached to the false or true branch of its parent. To maintain his unlabelled links, Gaines adds an extra type of rule node, a rule that starts a new context, i.e. a rule that is entered only from a satisfied rule. This captures the essence of RDR but allows a rule to be reached from more than one rule. This may or may not be useful.

Another approach (A.Srinivasan, Oxford University, personal communication) is to describe ripple down rules in terms of prioritised or labelled logics. The difference between these types of logic and other logics is that a predicate may have a label attached to it, indicating its priority over other predicates. Rather than recursively evaluating every predicate that may be considered, the priority labels may be used to determine that inferencing will proceed via a single predicate. This remains to be evaluated but may provide a basis for going beyond propositional reasoning with an RDR approach. It also opens up the use of a least general generalisation approach from inductive logic programming [47] to form rules.

8 Extending Ripple Down Rules

8.1 Multiple classifications

The Problem. A significant problem with ripple down rule trees is that only a single classification is produced. Ripple down rules share this limitation with many induction methods, e.g. ID3 and Induct. In many domains more than one classification may be required, e.g. a patient with multiple diseases. Knowledge bases produced either by manual ripple down rules or induction, in theory can be organised to produce multiple classifications either by having multiple knowledge bases or by designating a combination classification as a new single classification. The problem with the combination classifications is that of repetition. For example if classification **A** is independent of **B**, **C** and **D** then classifications **A**, **B**, **C**, **D**, **AB**, **AC**, **AD** are possible. The combination classification approach requires that all the rules to produce **A** have to be reproduced with each of **B**, **C** and **D**. This is the strategy currently used in PEIRS and the experts are starting to notice the redundancy caused by this problem although it has been minimised to date by the choice of chemical pathology sub-domains for which the system has been developed.

The obvious solution for both the inductive methods and manual ripple down rules is to have a series of knowledge bases for the different classifications. The first problem is that this requires a structuring knowledge engineering decision about what goes into the different knowledge bases and so is contrary to the ripple down rule philosophy. Each knowledge base must deal with a group of mutually exclusive classifications which are each independent from (and could co-occur with) the classification produced by the other knowledge bases. This is a very difficult knowledge engineering decision and in medicine at least is often impossible. The aim then is a single RDR knowledge base which can be used to make multiple classifications for a case.

Knowledge Base Structure. The required knowledge base structure is very simple and is an n-ary tree rather than a binary tree as previously used to describe RDR (Fig 1). With binary tree RDR when a rule is added the conditions to choose from come from the difference between the case and the case associated with the last true rule. The expert knows nothing about the rules that have failed to fire and their cases. The expert does not explicitly provide information that the new rule should not apply to cases that satisfy one of the failed rules. Since the false branch is irrelevant the tree is better seen as an n-ary tree and to achieve multiple classifications for a case all children of each satisfied rule are evaluated down through the tree. The classification from the last satisfied rule on each pathway is given. The structure of the knowledge base may be able to be used to comment on the quality of and relationship between the conclusions. If classification **A** is reached on one path but then on another path the case satisfies another rule giving a classification **A** followed by a refinement rule also satisfied that gives **B**, then the appropriate conclusion is **B** but probably not **A**. This approach may be used to reduce the brittleness of the system and warn when conclusions may not be appropriate [36, 37].

Such a system could of course be built as a decision table. The reason for not doing this is that the tree structure easily identifies rules as corrections. As above this can be exploited to assess the quality of the expert system's classifications and to partially identify what classifications may occur together. Algorithms have been proposed for controlling knowledge acquisition with case differences using a decision table, but no experiments have been conducted [22, 34]

Knowledge Acquisition: Where to put the rule. There are a number of ways in which a case may be wrongly classified:

- One or more of classifications may have to be deleted
- One or more classifications may be added
- Some combination of the above.

If a classification is to be deleted, then a rule producing a null classification is added after the rule. If an extra classification is to be added, the rule goes at the highest level. The difficult problem is if a classification is to be deleted and a classification added. Is the new rule a correction of the wrong rule to be attached at the bottom of the pathway or should it be available more generally and attached to the highest level, or somewhere in between with a null classification rule attached to the bottom of the path? The problem here is a clear example what Rappaport sees as the central issue in knowledge engineering: how does one achieve a balance between treating knowledge as purely situated or global [57]?

We are currently investigating an approach where the expert is presented with conditions to select for the rule which explicitly list the conditions satisfied in the rule path as well as the differences. The conditions the expert chooses from the pathway will not be used in the rule, rather they will determine where the new rule should be attached. This is a complex situation as there may be a number of pathways where the rule may be added. However, if the expert does not have a high level of expertise then the locating conditions selected could be too specific or too general resulting in either more rules to cover the same classification or more correction rules if the first rule was too high up the tree. Such choices effect whether the system will evolve rapidly with many cases being classified and many errors having to be corrected or evolve more slowly with many cases unclassified as the new rules are taken more strictly as corrections.

At present we are using the expert's choice of necessary rule conditions to determine where the rule should go. We speculate that it should be possible to develop an approach where the expert can specify whether he or she would like the system's development to provide rapid coverage with lots of errors, or slow coverage with fewer errors, or anywhere in between. As the KBS developed it may be possible to use learning techniques to decide from existing examples of rules and classifications in the knowledge base what the locating conditions should be and where the rule should go. The learning should be able to select the necessary conditions in a more or less conservative way as required by the expert and include the expert in identifying conditions if required.

Knowledge Acquisition: Case Differences. A key issue in knowledge acquisition for a multiple classification RDR system is how to organise the case differences. One of the advantages of single classification RDR is that only one case may be misinterpreted by the new rule and so the case difference list from which the expert selects one or more conditions to ensure a valid rule can be constructed easily. With the multiple classification environment other cases associated with other rules may now be able to reach the new rule added, causing them to have extra classifications.

With the RDR approach we do not want to the expert to make up a rule which is then checked, rather we wish to present the expert with a list of conditions to choose from which will ensure a valid rule. The difference between the intersection of the cornerstone cases which can reach the rule and the new case cannot be used. There may be a number of different reasons which exclude these cornerstone case so that

intersection may be empty or may not include the conditions the expert wishes to use. A very simple algorithm is to present the expert first with differences between the new case and one of the cornerstone cases to make a new rule. If this rule does not exclude all the cornerstone cases, differences between another of the cases not excluded and the new case are again presented to the expert and a further rule is added and so on till all cases are excluded.

In theory one should exclude all cornerstone cases. In practice however the knowledge base evolves so that a case that is initially given a single classification may later need other classifications as well. The algorithm we use is that whenever a case requires the addition of a rule and becomes a cornerstone case, it is also counted as a cornerstone case for any other rule which it correctly fires to give other classifications. This means that we gradually build up a number of cornerstone cases for each rule for which the case difference algorithm above must be used. We do not however apply the algorithm to all cornerstone cases; we wait for errors to occur.

Experience with Multiple Classifications. The GARVAN-ES1 expert system is currently being re-implemented as a multiple classification expert system using essentially the algorithm above. The knowledge base is about two-thirds complete and it appears that it will be of a similar size to the single classification version, but it is uncertain whether it will increase in size more rapidly or more slowly as it nears completion. Most of the acquisition has been done by the expert who has added the PIERS rules. He is happy with the somewhat different knowledge acquisition task, but one still based on selection of conditions which will produce a valid rule, except here the validity is weaker. Further evaluation of the rather speculative algorithms above using the GARVAN and PEIRS domains is proposed.

Multiple Classification Control Knowledge. From Clancey's perspective making the control knowledge explicit makes clear the type of model that is being constructed [14]. In contrast with multiple classification RDRs the control decision of where to put a rule primarily effects how the system develops. For practical real world systems this is a critical issue. Is it preferable for the system to make a lot of errors early on which will gradually diminish or have a low error rate requiring occasional addition of rule continuing for the life of the system.

8.2 Configuration Tasks

Clancey's distinction between classification and configuration[10] is still fundamental although Clancey would probably now make the distinction between classification and simulation [14].

A significant configuration task has been attempted with RDR [48], but in the initial study the "no knowledge engineering" principle of ripple down rules has been abandoned. The case is discussed here to make clear the problems that must be addressed if configuration tasks are to be handled in a similar way to classification tasks with no knowledge engineering decisions.

The configuration task attempted is to configure an ion chromatography system. There are eight components which may be varied giving many different combinations of equipment and reagents to try and optimise the system for different types of samples to be tested. In this particular case there was a data base of over 4000 published applications of ion chromatography. Induct [27] was used to develop eight separate RDR classification trees to decide on each of the eight components. The number of rules per KB varied from 67 to 640.

In the system developed the available data for a case were applied to each of the eight trees; any new data were added to working memory and the data run on the eight KBS again. With conventional ripple down rules, for multiple or single classification, an unknown condition results in the rule failing, unless the rule explicitly uses "unknown" as a condition. In the system developed, it was assumed that as long as a rule did not fail because a condition was false rather than unknown, then it was assumed that the rule may be true. This may result in a number of pathways through the tree, with possibly different choices for the component. A conservative approach was then followed by only adding to working memory conclusions from trees where a single conclusion had been reached, perhaps a number of times. This process is then repeated till no further changes in working memory occur. The rules are then run with a conventional RDR inference engine where unknown is false and only a single conclusion is reached for each tree. As a final check this data is now run again to see if anything changes, if it does the report indicates that the configuration has to be finalised by hand. This simple method performs satisfactorily on test data.

8.3 Merging RDR Classification and Configuration

The issue we wish to explore is whether the knowledge engineering decisions above, setting up a structure for configuration, could be hidden so that as a knowledge base developed it could gradually emerge that it was being used for configuration rather than classification, and the particular type of configuration further identified. These would be explanations of what the KBS was doing rather than control decisions to get the system to carry out the appropriate task. It should be noted that we do not have solutions to these problems, at best they are suggestions of where solutions may lie.

Data modelling. One key issue is how the data should be handled. Some data modelling is always necessary. In the KADS approach considerable effort is put into ensuring a good representation. For RDR a simple attribute value approach is sufficient and the data types matter little. It does not matter much for example if boolean variables such as TSH_HIGH, TSH_LOW and TSH_NORMAL are used rather than a variable TSH with three ordinal values. The choice of representation affects induction as knowledge is implicit in the representation which otherwise has to be gained from the examples but for RDR the expert choice of rule conditions depends on the expert and the meaning he or she attaches to names.

The problem becomes more difficult with configuration systems. With single classification RDR the classifications are, for all purposes, different values of a single

classification variable as the output of the system is restricted to a single classification at a time. With a multiple classification system, some classifications are independent and may occur together or are mutually exclusive and never occur together. The mutually exclusive nature of such classifications is not controlled by the representation but is determined by the knowledge added.

For a configuration system it is essential to know which classifications are mutually exclusive. Firstly the classifications produced are fed back into the system as data. Secondly in the approach outlined above rules do not fail because of an unknown condition but only because a condition is explicitly false. That is, it must be known whether a conclusion that has been reached is mutually exclusive with a condition in a rule being evaluated. It must be known whether the two classifications can be treated essentially as two mutually exclusive values of the one attribute.

In the RDR approach we do not wish to carry out knowledge acquisition to determine representations. We suggest that the developing knowledge base contains information about which conclusions can occur together and which are mutually exclusive. We speculate that an algorithm could be applied to the knowledge base to identify groupings of mutually exclusive and independent classifications. We imagine the development of such an algorithm is fairly straightforward, if necessary based on approaches to discovering new concepts in inductive logic programming [47]. These groupings would of course be open to ongoing revision as more knowledge is added.

We thus have an approach whereby the classifications that emerge as the system evolves can be organised as independent attributes with mutually exclusive values as required in configuration problems. Of course the system's use of data could evolve in the same way as in the TSH example above. This starts to blur the distinction between what is data and what are classifications and suggests that a system could start out based on simple literals and a more complex implicit data model be identified later as required.

It should be noted that this attempt to build a data model from information supplied in context seems to have much in common with Boy's approach to indexing documents [2].

In embedded applications such concerns are of less importance as the data model is largely determined by the rest of the system, and there should be only one data model for the whole system [34, 35]. Secondly there is no reason to rely solely on learning techniques to identify attributes and values. The expert should be free to identify attributes and values as long as these can be refined (see below).

Inferencing. There is no reason to have multiple knowledge bases for configuration as in the example above. The multiple classification approach to inferencing is equivalent to having separate knowledge bases, but also allows a more complex structure equivalent to what could be called nested separate sub-knowledge bases, which may allow more possibilities. The only extra requirement for configuration is the treatment of unknown conditions. For configuration, if a

condition in a rule is unknown, but yet could be concluded by the knowledge base somewhere, then it should be assumed that the conclusion of the rule is possible and added to the working memory for a further inference cycle. One must be able to decide if a condition in a rule is unknown, or is false because the attribute referred to, even if unnamed, has another value. We speculate that it does not really matter if the organisation of classifications into attributes and values proposed above either too readily identifies classifications as mutually exclusive or fails to pick these up early on. We speculate that what this will effect is the way the system's knowledge evolves.

At this stage we have not fully explored the issue of the number of inference cycles and stopping criteria if cyclic changes in attributes and values occur, nor alternate constraint based or truth maintenance methods. The critical issue with the approach we have outlined is that it will work for classification as well as configuration. In the final pass through the rules unknown is assumed to be false, the same approach as used for classification. The rules will not include conditions which can be concluded by other rules. Note that in this approach there is no requirement for intermediate conclusions.

Further Data Modelling and Expert Interface. It is essential in this approach that the expert find and reuse conclusions. If each conclusion is a new text string, it will be impossible to organise conclusions into attribute value groupings, an essential requirement for the configuration approach above. PIERS has almost half as many classifications as rules despite a simple text string find facility to enable the expert to find and reuse conclusions. It is important to develop some way of assisting and encouraging the expert to find the right classification. The relations that we surmise can be discovered (above) may be used to organise the presentation of classifications to the expert. Other techniques for relating rules may be relevant [15]. Secondly it seems appropriate to ask the expert about relationships between classifications in terms of new attributes, perhaps using a personal construct approach [29]. These may be used for nothing more than to help the expert identify a classification. Asking for further information of this nature is not contrary to the ripple down rule principle for no analysis is required, only acquisition. No decisions are being made to structure the knowledge base for inference, rather, this knowledge is used to structure the choices presented to the expert. Again the adequacy of the information provided here by the expert is not intrinsic to the success of the system rather it determines how the system will evolve. In contrast to other approaches each maintenance incident takes the same effort regardless of the size of the knowledge base. However, expert choices like the will effect the pattern of maintenance incidents over the life of the system.

A related issue is the identification of features in the data. The expert may wish to refer to data (or classifications) from a higher level viewpoint. The most obvious application of this is in dealing with images, but it is a widespread requirement. The conventional approach is that this is part of the data modelling task, however the identification of features is never satisfactory (e.g. note above the frequent use of numeric data in PIERS to implicitly refine identification of data as high, low, etc). The RDR approach allows one to refine pre-identified features in specific contexts

and this works well dealing with temporal data in PIERS. However, the identifier for the feature still refers to the old definition. The problem with any changed definition of a feature is that it cannot be changed globally because there may be earlier rules which worked well with an earlier definition of the feature and which will introduce errors with the new feature definition. Menzies has proposed time stamping the changes in feature definition so that they will only be use with new rules [45]. This has not yet been evaluated and may need some refinement to avoid oscillating feature definitions generated from different contexts. The purpose of introducing features is not to organise the knowledge structure using intermediate conclusions but to manage the way the choice of conditions is presented to the expert so that he or she is not overwhelmed with choices.

Explanation. From Clancey's perspective a knowledge base should be able to provide a variety of explanations[14]. We suggest that the extra information required about the relationships between various classifications and data (if these should be any longer distinguished) that we hypothesise can be gained either by induction or from the expert could also be used to provide a variety of explanations. For example in PEIRS and GARVAN-ESI there is no relationship between the classifications of thyrotoxicosis (hyperthyroidism) and hypothyroidism. In fact they are mutually exclusive and ordinal as they represent under and overactivity of the thyroid gland. There are also a number of more subtle variants. If these relationships are discovered or acquired, even in very simple form, it seems to us that the "heuristics" then have the potential to be converted into a so called "deeper" model. The relationship between the ordinal relationships between TSH levels (high and low) and hyper and hypothyroidism as found in the rules. could be expressed as monotonic decreasing in QSIM terms [38], capturing the implication of negative feedback. This proposal links with other work on translating between models [39], linking models and heuristics in causal explanations [46] testing models against data [26] and discovering models [42].

The details of what we are suggesting are highly speculative; the important point we are claiming is that it seems possible to build knowledge bases by the incremental addition of situated knowledge for both classification and configuration tasks without having to carry out analysis and distinguish between these tasks. Extra knowledge can be added along the way to improve access to the system. This is in many ways the same notion as explanation since it is concerned with providing viewpoints to encapsulate what the system is doing. The success with which this knowledge is acquired or induced will affect the explanations of the system and access to the system, but in contrast to conventional systems this knowledge is not used by the system to structure the way in which it reaches conclusions.

9 Summary

Knowledge engineering has moved from a knowledge extraction perspective to a modelling perspective. However, the modelling perspective, finds its roots in a philosophical viewpoint that the knowledge that an expert expresses is constructed to justify the expert's insights and judgements. These justifications have nothing to do

with how the expert carries out problem solving. This then suggests that it is not necessary to carry out problem solving in terms of any particular task analysis. It is hypothesised that this type of difficult analysis is not necessary. Rather it is possible to incrementally acquire small pieces of situated domain knowledge which will evolve into the type of problem solver required. The goal is that the system itself should incorporate these fragments into the knowledge based system, rather than relying on the expert and knowledge engineer to make decision about how to structure the knowledge.

This approach could be called apprenticeship learning, because the apprentice gradually learns to overcome their mistakes and develop a more expert performance. As well, the apprentice gradually learns how to explain their conclusions in various frameworks. The apprentice does this by reflecting on their own knowledge as well as acquiring the bits and pieces of knowledge necessary to create explanations in these frameworks. The approach we have proposed, if feasible, would seem likely to produce such apprentice systems. The teacher doesn't know how the apprentice organises their knowledge. The teacher just keeps on pointing out errors and providing guidance and eventually the apprentice becomes an expert. Like a human the system will have the potential to explain itself in whatever framework is required, but as with humans this will be a difficult skill to develop and as with humans a crucial issue will be in trying to find the right buttons to press to get the right answers

Apprentices also make better and better guesses as their skills develop. We surmise that in the approach we have outlined one could use either an expert or an appropriate learning algorithm to decide which of the allowed conditions should be used in a rule. We have emphasised the expert, but as with Gaines development[27], there seems no reason that induction could not be used to identify the conditions to be selected if sufficient data was available.

Finally, in keeping with Clancey's proposal the expert or knowledge engineer cannot make or embed control decisions in domain knowledge[14]. However, there are further levels of domain knowledge implicit in the structure of the knowledge base as it evolves that we hypothesise can be identified and used. The control decisions that an expert or knowledge engineer can make with this approach determine the evolution of the system; they can decide what sort of evolution is suitable for the domain.

The modelling approach to knowledge acquisition is a major advance over the knowledge extraction approach. Our highly speculative hypothesis is that it is not necessary for apprentices to be told how to organise their knowledge. If their knowledge is constantly corrected and added to they will organise it themselves to carry out the required problem solving. The essence of our hypothesis is that it may be possible to organise a knowledge base system like a human apprentice so that the difficulty of adding each new piece of knowledge is independent of the amount of knowledge already accumulated. Acquiring knowledge becomes a very simple incremental activity.

Acknowledgements

This work is supported by Australian Research Council grants (AC9031495 and AC9030091) and a Digital Equipment Corporation research contract (1106). The insights provided by Glenn Edwards and Ashwin Srinivasan and other colleagues in the UNSW AI lab and the Department of Chemical Pathology St. Vincent's Hospital Sydney are gratefully acknowledged.

References

1. D. Allemang: Modelling a configuration problem with Generic Tasks. In: M. Linster (eds.): *Sisyphus'91: Models of Problem Solving*. GMD (Gesellschaft für Mathematik und Datenverarbeitung mbH) 1992.
2. G. Boy: Some theoretical issues on the computer integrated documentation project. In: (eds.): *6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop*. Banff, 1991, pp. 3.1-3.15
3. J. Catlett: Ripple-Down-Rules as a Mediating Representation in Interactive Induction. In: R. Mizoguchi, H. Motoda, J. Boose, B. Gaines and R. Quinlan (eds.): *Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop*. Kobe, Japan, 1992, pp. 155-170
4. B. Chandrasekaran: Towards a taxonomy of problem solving types. *AI Magazine Winter/Spring*, 9-17 (1983)
5. B. Chandrasekaran: Generic tasks in knowledge-based reasoning: high level building blocks for expert system design. *IEEE Expert* 1 (3), 23-30 (1986)
6. W. Clancey: The frame of reference problem in cognitive modelling. In: K. VanLehn and A. Newell (eds.): *Proceedings of the annual conference of the cognitive science society*. Hillsdale, New Jersey: Lawrence Erlbaum 1989, pp. 107-114
7. W. Clancey: The knowledge level revisited: modelling how systems interact. *Machine Learning* 4, 285-291 (1989)
8. W. Clancey: Viewing knowledge bases as qualitative models. *IEEE Expert Summer*, 9-23 (1989)
9. W. J. Clancey: The epistemology of a rule bases system - framework for explanation. *Artificial Intelligence* 20, 215-251 (1983)
10. W. J. Clancey: Heuristic classification. *Artificial Intelligence* 27, 289-350 (1985)
11. W. J. Clancey: Book review of "The Invention of memory". *Artificial Intelligence* 50 (2), 241-284 (1991)
12. W. J. Clancey: A boy scout, Toto, and a bird: how situated cognition is different from situated robotics. In: (eds.): *NATO workshop on emergence, subsumption and symbol grounding*. 1991.
13. W. J. Clancey: *Situated Cognition: stepping out of the representational flatland*. *AI Communications*, (1991)

14. W. J. Clancey: Model construction operators. *Artificial Intelligence* 53 (1-115), (1992)
15. R. M. Colomb: Computational Stability of Expert Systems. In: J. Liebowitz (eds.): *Proceedings of the World Congress on Expert Systems*. Orlando, Florida, Pergamon Press 1991, pp. 1123-1131
16. P. Compton: Insight and Knowledge. In: J. Boose, W. Clancey, B. Gaines and A. Rappaport (eds.): *AAAI Spring Symposium: Cognitive aspects of knowledge acquisition*. Stanford University, 1992, pp. 57-63
17. P. Compton, G. Edwards, B. Kang, L. Lazarus, R. Malor, T. Menzies, P. Preston, A. Srinivasan and C. Sammut: Ripple down rules: possibilities and limitations. In: J. Boose and B. Gaines (eds.): *6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop*. Banff, 1991, pp. 6.1-6.18
18. P. Compton, G. Edwards, A. Srinivasan, R. Malor, P. Preston, B. Kang and L. Lazarus: Ripple down rules: turning knowledge acquisition into knowledge maintenance. *Artificial Intelligence in Medicine* 4 , 47-59 (1992)
19. P. Compton, R. Horn, R. Quinlan and L. Lazarus: Maintaining an expert system. In: J. R. Quinlan (eds.): *Applications of Expert Systems*. London: Addison Wesley 1989, pp. 366-385
20. P. Compton and R. Jansen: Knowledge in context: A strategy for expert system maintenance. In: C. J. Barter and M. J. Brooks (eds.): *AI'88 (Proceedings of the 1988 Australian Artificial Intelligence Conference)*. Berlin: Springer-Verlag 1989, pp. 292-306 (283-297 original Proceedings)
21. P. Compton and P. Preston: A minimal context based knowledge acquisition system. In: (eds.): *"Knowledge Acquisition: Practical Tools and Techniques" AAAI-90 Workshop*. Boston, 1990,
22. P. Compton, W. Yang, M. Lee and B. Jansen: Cornerstone cases in a dictionary approach to knowledge based systems. In: B. Jansen, B. Gaines, J. Carlis and J. Kontio (eds.): *IJCAI-91 workshop on software engineering for knowledge-based systems*. Sydney, 1991, pp. 24-40
23. P. J. Compton and R. Jansen: A philosophical basis for knowledge acquisition. *Knowledge Acquisition* 2 , 241-257 (1990)
24. H. Dreyfus and S. Dreyfus: Making a mind versus modelling the brain: artificial intelligence back at a branchpoin. *Daedalus* 117 (Winter), 15-43 (1988)
25. G. Edwards, P. Compton, R. Malor, A. Srinivasan and L. Lazarus: PEIRS: a pathologist maintained expert system for the interpretation of chemical pathology reports. *Pathology* 25 , 27-34 (1993)
26. B. Z. Feldman, P. J. Compton and G. A. Smythe: Hypothesis testing: an appropriate task for knowledge based systems. *Proc 4th Knowledge Acquisition for Knowledge Based Systems Workshop* , 10.1-10.20 (1989)
27. B. Gaines: Induction and visualisation of rules with exceptions. In: J. Boose and B. Gaines (eds.): *6th AAAI Knowledge Acquisition for Knowledge Based Systems Workshop*. Banff, 1991, pp. 7.1-7.17

28. B. Gaines: The Sisyphus problem-solving example through a visual language with KL-ONE-like knowledge representation. In: M. Linster (eds.): Sisyphus'91: Models of Problem Solving. GMD (Gesellschaft für Mathematik und Datenverarbeitung mbH) 1992,
29. B. Gaines and M. Shaw: Cognitive and Logical Foundations of Knowledge Acquisition. In: (eds.): 5th AAAI Knowledge Acquisition for Knowledge Based Systems Workshop. Bannf, 1990, pp. 9.1-9.25
30. B. R. Gaines: An interactive visual language for term subsumption languages. In: (eds.): Proceedings of the 13th International Joint Conference on Artificial Intelligence. Sydney, Morgan Kaufmann 1991, pp. 817-823
31. B. R. Gaines and P. J. Compton: Induction of Ripple Down Rules. In: A. Adams and L. Sterling (eds.): AI '92. Proceedings of the 5th Australian Joint Conference on Artificial Intelligence. Hobart, Tasmania, World Scientific, Singapore 1992, pp. 349-354
32. T. R. Gruber: A translation approach to portable ontology specifications. In: R. Mizoguchi, H. Motoda, J. Boose, B. Gaines and R. Quinlan (eds.): Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop. Kobe, Japan, 1992, pp. 89-107
33. K. Horn, P. J. Compton, L. Lazarus and J. R. Quinlan: An expert system for the interpretation of thyroid assays in a clinical laboratory. *Aust Comput J* 17 , 7-11 (1985)
34. B. Jansen and P. Compton: The Knowledge Dictionary. London: Academic Press *in press*
35. R. Jansen and P. Compton: The knowledge dictionary: storing different knowledge representations. *Proc 5th Aust Conference on Applications of Expert Systems* , 143-162 (1989)
36. B. Kang and P. Compton : Knowledge acquisition in context: the multiple classification problem. In: (eds.): Proceedings of the Pacific Rim International Conference on Artificial Intelligence. Seoul, 1992, pp. 847-854
37. B. Kang and P. Compton: Towards a process memory. In: J. Boose, W. Clancey, B. Gaines and A. Rappaport (eds.): AAAI Spring Symposium: Cognitive aspects of knowledge acquisition. Stanford University, 1992, pp. 139-146
38. B. Kuipers: Qualitative simulation. *Artificial Intelligence* 29 , 289-338 (1986)
39. M. Lee, P. Compton and B. Jansen: Modelling with Context-Dependant Causality. In: R. Mizoguchi, H. Motoda, J. Boose, B. Gaines and R. Quinlan (eds.): Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop. Kobe, Japan, 1992, pp. 357-370
40. D. Lenat and E. Feigenbaum: On the thresholds of knowledge. *Artificial Intelligence* 47 (1-3), 185-250 (1991)
41. B. Lonergan: *Insight*. London: Darton, Longman and Todd 1959

42. A. Mahidadia, P. Compton, C. Sammut, T. Menzies and G. Smythe: Inventing Causal Qualitative Models: A Tool for Experimental Research. In: A. Adams and L. Stirling (eds.):AI'92, Proceedings of the 5th Australian joint conference on artificial intelligence. Hobart, World Scientific, Singapore 1992, pp. 317-322
43. Y. Mansuri, P. Compton and C. Sammut: A comparison of a manual knowledge acquisition method and an inductive learning method. In: J. Boose, J. Debenham, B. Gaines and J. Quinlan (eds.):Australian workshop on knowledge acquisition for knowledge based systems. Pokolbin, 1991, pp. 114-132
44. D. Marques, G. Klinker, G. Dallemagne, P. Gautier, J. McDermott and D. Tung: More data on useable and reusable programming constructs. In: J. Boose and B. Gaines (eds.):Proceedings of the 6th Bannf Knowledge Acquisition for Knowledge-Based Systems Workshop. Bannf, 1991, pp. 14.1-14.19
45. T. Menzies: Concerning the use of procedural constructs as a knowledge acquisition technique. In: J. Boose, J. Debenham, B. Gaines and J. Quinlan (eds.):Australian workshop on knowledge acquisition for knowledge based systems. Pokolbin, 1991, pp. 133-156
46. T. Menzies, A. Mahidadia and P. Compton: Using Causality as a Generic Knowledge Ontology: Experiences with CHuTE. In: J. Boose, B. Gaines and M. Musen (eds.):7th AAAI-sponsored Workshop on Knowledge Acquisition for Knowledge Based Systems Workshop,. Bannf, 1992, pp. 18.1-18.19
47. S. Muggleton: Inductive Logic Programming. London, Academic Press. (1992)
48. M. Mulholland, P. Preston, B. Hibbert and P. Compton: An expert system for ion chromatography developed using machine learning and knowledge in context. In: (eds.):Proceedings of the Sixth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. Edinburgh, 1993, *in press*
49. M. Musen: Automated Generation of Model-Based Knowledge Acquisition Tools. San Mateo: Morgan Kaufmann 1989
50. A. Newell: The knowledge level. Artificial Intelligence 18 , 87-127 (1982)
51. A. Newell and H. Simon: Computer Science as empirical enquiry: symbols and search. In: J. Haugeland (eds.): Mind Design. Cambridge, MA: MIT Press 1981
52. J. Piaget: The child's conception of the world. London: Routledge and Kegan Paul 1929
53. K. Popper: Conjectures and refutations. London: Routledge and Kegan Paul 1963
54. O. Preston, G. Edwards and P. Compton: A 1600 rule expert system without knowledge engineers. In: J. Leibowitz (eds.):Proceedings of the Second World Congress on Expert Systems. Lisbon, Pergamon 1993, *in press*
55. A. Puerta, J. Egar, S. Tu and M. Musen: A multiple method knowledge acquisition shell for the automatic generation of knowledge acquisition tools. Knowledge Acquisition 4 , 171-197 (1992)

56. A. Rappaport: A theory of local and interstitial knowledge. In: (eds.):6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop. Banff, 1991, pp. 21.1-21-18
57. A. T. Rappaport: On cognitive invariants and the knowledge milieu. In: R. Mizoguchi, H. Motoda, J. Boose, B. Gaines and R. Quinlan (eds.):Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop. Kobe, Japan, 1992, pp. 435-446
58. J. Sandberg and B. Wielinga: How situated is cognition. In: (eds.):12th International Joint Conference on Artificial Intelligence. Sydney, Morgan Kauffman 1991, pp. 341-346
59. A. T. Schreiber: Sysphus'91: Modelling the Office-Assignment Domain. In: M. Linster (eds.):Sysphus'91: Models of Problem Solving. GMD (Gesellschaft für Mathematik und Datenverarbeitung mbH) 1992, pp.
60. A. Srinivasan, P. Compton, R. Malor, G. Edwards and L. Lazarus: Knowledge Acquisition in Context for a Complex Domain. In: B. Weilinga, J. Boose and B. Gaines (eds.): Proceedings of the Fifth European Knowledge Knowledge Aquisition Workshop. Pergamon 1992, in press
61. L. Steels: Components of Expertise. AI Magazine 11 (2), 29-49 (1990)
62. L. Steels: End-user configuration of applications. In: R. Mizoguchi, H. Motoda, J. Boose, B. Gaines and R. Quinlan (eds.):Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop. Kobe, Japan, 1992, pp. 43-63
63. B. J. Wielinga, A. T. Schreiber and J. A. Breuker: KADS: a modelling approach to knowledge engineering. Knowledge Acquisition 4 (1 (special issue on KADS)), 5-54 (1992)
64. T. Winograd and F. Flores: Understanding computers and cognition. Reading, MA: Addison Wesley 1987
65. L. Wittgenstein: Philosophical Investigations. London: Blackwell 1953