AI Models as a Variety of Psychological Explanation

Kieron O'Hara and Nigel Shadbolt
Artificial Intelligence Group
Dept. of Psychology
University Park
Nottingham NG7 2RD
U.K.

Abstract

The paper attempts to establish a basis upon which it could plausibly be said that knowledge level models typically used in the development of AT systems such as expert systems could have psychological import. Various modelling methodologies are set out, and it is shown that these methodologies cannot supply psychological explanations of expertise on the basis of ordinary realist assumptions about the mind, since the knowledge level primitives cannot supply the right sort of links between tasks and AI methods. In contrast, an anti-realist, interpretative view of mind is set out, and it is shown how AI modelling methodologies could, in that context, be of psychological value. Finally, a short example gives some concrete expression to these ideas.

1 Introduction

A significant area of research in AI is the field of model-based approaches to knowledge engineering, the field of developing knowledge-based systems which, in some sense, *depict* the target domain of the system. Influential examples of this sort of approach are the KADS methodology [Wielinga *et* al., 1992], the generic task methodology [Chandrasekaran, 1990], the problem-solving method approach [Puerta *et* al., 1992], and the Components of Expertise approach [Steels, 1992]. These approaches have a number of core assumptions in common. For example:

1. Knowledge acquisition is *not* a process of transferring knowledge from the expert to the machine. The task to be performed should, instead, be *modelled* at the knowledge level.
2. Where possible, *generic* modelling structures should be available, to facilitate reuse of (parts of) models.

The question which we wish to address in this paper is: What is the relationship between the modelling apparatus provided by approaches such as these, and a vocabulary suitable for psychological explanation of the expertise? Can it be the case that models developed under the rubric of one of these approaches can play the *dual* role of the chief determinant of an implementation combined with a psychologically plausible description of expertise?

The conceptual separation of the model and the implementation means that there is, in all these approaches, a distinction to be made between the description of the task to be solved and the methods used to carry out that task. What we wish to do in this paper is to examine the status of the knowledge level entities that are postulated by the various approaches, in contrast to the lower, implementational apparatus that they provide, and try to sketch the requirements which have to hold if these modelling views are correctly to be seen as descriptions of the experts' problem-solving.

If we define a task roughly as a classification of a problem, the difficulty arises that a rich task vocabulary has grown up over the years, generally in non-knowledge engineering contexts. These classifications are generally vague or underspecified, which leads to a trade-off between ease of classification of a problem, and the classification of a problem's being of some help with the implementation of a system to solve that problem. Merely classifying a problem as an instance of a certain task does not, all things being equal, tell you how actually to solve that problem.

We might- say at the outset that we are taking rather a non-standard line. Standard claims for the psychological importance of the knowledge level entities postulated by KBS development methodologies generally argue in terms of 'cognitive emulation' a straight 11 mapping between expert system features and findings in cognitive research (e.g. [Slatter, 1987, p.58]). In this paper, we will not attempt to defend such a close association between the two fields; we shall explore some conditions that make .some sort of association possible (and plausible). Our claim is that, in knowledge level modelling methodology development, it need not be the case that psychological explanatory power is lost when 'psychological' considerations are suppressed in favour of knowledge engineering considerations.

2 AI Methods and Their Constraints

The problem-solving method community tends to address this problem by producing a *theory* of tasks, the purpose of which is to constrain the number of methods available for the performance of the tasks under scrutiny. Hence, conceptualizing a problem as open to solution by performing such and such a task will radically reduce the search space among possible methods.

So, for example, the early work of Chandrasekaran on generic tasks [Chandrasekaran, 1983] associated control structures and forms of knowledge with each task (defined with an input/output specification) — perhaps the strongest connection between the task and the method. This approach had a number of nice properties. The generic task methodology as originally set out by Chandrasekaran had the effect of giving you the knowledge representation and control for free. The knowledge and its use (both in terms of its computational manipulation, and its wider function — the input/output relation that in part determined the content of the generic task) were both bundled together into the generic task. Once each generic task was allied to a special purpose knowledge representation language, the result was, in effect, a special purpose shell. The generic tasks thus defined were nicely reusable, could be used to provide explanations, and were of great utility for system development.

On the other hand, the early work in KADS made a conceptual separation of knowledge and implementation [Wielinga et al, 1992]. However, the problem of assigning methods to tasks was made simpler by the provision of a library of interpretation models, a skeletal model at a high level of abstraction, which could, suitably instantiated, be used as a governing model over a number of domains. To be sure, these interpretation models were seen as tools which could be used to develop the conceptual models which are the products of the anal ysis phase of KADS; nevertheless, the existence of a library, suitably taxonomized, had the agreeable effect of cutting down the knowledge engineer's options, which would then ameliorate the transition from conceptual model to implementation simply by its regimentation of the conceptual models that got produced. The amelioration was further enhanced by the insistence that the conceptual model (or at least the model of expertise) was a functional specification of the problem-solving part of the target system -- i.e. the model of expertise was not explicitly a cognitive model of the expert, but was biased towards the intended function of the target- system. Hence the conceptual separation of the knowledge and the system design was never as great as it might have been. In this way, KADS was able to go some way towards tackling what Chandrasekaran calls the interaction problem, the problem that the representation of knowledge should be related to its use.

Nevertheless, that separation was there Chandrasekaran's early version of generic tasks and the KADS methodology occupy opposite ends of a spectrum. The conceptual duality characteristic of KADS has been seen as having advantages and disadvantages. The chief advantage was the freedom the knowledge engineer was given to model the expertise; the disadvantage was the dual —- since the conceptual models were not executable (necessarily), implementation remained an extra step. However, the preferred methodological point of view of the KADS project was that the design step was structure preserving. This methodological stipulation was regarded as unnecessarily strict in some quarters (where the systems designer was to have maximal leeway to mangle the conceptual model in the cause of efficient implementation), but had four clear advantages. Firstly, the design step was rendered easier than, for example, starting from scratch. Secondly, the provision of knowledge level explanations by the system would be thus facilitated. Thirdly, the conceptual model could be used to guide the use and deployment of knowledge acquisition tools (this possibility was explored to a greater depth in the ACKnowledge project [Shadbolt and Wielinga, 1990; van Heijst et a/., 1992; O'Hara, 1993]). And fourthly, the conceptual model could be used as a debugging aid — problems could be traced back to inconsistencies in the conceptual model. Hence even the KADS project, at least on its preferred methodology, induced a fairly strong connection between the knowledge level description of the expertise, and the structure of the developed system.

A similar story could be told for the other approaches in this field. If the early generic tasks of Chandrasekaran could be seen as 'black boxes', Steels' approach [Steels, 1992] is to construct what he calls 'white boxes'. Here, applications are described at the knowledge level, but the knowledge level components have associated with them execution level objects. Hence, when the application is being developed, the application need only be specified at the knowledge level, while these decisions will have repercussions at the lower levels via an interactive graphical workbench.

The latest view from Ohio on generic tasks sees them rather more as loose assemblages of methods, rather than as method-specifying. Task structure analysis [Chandrasekaran, 1990] redraws generic tasks as useful combinations of tasks, methods, and (recursively) subtasks. Tasks are seen as families of instances of a type of problem; methods are more or less abstract specifications of classes of objects and operators which can solve problems. On the basis of experience and research, methods can be associated with tasks. So, for example, deciding to perform a task with a particular method will to an extent at least dictate the subtask structure subtasks may be required to isolate the objects and operators that would be appropriate in the particular context.

As a final example, the problem-solving method view at the Knowledge Systems Laboratory at Stanford developed the following hierarchy of concepts. At the top there is the task; this is an activity in the real world ([Puerta et al., 1992] gives the examples of word processing and job scheduling). Then there is the method, which is a 'procedure that implements an abstract model of problem solving and that is applicable to a class of tasks'. Below that is the subtask, identical to a task except in its occurrence in task decompositions, and finally there is the mechanism, which is a method which does not decompose a task into subtasks (i.e. it is at the bottom of the tree). The criterion by which a method becomes a mechanism is a purely instrumental one: 'if decomposing a method into subtasks does not provide corresponding mechanisms for each subtask that can be easily reused, the method is not decomposed and it becomes a mechanism'. It is defined by three components: an input/output declaration; a collection of data structures; and a control-flow configuration (there is also an associated interface specification, but that is not integral to the mechanism). Hence mechanisms are not unlike the early version of generic tasks. Configuring methods to solve tasks, at least as performed by PROTEGE-II, is done using a library of mechanisms. This library contains mechanisms as basic elements, but indexes these mechanisms on the basis of their three components using tasks, domain ontologies and task models. Hence the library can provide indexing facilities enabling the retrieval of a group of candidate mechanisms for a particular task, while simultaneously helping to provide criteria for the selection of a particular mechanism from the suite available.

3 Knowledge Level Modelling Primitives and Their Constraints

We can see a common strand in all these approaches. The knowledge engineer is to be given the option at least of choosing some knowledge level primitive from a library, which in turn will have important effects upon the implementational level. Hence the implementational detail can be left to one side until the knowledge analysis is done, and when the analysis is done, the result will be a partial specification of the nuts and bolts (there will always be input to technical design from areas other than the knowledge level, of course; architectural issues, managerial issues and various other non-functional issues will all impinge on the technical design, so the specification will only be partial). Steels for one is prepared to claim that even non-programmers will be able to put together some systems [Steels, 1992].

Now we can begin to address the question with which we began: how good a cognitive model of the expert do we get when we put together this sort of knowledge level model for the purposes of KBS development? Are the modelling primitives set out by these modelling approaches general primitives for cognitive description, or are they idiosyncratic to each approach? Methods available in the approach are bound to affect choices to be made at the knowledge level. If some class of algorithms is unavailable to you at the lower level in some methodology, you are hardly likely to include a knowledge level specification of that class further up. Hence, we can agree that the ability to build a system is greatly enhanced by a knowledge level model of the appropriate type, but this leaves the question as to the psychological significance of these models open.

Certainly, some of the approaches here do make strong claims for their psychological import. Chandrasekaran wishes to ground the generic tasks in fundamental categories. [Puerta et al., 1992] notes that there is a lot of common ground between these methods, and is 'encouraged' by that fact: why be encouraged unless there is a genuine convergence on something substantial? The convergence is summed up by [Wielinga et a/., 1992]:

> Although terminology is different, a common view appears to emerge based on the idea that different types of knowledge constitute the knowledge level and that these different types of knowledge play different roles in the reasoning process and have inherently different structuring principles.

The idea seems increasingly that there is a knowledge level reality which is constituted by primitives such as are developed in the approaches under discussion.

Is this idea sustainable? The problem in sustaining the idea stems from the requirement that the knowledge level models should constrain the lower level, implementational, possibilities it is not clear whether the locus of the posited knowledge level reality is in knowledge engineering practice or in the competence of experts. Let us define some sort of generic vocabulary. Let us call a task a real world conceptualization of a problem — tasks might include design or classification, construed independently of any of the modelling methodologies of the type we have discussed above. The idea is that this notion captures the requirements of the prior notion of task, independent of AI methodologies. Let us call a method a conceptualization of a problem that renders its solution practical (e.g. some sort of implementation module). Finally, let us call a Knowledge Level Primitive (KLP) a technical term of an approach that attempts both to conceptualize a problem in the real world, and to constrain the methods appropriate for that problem. Our question is: can KLPs give an account of tasks while simultaneously satisfying the requirement that they enable a suite of methods to be assembled to solve problems[7]

Now we see the difficulty we have in maintaining that KLPs can attach themselves firmly both to tasks and methods. A task can be done in an unlimited number of ways; all a task is is a means of referring to a problem in a general way. Typically, criteria for calling a problem an example of one task rather than another are various, and will include social factors, contextual factors, historical factors (how similar problems have been solved before), and so on, as well as general input/output constraints. When the question arises as to how to solve a problem computationally, the problem will still have to be configured into a structure of KLPs. This configuration may or may not depend on the problem's conceptualization in terms of the task vocabulary. Suppose the configuration does not depend on the terms of the task vocabulary. Then it is hard to see that KLPs have anything to say about tasks.

But equally it is difficult to see how the configuration can depend on the terms of the task vocabulary. The KLPs abstract away from a large number of the issues relevant to task identification, in order to be able to concentrate their fire on constraining the available methods Clancey has been alive to this point for some time [Clancey, 1985, p.313]. The results of the Sisyphus problem, and the different configurations thereof, make interesting reading [Gaines, 1991]. The room allocation task (the benchmark of the Sisyphus experiments) can quite easily be read as the problem of applying a classify KLP to classify workers by the rooms they ought to have, or of applying an assembly KLP to assemble a series of consistent hypotheses about where workers ought to go, for example. Doubtless there are many more ways of conceptualizing the original problem. The significance of this point should not be underestimated — there is a tenuous connection only between tasks and KLPs.

On the other hand, in the downward direction, the connection between KLPs and methods is a very strong one indeed. This is how a KLP-approach can constrain methods. A KLP will determine a class of methods that can be used — possibly a singleton class, but nevertheless a small class. The result of this is that KLPs can be indexed by the methods they legitimize — the content of the KLP, what the KLP will do, will depend crucially on the methods underlying it. It is not clear how any other consideration can possibly have a bearing on the content of the KLP. Suppose, for example, the KLP was partially indexed by the tasks it was used to solve. Then, since a task can be solved by a number of methods, and a method can be used in the solution of a number of tasks, if the past experience of the use of a particular KLP in the context of a particular problem conceptualization was to count for too much, the result may well be that the KLP wouldn't get used in different tasks where it might be efficacious (none of this is intended to entail that one should not take note of where and how a KLP has been successfully used, and therefore not *learn* from that experience).

KADS, perhaps, might be excepted from a number of these points, since it is the methodology which most enables the knowledge engineer to pursue an attempt to describe and formalize a real-world task *without* being 'shackled[7] by the KLP/method identification. However, although it does allow a conceptual model to be constructed without recourse to interpretation models (which are but 'tools'), the KADS methodology shorn of interpretation models is the KADS methodology without its most distinctive aspect (not forgetting the link between interpretation models and the interaction problem, as discussed above).

In short: KLPs can be defined by the methods that they employ. Tasks cannot.

4 An Alternative View of Tasks and Methods

We have attempted to establish that someone who wishes to make strong psychological claims for the knowledge level models she has developed in the course other KBS development will be impaled on the horns of a dilemma. The dilemma for her is this: how strong a connection should there be between the psychological primitives being used and the KLPs of her preferred problem-solving approach? The first horn of the dilemma is that she claims that KLPs *are* psychological primitives, and, say, generic tasks or interpretation models are, *inter aha,* actual 'descriptions' of psychological structures employed by and instantiated in the expert in problem-solving. The claim here is *that we actually solve problems in this way.* But we don't, for example, generally solve a design problem using exactly those methods Chandrasekaran sets out in his analysis of the generic task design [Chandrasekaran, 1990] — and even if we did, there would still be the possibility of a radically new way of designing being used in the future that would not be entailed by the generic task theory of design. The second horn of the dilemma is that KLPs are *not* psychological primitives. So, say, the generic task design just cuts across the explananda of psychology — this is suggested by the possibilities of reconfiguring problems in terms of different combinations of KLPs. Hence the explanandum — real world design is just in a different category from the loose collection of methods grouped together under the rubric of the generic task of the same name. So on the first horn of the dilemma, the KLP is taken as a potential (and inadequate) *definition* of the task; in the second case, the KLP is taken as a model of a pre-existing structure, and its inevitable implementational bias renders it incapable of fulfilling that function.

How can this dilemma be avoided? The key is to note that the dilemma is premissed on the assumption that the explananda of psychology beliefs, desires, complexes are determinate, genuine, real and independent of outside interpretation. When a psychological theory is produced, the truth or falsity of the theory can be 'read off' the psychological structures that are empirically determinably there. This is not to say that there might not be problems in interpreting the empirical data of reaction times, etc.; but does mean that these problems are in principle at least surmountable.

This psychological realism allows only two possible views of AI. The first view is that AI has no interesting psychological value AI may receive *input* from psychology, but will provide no *output* into psychology. If an AI system works it works. If it doesn't it doesn't, and there's an end on't, as Dr Johnson would say. This is a highly instrumental view that makes no psychological claims at all. However, this view is a very unambitious view we might say that it is unfortunate to do all this modelling of experts without getting any feedback into psychology at all. We avoid the dilemma sketched above, but at the cost of surrendering any independent interest for our expert models. This is the official KADS line [Wielinga *et al.,* 1992, p.II], but note the tension with the quote in section 3.

The second view is a cognitive realism that might claim that experts *really do* manipulate such things as knowledge sources and metaclasses, for example. Chandrasekaran makes analogous claims as hypotheses [Chandrasekaran, 1990, p.60], but we have attempted to establish that these claims are implausible. The cognitive realist is impaled on our dilemma.

The alternative suggestion we wish to make for avoiding the above dilemma is to endorse a cognitive *anti-rcahsm.* Under this view, the explananda of psychology are no more nor less than by-products of the process of interpretation that must take place in order for any social intercourse to flourish, and therefore are functions partly of the interpreters, or the environment, of the object system.

In the current context, we wish to claim that the realist has the wrong picture. The realist postulates three *spaces:* a task space, a KLP space and a method space

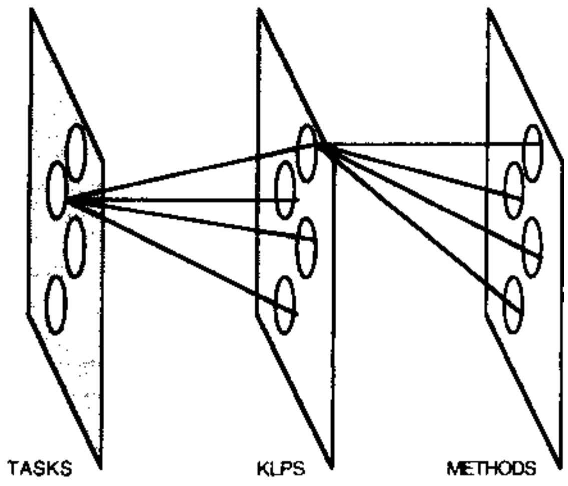

Figure 1: The Realist View of Tasks, KLPs and Methods

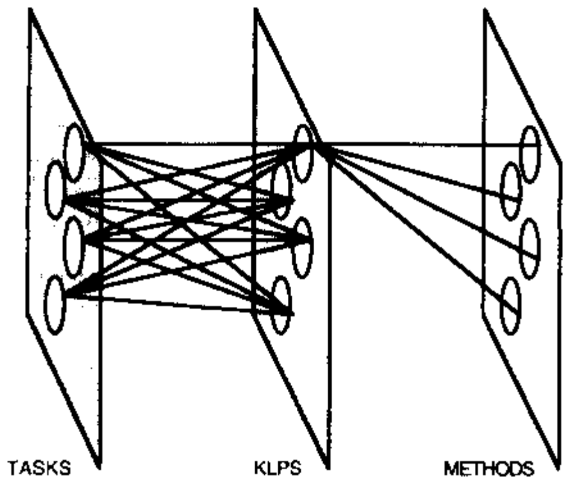

Figure 2: The Danger of Assuming the Existence of a Task Space

(fig.1), where a space is to be understood as objectively organized and indexed. KLPs work by being determinate filters on the method space. They then have *psychological significance* if tasks are determinate filters on the KLP space. Our contention is that tasks are absolutely not determinate filters on the KLP space. If a task space is to be posited, the result will be the sort, of cat's cradle envisaged in fig.2.

The anti-realist solution to the problem is to maintain that the task space, in this objective sense, is illusory. There are too many influences on the shape of a task for it to remain a coherent entity in all circumstances; there is no simple, determinate structure for a task In Wittgenstein's terminology, the notion of'task' is a *family resemblance concept* [Wittgenstein, 1953], i.e. every example of a particular task bears a strong connection to some other example of that task, but there is riot-necessarily any condition which will determine the extension of the terms. Certainly one could concoct some huge disjunction to cover all the past examples, but even that will not necessarily determine the status of any future examples. Nevertheless — and we emphasise this — despite the lack of necessary and sufficient conditions here, there *is* stability and unity here, and that stability and unity is of vital importance when it comes to the reliability of knowledge engineering practice.

So the psychological significance of a KLP comes about as follows. If psychological phenomena are socially constructed and functions of context, then one has to take notice of the context in which cognitive models of the sort in question arise. Specifically, the reason that one wants a model of the expert in Al is that one wants to build something that will produce the same behaviour (there may be more constraints than this, of course; one may want something that will, say, manipulate the same information in producing that behaviour). In this context, it just is the case that no psychological explanation could provide any more germane information. The realist can agree with that, of course. We, however, wish to maintain that, further, there is no psychological explanation whose context of evaluation is distinguished or privileged, and therefore no psychological explanation that can be seen as being the 'correct/ explanation.

This anti-realist view leaves open the possibility of a sliding scale of psychological significance. At one end, there is the significance a model would have if it allowed a simulation of the target expertise to be built. At the other, there is the significance a model would have if it turned out to be impossible to built such a simulation without using that model. In the middle, there are various degrees of significance, which might depend, for example, on whether the machine simulation reproduced various empirical conditions (e.g. reaction times, patterns of error) sufficiently well. All these notions define different senses of 'psychological significance[1]; any one might be appropriate for various purposes.

5 Example: Abstraction

We can illustrate the import of these ideas by considering a component of a well known KLP, that of *heuristic classification,* first characterised in [Clancey, 1985], where Clancey was careful to relate components of the model to supposed psychological processes and categories of problem solving. The KLP component of the heuristic classification model which we shall consider is abstraction[1]. This is the process by which observations or data are transformed into abstract observations or findings. The process of trcinsformation can be effected by a number of means. One of these is the process of qualitative abstraction. A famous example is shown below — here we move from a quantitative observation to a qualitative finding.

```
if white blood cell count < 2500
then low white blood cell count
```

'Note that a KLP can be composed of other KLPs.

We may ask the question: what has the abstraction step in such models to do with any psychological explanation of problem solving? It is our contention that we can find a variety of interpretative stances that make sense of this abstraction, all of which can lay claim to psychological import.

Suppose that a rule such as that shown above was the result of a knowledge engineer's analysis of a behavioural protocol. Such a knowledge level description arises as a result of third person interpretation of a problem solving trace. This is a classic form of psychological data — although not without methodological concerns, the status of such traces and their interpretation has a long and respected history in psychology [Ericsson and Simon, 1984].

However, the same rule might have been derived from a self report session with an expert. Here the expert explicitly describes the rule. Of course, first person reports are also beset with methodological problems. Nevertheless they have often been taken as legitimate accounts of psychological processes.

A third way in which the rule might have arisen could have been via one of the contrived methods of knowledge acquisition [Shadbolt and Burton, 1989]. For example, the repertory grid technique [Shaw and Gaines, 1987] is a method in which an expert, makes rating and ranking judgements. These judgements are then used as a basis for determining implicit structure. A rule such as that of abstraction could be induced through the process of repeated ranking of elements (perhaps individual case histories) along dimensions which are elicited from the expert. One might say in this case that the knowledge of the rule is tacit. Nevertheless it has realisation just in so far as the expert makes consistent categorisations which always place patients with a white blood cell count within a certain range labelled as "low".

Finally we might consider the case in which we construct a neural network account. This might succeed, given a set of training instances, in determining various output classes one of which might, be "low white blood cell count'\ In this case we might claim that we have an account which is defined only in terms of a mechanism, perhaps neurologically plausible, that delivers the "right" response. Again such accounts have been offered as having a psychological status.

These examples illustrate how the same structure can serve as a locus for a variety of accounts which can all deliver domain level content. At the same time these accounts all have some psychological pedigree —- they could serve as the ingredients of various types of psychological explanation. The interpretative stance with respect to these matters is pervasive. It is important to disentangle these various threads in the growing discussion surrounding the accounts of problem solving methods in KBS and AI work[2].

References

[Chandrasekaran, 1983] B. Chandrasekaran. Towards a Taxonomy of Problem Solving Tvpes. AI Magazine, 4(1):9-17, Spring 1983.

[Chandrasekaran, 1990] B. Chandrasekaran. Design Problem Solving: A Task Analysis. AI Magazine, 11(4):59 71, Winter 1990.

[Clancey, 1985] William J. Clancey. Heuristic Classification. Artificial Intelligence, 27:289-350, 1985.

[Ericsson and Simon, 1984] K.A. Ericsson and II.A. Simon. Protocol Analysis: Verbal Reports as Data. Cambridge, Massachusetts, 1984.

[Gaines, 1991] Brian Gaines. The Sisyphus Problem Solving Example Through a Visual Language with KL-ONE-Like Knowledge Representation. In Sisyphus Working Papers Part 2: Models of Problem Solving, Glasgow, U.K., 1991. University of Strathclyde.

[O'Hara, 1993] Kieron O'Hara A Representation of KADS-I Interpretation Models Using a Decompositional Approach. In Christiane Lockenhoff, Dieter Fensel and R.udi Studer (eds.) Proceedings of the 3rd KADS Meeting, Siemens AG Munich, 1993.

[Puerta el al., 1992] Angel R. Puerta, Samson W. Tu and Mark A. Musen. Modeling Tasks with Mechanisms. Technical Report KSL-Report 92-30, Knowledge Systems Laboratory, Stanford University, 1992.

[Shadbolt and Burton, 1989] N.R. Shadbolt and A.M. Burton. Knowledge Elicitation. In Wilson and Corlett (eds.) Evaluation of Human Work: Practical Ergonomics Methodology Taylor &: Francis, 1989.

Shadbolt and Wielinga, 1990] Nigel Shadbolt and Bob Wielinga. Knowledge Based Knowledge Acquisition: The Next Generation of Support Tools. In Bob Wielinga, John Boose, Brian Gaines, Guus Schreiber and Maarten van Someren (eds.) Current Trends in Knowledge Acquisition, 10S Press, Amsterdam, 1990.

[Shaw and Gaines, 1987] M.L.G. Shaw and B.R. Gaines. An Interactive Knowledge Elicitation Technique Using Personal Construct. Technology. In A. Kidd (ed.) Knowledge Acquisition for Expert Systems: A Practical Handbook, New York, 1987.

[Shatter, 1987] Philip E. Shatter. Building Expert Systems: Cognitive Emulation Chichester, 1987.

[Steels, 1992] Luc Steels. Reusability and Configuration of Applications by Non-Programmers. VUB AI Memo 92-4, Free University of Brussels, 1992.

[van Heijst et al., 1992] G. van Heijst, P. Tersptra, B. Wielinga and N. Shadbolt. Using Generalised Directive Models in Knowledge Acquisition. In Th. Wetter, K.-D. Althoff, J. Boose, B.R. Gaines, M. Linster and F Schmalhofer (eds.) Current Developments in Knowledge Acquisition — EKAW '92, Springer Verlag, Berlin, 1992.

[Wielinga et al., 1992] B.J. Wielinga, A.Th. Schreiber and J.A. Breuker. KADS: A Modelling Approach to Knowledge Engineering. Knowledge Acquisition 4:5 53,1992.

[Wittgenstein, 1953] Ludwig Wittgenstein. Philosophical Investigations. Blackwells, Oxford, 1953.

[2]We would like to thank the IJCAI referees for their helpful comments.