Mind as Machine: Can Computational Processes be Regarded as Explanatory of Mental Processes?

Kieron O'Hara

Worcester College, Oxford

Abstract

K.M. O'HARA
D.PHIL. THESIS

WORCESTER COLLEGE
TRINITY, 1994

MIND AS MACHINE: CAN COMPUTATIONAL PROCESSES BE REGARDED AS EXPLANATORY OF MENTAL PROCESS?

The aim of the thesis is to evaluate recent work in artificial intelligence (AI). It is argued that such evaluation can be philosophically interesting, and examples are given of areas of the philosophy of AI where insufficient concentration on the actual results of AI has led to missed opportunities for the two disciplines — philosophy and AI — to benefit from cross-fertilization. The particular topic of the thesis is the use of AI techniques in psychological explanation. The claim is that such techniques can be of value in psychology, and the strategy of proof is to exhibit an area where this is the case. The field of model-based knowledge-based system (KBS) development is outlined; a type of model called a *conceptual model* will be shown to be psychologically explanatory of the expertise that it models.

A group of major philosophies of explanation are examined, and it is discovered that such philosophies are too restrictive and prescriptive to be of much value in evaluating many areas of science; they fail to apply to scientific explanation generally. The importance of having sympathetic yardsticks for the evaluation of explanatory practices in *arbitrary* fields is defended, and a series of such yardsticks is suggested. The practice of providing information processing models in psychology is discussed.

A particular type of model, a *psychological competence model*, is defined, and its use in psychological explanation defended. It is then shown that conceptual models used in model-based KBS development are psychological competence models. It follows therefore that such models are explanatory of the expertise they model. Furthermore, since KBSs developed using conceptual models share many structural characteristics with their conceptual models, it follows that a limited class of those systems are also explanatory of expertise. This constitutes an existence proof that computational processes can be explanatory of mental processes.

Acknowledgements

Many thanks go to all the people who have given useful comments on the various philosophical ideas in this thesis. No doubt the following list is not comprehensive: Selmer Bringsjord, Paul Feltovich, Ken Ford, Robert Hoffman, Robert Kirk, Barry Silverman, Peter Smith, Tim Williamson. Particular thanks to Tom Scutt, with whom I have had many fruitful discussions about cognitive science over the past three years. Thanks also to the following, discussions with whom helped me develop ideas in the field of knowledge engineering: Willem Jonker, Marion Koopman, Mandy Mepham, Enrico Motta, Han Reichgelt, Jan Willem Spee. Thanks to Joseph Goguen, who nursed me through the basics of software engineering, and to the audiences at the 3rd Human and Machine Cognition Workshop, Seaside, Florida, the AI Research Seminar at the Laboratory of AI Research, University of Leeds, the Cognitive Science Seminar at the Dept. of Computer Science, University of Birmingham, and the Staff-Student Seminar, Dept. of Philosophy, University of Nottingham, who asked interesting and difficult questions when ideas from the thesis were presented. My first supervisor, Andrew Rein, helped me find my feet in the philosophy of mind, after my formative years were spent in logic. My third supervisor, Elizabeth Fricker, was instrumental in helping me to bring my ideas from knowledge engineering into a philosophical context. Kathleen Wilkes was also my supervisor for an all too brief period. Gregory McCulloch was very helpful in keeping me in touch with philosophy when I moved to the Dept. of Psychology at the University of Nottingham, and asked the awkward questions about explanation that prompted the main direction of this thesis. The secretary of the sub-faculty of Philosophy, Jane Hardie, has been invaluable in keeping me abreast of my administrative status and requirements.

Three people have been very important in the development of the position I defend, and to them I am particularly grateful. Nigel Shadbolt, Professor of Intelligent Systems at the University of Nottingham, was foolhardy enough to employ me, patient enough to listen to my uninformed ramblings on the subject of AI, and kind enough to encourage the philosophical aspects of my work. Most of what I know about AI I have gleaned from working with him. My fourth supervisor, Martin Davies, who has given very freely of his time, has been a mine of useful comments and discussions on my ideas. His contributions throughout the thesis, both to the philosophical argumentation, and on the practical side of getting one's ideas across to an audience, are too numerous to

list. I am particularly grateful for his ability to spot an argument lurking at the bottom of an unfocused paragraph of mine. Finally, my thanks to Rebecca Hughes, of the Centre for English Language Education at the University of Nottingham, whose advice, guidance and support has been invaluable, not least in helping me place my philosophical ideas in wider contexts.

In the months leading up to the completion of this thesis, my father, Derek O'Hara, and my grandparents, Joe and Florrie Dixon, died. They all gave much love and encouragement, and looked forward to the end product of my years of labour with anticipation and pride. I'd like to dedicate that end product, such as it is, to them, with all my love.

Table of Contents

Preface ..... **vii**

Analytical Table of Contents ..... **ix**

Chapter One: A Metathesis Concerning the Philosophy of AI **23**

1.1 The Importance of Being Artificial 23

1.2 Three Examples of Philosophical Debate About AI 26

1.2.1 Example: The Turing Test 26

1.2.2 Example: The Problem of Induction 33

1.2.3 Example: Narrow versus Wide Content 38

1.3 The Philosophy of AI 44

1.3.1 AI and Intentionality 45

1.3.2 Two Views of AI 47

1.4 Example: Modelling Expertise 51

Chapter Two: Of MYCIN Men: Knowledge-Based System Development Methodologies **52**

2.1 What Knowledge-Based Systems Are 52

2.1.1 Brief Historical Notes 53

2.1.2 Knowledge Representation 55

2.1.3 Verification and Validation 60

2.1.4 Explaining KBS Output 61

2.1.5 Knowledge Acquisition 64

2.2 Model-Based Knowledge Acquisition and KBS Development 65

2.2.1 KBS Development as Model Refinement 66

2.2.2 Conceptual Models 70

2.2.3 Two Philosophies of Modelling 76

2.2.4 Knowledge Acquisition Techniques 78

2.2.5 The Psychology of Knowledge Acquisition 80

Chapter Three: Scientific Explanation **83**

3.1 What We Are Talking About 84

3.1.1 Scientific Explanation and Scientific Understanding 84

3.1.2 Explanation as a Process and Explanation as a Product 87

3.1.3 Full and Partial Explanations 89

3.1.4 Good Explanation and Bad Explanation 91

3.2 Top Down Accounts of Explanation 92

3.2.1 Review of Some Top Down Accounts 92

3.2.2 The Lack of Top Down Consensus 96

3.2.3 Ideals of Explanation Do Not Explain Alternative Explanatory Practices 100

3.2.4 Actual Explanatory Practice Needs a Philosophical Account 104

3.2.5 Two Top Down Responses 106

3.3 A Bottom Up Account of Explanation 109

3.4 Causal and Non-Causal Explanation 114

3.4.1 Non-Causal Explanation 114

3.4.2 Normative Explanation 116

3.4.3 Causal Explanation 119

3.4.4 Program Explanation 120

Chapter Four: Psychological Explanation 122
4.1 Computational Models in Psychology 122
4.2 Levels of Explanation 125
4.2.1 Marr's Three Levels 126
4.2.2 Explanation at Level 1.5 129
4.2.3 Explanation at Level 1.6 135
4.2.4 Models of Expertise 137

Chapter Five: Conceptual Models, Competence Models and Psychological Explanation 144
5.1 Competence Models and Psychological Explanation 145
5.1.1 Models of Problem-Solving 145
5.1.2 Psychological Competence Models 153
5.1.3 The Explanatoriness of Psychological Competence Models 156
5.2 Conceptual Models and Competence Models 166
5.2.1 Modelling Methodologies and the Alternatives 166
5.2.2 Conceptual Models and Competence Models 169
5.2.3 Off-the-Peg Models 176
5.2.4 The Argument Completed: Conceptual Models, Knowledge-Based Systems and Psychological Explanation 179
5.3 Knowledge-Based Systems, Normative and Causal Explanation 180
5.3.1 Normative Explanation? 181
5.3.2 Causal Explanation? 182
5.3.3 Conceptual Models and KBSs are Both Normatively Explanatory and Causally Explanatory 185

Chapter Six: Conclusion: Expertise and Artificial Intelligence 187
6.1 Review of Chapters One-Five 187
6.2 Artificial Intelligence as Explanatory 189
6.3 Expertise in Context 190
6.4 Normative and Causal Explanations 191

References 194

Preface

This thesis is the result of quite a long intellectual journey. As I began my D.Phil. research in the philosophy of AI with Andrew Rein in 1986, I was intrigued by the possibilities of AI. At that time, the thought experiments of Dennett, Hofstadter and Searle, together with the genuine excitement that accompanied the first major books on connectionism by Rumelhart, McClelland et al, seemed to suggest that an exciting science fiction world was just around the corner. On the other hand, sceptical Cassandras such as Roy Harris, and Baker and Hacker, were making play with the notion that human behaviour — particularly *linguistic* behaviour — was intrinsically too complex to be mechanistically describable or reproducible. The assumptions of the two camps seemed irreconcilable; I did what seemed logical and turned to the discipline itself in an attempt to resolve the matter.

I was fortunate to go to Eugene Charniak and Drew McDermott's *Introduction to Artificial Intelligence*, still today one of the finest textbooks on AI around, even after ten years. I was amazed that the work reported there, on the properties of Lisp, vision, language parsing, search strategies, memory, uncertain reasoning, planning, etc., was simply absent from the work both of the thought experimenters (pro- and anti-AI), and the neo-Wittgensteinians. I began to wonder what relevance their *a priori* ruminations had to the functioning discipline of AI, and I resolved to investigate in a more practical fashion.

Hence, in 1988-9, I took an M.Sc. in software engineering at the Programming Research Group at Oxford, and in 1990, took up a post as a researcher into knowledge acquisition for knowledge-based systems in the AI Group, Dept. of Psychology, University of Nottingham. Gradually, I have come to realize that the gap between the philosophy of AI and the practice of AI is even wider than I anticipated. In particular, what seems to be missed from most philosophical discussion is an appreciation of the importance of having a functioning system. It is important for AI, since naturally a program that doesn't work is useless. But it is equally important for philosophy and psychology, because an important factor in the evaluation of proposed psychological mechanisms is the question of whether they could produce the required output in real time under realistic memory constraints. The computer is the only tool we have for attempting to answer such questions.

So when a thought experiment begins "Suppose we had a machine that ...", that assumption covers a multitude of questions related to the actual construction and operation of such a machine. In AI, we often learn more from the process of building systems than we do from the finished system itself.

Finally, one attractive aspect of the subfield of AI reported in this thesis is that it is relatively mundane, and yet (as I hope to show) psychologically significant. Much attention has (rightly) been given to high-profile work in AI and cognitive psychology which purports to uncover *the* secrets of cognition (e.g. the work of Newell, Simon and Shaw, the work of Brooks, work on SOAR, work on PDP, the work of Edelman). However, as a result of my work at Nottingham, I am firmly convinced that, if you want to know about expertise, you would be well-advised to eschew the monolithic approach and go to knowledge-based systems research. It will tell you nothing about vision, or language comprehension, or the operation of the brain; but it gives an unrivalled account of expertise. This suggests an agreeable picture of AI as a heterodox activity, with different people working under different sub-paradigms, all contributing to a whole which is greater than the sum of its parts.

Analytical Table of Contents

Chapter One: A Metathesis Concerning the Philosophy of AI

Two competing styles of philosophy are discerned, and evaluated, in the context of the philosophy of AI.

1.1 The Importance of Being Artificial

The competing notions of *top down* and *bottom up* philosophy are introduced. Top down philosophy involves the development of a theory and the application of that theory to various problems or areas of interest. The result of top down philosophy tends to be that discussion is driven by philosophical concerns. Bottom up philosophy, in contrast, is driven by concerns that have arisen in non-philosophical contexts. In particular, a bottom up approach to the philosophy of AI will tend to attempt to interpret the actual results of AI research (as opposed to interpreting unrealistic thought experiments). This should make bottom up philosophy more influential on the practice of AI.

1.2 Three Examples of Philosophical Debate About AI

To indicate that there can be problems with top down philosophy, three examples of the application of philosophical theory applied to AI are given.

1.2.1 Example: The Turing Test

For example, most philosophical discussion of the Turing test has concentrated on the alleged over-generosity of the test in ascribing intelligence to intuitively non-intelligent systems. However, consideration of standard AI systems shows that the Turing test is also biased against existing and foreseeable systems, by disregarding the likely ways that intelligent machines would be used.

1.2.2 Example: The Problem of Induction

As another example, we note that the field of machine learning poses an extra problem for the philosophy of induction; not only do inductive hypotheses have to be justified, but also such hypotheses have to be selected from other

competing hypotheses. This problem of hypothesis selection has generally been neglected by philosophers, despite its pressing nature.

1.2.3 Example: Narrow versus Wide Content

As a third example, we consider some arguments that have followed on from Putnam's Twin-Earth thought experiment, and from the eliminativist arguments, both of which have been thought to have methodological implications for AI. The implications turn out to be impractical, (a) because there is no well-developed computational paradigm that will exactly fit the requirements of these arguments, and (b) because standard, 'vanilla flavoured' symbolic AI is the only branch of AI with relatively tractable semantics (this at least in some cases will be important). Furthermore, the arguments we consider have a very monolithic view of which computer programs could have psychological interest.

1.3 The Philosophy of AI

Having established that bottom up philosophy has some applications in the philosophy of AI, we move to consider a definition of what AI is.

1.3.1 AI and Intentionality

It is unhelpful for AI to be viewed as an attempt to produce intentionality, or human-like intelligence.

1.3.2 Two Views of AI

On the contrary, AI should be seen as research into a disparate collection of computational problem solving methods. In fact, because of the unclarity of philosophical definitions of such concepts as 'intelligence', this way of viewing AI is likely to make important contributions to the general philosophical discussion.

1.4 Example: Modelling Expertise

Our aim will be to look at the particular AI practice of building knowledge-based systems, with a view to looking at its psychological significance.

Chapter Two: Of MYCIN Men: Knowledge-Based System Development Methodologies

The ideas underlying the field of KBS development are put forward in this chapter.

2.1 What Knowledge-Based Systems Are

We begin with the general ideas behind KBSs.

2.1.1 Brief Historical Notes

Knowledge-based systems (KBSs) or expert systems are introduced in their historical context. A KBS is a computer system that attempts to reduplicate expert performance by the manipulation of large quantities of knowledge about the domain in which the expert has her expertise. These systems are an increasingly important technology in science and industry.

2.1.2 Knowledge Representation

Knowledge is stored in KBSs using a special-purpose computer language called a knowledge representation language (KRL). KRLs are usually based either on logic or on 'chunks' of knowledge called frames (or both). Neither formalism is entirely satisfactory, either in terms of efficiency, semantics, or transparency to an expert.

2.1.3 Verification and Validation

The ability of a KBS to do the job it is intended for is estimated by a process called verification and validation (V&V). The requirements of V&V mean that often high level descriptions of the KBSs functions are required, as well as the implementational (programming language) specification.

2.1.4 Explaining KBS Output

Furthermore, the output of a KBS has to be explainable by the system for the system's user. This is so that controversial output can be verified (and, if necessary, overridden). This also means that the knowledge stored in the KBS

has to be describable at higher levels that would simply be required for the KBS to produce the right output.

2.1.5 Knowledge Acquisition

These various requirements have led to the development of a field of AI practice called knowledge acquisition (KA). This is the practice of getting information out of the expert and into the machine. The KA process needs to identify which knowledge will be required for the functioning of the system, which knowledge is required for V&V, and which for explanation, and also needs to represent that knowledge perspicuously in the system's KRL. For this reason, KA is often referred to as a 'bottleneck' in KBS development.

2.2 Model-Based Knowledge Acquisition and KBS Development

We now turn our attention to a particular group of methods for developing KBSs, model-based methodologies.

2.2.1 KBS Development as Model Refinement

To get round the KA bottleneck, one type of KBS development methodology concentrates on the construction of models to guide the KA process. Three models are developed in the course of a KBS development project: a requirements specification, which only takes into account the task to be performed by the system; a conceptual model, which models the expertise required to perform the task; and a design model, which further considers the computational constraints on the system. The relationship between the conceptual model and the expertise is introduced and discussed briefly.

2.2.2 Conceptual Models

This is an important section, for it discusses in detail the properties of conceptual models; our claim in Chapter Five will be that such models are explanatory of expertise. A conceptual model is a model of the expertise required to solve a particular problem. It will not always be a model of a single expert. The knowledge acquired and put in the model can be typed epistemologically, resulting in a four-layer structure. The domain layer contains the static knowledge of the domain objects and their interrelations. The inference layer is a

specification of the basic inferences made by the experts in problem-solving; these inferences can be grouped together in an inference structure. The task layer contains the knowledge about how these basic inferences are grouped together to achieve goals. This knowledge controls the inferences. Finally, the strategic layer contains the knowledge about how to configure a set of goals to solve a large scale problem. It is arguable that there is not principled distinction between the task and strategic layers, and many models will not have a strategic layer at all. Karbach et al have shown that the four-layer model can be seen as representative. Groups of domain-independent skeletal models (containing inference and task layer knowledge) can be put together as model libraries, to ease the modelling task for knowledge engineers.

2.2.3 Two Philosophies of Modelling

Two ways of modelling expertise are introduced. The first way is to emphasize the similarities between the expert and other experts; the second is to emphasize the dissimilarities. Most KBSs are built using the first philosophy. This introduces the need for us to argue — as we will in §5.2.3 — that this philosophy is adequate for the explanation of expertise.

2.2.4 Knowledge Acquisition Techniques

There are various KA techniques to capture various types of knowledge. Unstructured interviewing, structured interviewing, self-reports, protocol analysis, card sorting, laddering and repertory grids are discussed. Repertory grids are particularly interesting because they can uncover knowledge in a form in which the expert will not typically recognize it.

2.2.5 The Psychology of Knowledge Acquisition

Much psychological investigation has been performed into the efficacy of these techniques. In particular, it has been noted that the contrived techniques (such as repertory grids) result in the acquisition of knowledge that is used in problem-solving (since the solution of the problem by the expert cannot be easily explained without the postulation of the knowledge), of which the expert is not conscious. Furthermore, KA techniques can often be used to critique faulty expert performance. Therefore, KA techniques will not automatically be used to create a model of the expert's conscious reasoning.

Chapter Three: Scientific Explanation

Having developed an account of conceptual modelling, we will try to show that such models are psychologically explanatory of expertise. The first step, then, is to say what is meant by 'explanation'. Our account must satisfy two constraints. On the one hand, since we have adopted the bottom up approach to philosophy, we should be certain that the philosophy of explanation that we endorse is maximally congenial to current scientific practice. On the other hand, the philosophy of explanation that we endorse must be *independently* convincing.

3.1 What We Are Talking About

We shall take explanations as being intended to confer understanding.

3.1.1 Scientific Explanation and Scientific Understanding

We decide only to discuss scientific explanation in this thesis, and leave unanswered the question as to how far our comments will apply to non-scientific explanation. Scientific explanation is keyed to the notion of scientific understanding. Again we do not define this notion rigidly. However, we do set the condition that scientific understanding is (at least partly) an instrumental notion. If a scientist understands some phenomenon, this means that she can carry out her research/development in the context of that phenomenon.

3.1.2 Explanation as a Process and Explanation as a Product

Explanations can be seen as processes or products. We show that conceptual models can be seen as either processes or products, and therefore the priority of process over product or vice versa is irrelevant for our results.

3.1.3 Full and Partial Explanations

A distinction is drawn between full and partial explanations depending on how far the explanation gets the scientist towards her goal. On this view, a conceptual model can be seen as a full explanation (assuming for the moment that it is explanatory at all) of the expertise required to solve a problem, or a partial explanation of how to solve the problem with current computational tools. We will take the psychological problem to be solved to be that of providing an

account of expertise, so that we will take conceptual models to be fully explanatory or not at all. Finally, we show that the full/partial distinction is not the same as an honorific distinction between scientific and non-scientific explanation.

3.1.4 Good Explanation and Bad Explanation

Explanations can be good (successful in conferring the intended understanding) and bad (unsuccessful). This applies to conceptual models too. Indeed, depending on how the conceptual model is to be taken, a model can at once be a good explanation of one phenomenon and a bad explanation of another.

3.2 Top Down Accounts of Explanation

We now examine some current theories of explanation in the philosophy of science.

3.2.1 Review of Some Top Down Accounts

We mention Hempel's deductive-nomological model which sees explanation as the provision of an argument whose conclusion is the explanandum; Salmon's statistical-relevance model which provides an argument in terms of the probabilities of the possible events; Brody's causal model and essential property model both try to marry a Hempelian account with Aristotelian insights; Lewis and Railton suggest that a full explanation of an event is nothing less than its complete causal history; Ruben's dependency model generalizes Lewis's account to include non-causal relations. These accounts are not descriptive, but are concerned with setting a standard.

3.2.2 The Lack of Top Down Consensus

One reason to reject top down views is that there seems to be no consensus emerging in the philosophy of explanation. If there were such a consensus, then we might represent ourselves as approaching a correct view, but not if not. Salmon discerns three types of top down view, and we note that all three views are vulnerable to particular types of counterexample. The deductivist view wants explanations to be arguments, and is vulnerable to counterexamples exploiting the asymmetries of explanation. The mechanist view wants explanations to posit

mechanisms, and is vulnerable to new contexts where such mechanisms are irrelevant. The pragmaticist view suggests an account that takes contextual elements into account, and is vulnerable to suggestions of 'anything goes' relativism. The pragmaticist view is most nearly similar to a bottom up view.

3.2.3 Ideals of Explanation Do Not Explain Alternative Explanatory Practice

A second reason is that ideal explanatory forms do not explain actual practices. If an actual practice departs from the ideal, then the ideal needs to be supplemented with an account of the differences. But this account is sufficiently explanatory of the practice in its own right. This is not true of a (good) bottom up account.

3.2.4 Actual Explanatory Practice Needs a Philosophical Account

A third reason is that there is an explanatory (or quasi-explanatory) practice going on in science that, even if it falls short of an ideal, stands in need of philosophical analysis. Scientists often criticize colleagues' explanations, and they standardly do not refer to the philosophical literature to do that. Indeed, were they to do this, they would have to resolve all the philosophical problems because of the lack of consensus discussed in §3.2.2. This critical practice is philosophically unexamined.

3.2.5 Two Top Down Responses

The top down theorist may respond in one of two ways. He may insist that his account of explanation defines the correct scientific practice. In that case, he falls foul of the arguments for bottom up philosophy in general discussed in Chapter One. Or he may claim that a discipline that fails to meet the ideal conditions is simply not explanatory — the instrumental account of understanding given in §3.1.1 is more of an engineering than a scientific norm. This begs the question, but risks leaving the philosophical account irrelevant.

3.3 A Bottom Up Account of Explanation

A bottom up account of explanation would need to give accounts of: (1) the audience and its interests; (2) the basis for the fruitfulness of the explanation;

(3) the responsibility the explanans has for the explanandum; (4) the contrastive class should be clear; (5) it should be clear that the account does not entail total relativism; (6) what is being explained should be clear.

3.4 Types of Explanation: Causal and Non-Causal

Another distinction is between causal and non-causal explanation. The very existence of the latter is a controversial issue.

3.4.1 Non-Causal Explanation

An argument, endorsed by both the mechanist Ruben and the pragmaticist Achinstein, is given to show that non-causal explanation is possible.

3.4.2 Normative Explanation

One type of non-causal explanation is normative explanation, the production of norms governing a practice. It is shown how normative explanations can be explanatory even when the practice fails to achieve its goal, or when the norms are not consciously followed.

3.4.3 Causal Explanation

Causal explanation is the production of a causally relevant property of the cause. The obvious type of causal relevance is causal efficacy. However, there will be problems is this is the only type of causal relevance.

3.4.4 Program Explanation

We circumvent this problem by using Jackson and Pettit's notion of a program explanation. Such an explanation produces a property of the cause that will make probable that a causally efficacious property is available.

Chapter Four: Psychological Explanation

Having discussed explanation generally, we now move on to the specific topics that are of relevance for psychological explanation.

4.1 Computational Models in Psychology

We first establish the possibility of computational models being explanatory in psychology. Computational models suggest mechanisms, that may be more or less metaphorical, for psychological behaviour. If they are content using models, they will also lead to the understanding of some behaviour in terms of the components of such behaviour.

4.2 Levels of Explanation

If a computational model is explanatory, then it has to be decided which of the various properties of the model are taken to be the explanatory ones.

4.2.1 Marr's Three Levels

Marr's three levels of explanation, the input/output level, the algorithmic level and the implementational level, are introduced. However, they are unlikely to be adequate for the purposes of KBS development.

4.2.2 Explanation at Level 1.5

Peacocke's level 1.5 is introduced. At this level, the explanatorily relevant parts of the model are the input/output relations, together with a specification of the information drawn upon. This turns out to be ambiguous between a specification of the body of information to be drawn upon, and a specification of the states of information to be drawn upon. Since this distinction is crucial for AI purposes, we need to resolve the ambiguity.

4.2.3 Explanation at Level 1.6

We introduce a new level of explanation, level 1.6, which is as level 1.5 with the addition of the constraint that not only must the body of information the model draws upon be specified, but so must the ontology of that body. Level 1.5 is defined for our purposes by its neglect of this ontology. If one takes a firmly extensional view of information, levels 1.5 and 1.6 collapse into each other; we will not be insisting on such a view. We also make no claim that level 1.5 as it is defined here is the same as the level 1.5 that Peacocke discusses.

4.2.4 Models of Expertise

To show that level 1.6 can be informative, we use the example of a model for KBS development, which shows that the distinctions made at level 1.6 are crucial in our chosen field of AI. A model at level 1.5 would not be as useful in this context. This does not entail that such models are explanatory, only that level 1.6 models are required.

Chapter Five: Conceptual Models, Competence Models and Psychological Explanation

We now have to show that conceptual models are psychologically explanatory of the expertise they model.

5.1 Competence Models and Psychological Explanation

We begin with a lemma: competence models are psychologically explanatory.

5.1.1 Models of Problem-Solving

A competence model is defined as a decomposition of a problem domain. A problem domain is a triple containing a set of problems, a set of solutions and a solution relation. A competence model is built around a decomposition of the solution relation into a set of inferential relations and a set of inferential types, each of which is a set of inferential primitives. The model is completed by the addition of a competence theory, which specifies the relationships between the inferential relations and the solution relation.

5.1.2 Psychological Competence Models

We can put further conditions on admissible competence models: the decompositions should be predictive; competence must be related to but not restricted to performance; competence models must rationalize performance; they must be developed within information processing contexts; they must describe the production of performance; and they must describe the set of admissible outputs, not attempt to determine output. Competence models that meet these conditions are defined as psychological competence models.

5.1.3 The Explanatoriness of Psychological Competence Models

We show that the six conditions on good explanations as given in §3.3 apply to psychological competence models. As explanations, psychological competence models answer such questions as *why did X do that?* and *how did X accomplish that?* Various audiences might be interested in such explanations, particularly ones which are interested in information processing accounts of problem-solving. These models are explanatory at level 1.6 by virtue of the decomposition into inferential relations, inferential types and inferential primitives, and, if the inferential relations are suitably chosen, can be explanatory at level 2.

5.2 Conceptual Models and Competence Models

The argument is completed by showing that conceptual models are psychological competence models.

5.2.1 Modelling Methodologies and the Alternatives

Our results will only apply to systems developed using models. Systems built using other methodologies such as rapid prototyping will not be covered by our result. However, a number of systems are built using model-based development methodologies, and there are a number of advantages to building systems that way.

5.2.2 Conceptual Models and Competence Models

Conceptual models are psychological competence models. First of all, they are competence models. The inferential primitives of the conceptual model are contained in the domain layer. The inferential types and inferential relations of the conceptual model are located at the inference layer. The competence theory is contained in the task and strategic layers. Conceptual models also meet the requirements on psychological competence models that were set out in §5.1.2. The interested audience for conceptual models as explanations is, of course, the same audience that is interested in psychological competence models. This includes KBS developers.

5.2.3 Off-the-Peg Models

There is a worry that only custom-built conceptual models can truly be assimilated to psychological competence models. However, some conceptual models are taken from model libraries. This is shown not to be a problem, in at least a large number of cases. By examining the KADS interpretation model library, the Generic Task library and the GDM library, we found that either static models could be altered in domain-dependent ways to customize them, or that decompositional modelling techniques were able to model a much larger variety of problem-solving than static techniques.

5.2.4 The Argument Completed: Conceptual Models, Knowledge-Based Systems and Psychological Explanation

We are now able to complete the main arguments of the thesis. 1) Since conceptual models are psychological competence models, and since psychological competence models are explanatory of problem-solving, conceptual models are explanatory of problem-solving. 2) If a KBS is developed from an explanatory conceptual model by model refinement, then to the extent that it retains the structure and ontology of the original conceptual model, it too is a psychological competence model of the expertise, and is therefore explanatory of the expertise.

5.3 Knowledge-Based Systems, Normative and Causal Explanation

The final task is to decide which type of explanation KBSs provide.

5.3.1 Normative Explanation?

A conceptual model provides a normative explanation of a practice, in the sense that it provides a manual for how to perform the task in an optimal way.

5.3.2 Causal Explanation?

It is difficult to see the items in a conceptual model corresponding to causally efficacious properties in the brain. However, for an explanation to be causal, the properties singled out have only to be causally relevant, not efficacious. Hence it is reasonable to suggest that conceptual models are causal explanations, because

their properties are causally relevant by programming for causally efficacious properties.

5.3.3 Conceptual Models and KBSs are Both Normatively Explanatory and Causally Explanatory

Hence we get the interesting result that conceptual models (and KBSs) are explanatory in the *two* ways suggested above. However, no firm consensus has emerged with respect to causal explanation in psychology, so this result must be more tentative than the main results of §5.2.

Chapter Six: Conclusion: Expertise and Artificial Intelligence

Finally, we review our conclusions.

6.1 Review of Chapters One-Five

We begin with a reiteration of our results about: bottom up philosophy, KBS development methodologies, scientific explanation, computational explanation in psychology, and psychological competence models.

6.2 Artificial Intelligence as Explanatory

We have given an existence proof of the question in the thesis subtitle: there are artificially intelligent systems (KBSs) which are explanatory of mental processes (the operation of expertise).

6.3 Expertise in Context

Expertise is a highly context-sensitive faculty. Conceptual models must include a great deal of contextual information: what they model is a system containing expert and world. Hence we must be clear that what is explained by conceptual models is the operation of expertise in a context.

6.4 Normative and Causal Explanations

On the basis of the fact that expertise can be explained, using the same model, either causally or normatively as is required, we make the tentative suggestion

that such coincidence of explanatory types might be characteristic of expertise: expertise can be defined as that psychological faculty which can be explained at once normatively and causally.

Chapter One: A Metathesis Concerning the Philosophy of AI

"The devil take you!" said Don Quixote. "What a peasant you are, and yet what apt things you say at times! One would almost think you had been to school."
"But I swear I can't read," replied Sancho.
Cervantes *Don Quixote*

1.1 The Importance of Being Artificial

Robert Sokolowski once asked an interesting question about artificial intelligence (AI): is intelligence like flowers or light (Sokolowski 1988)? In particular, when it is fabricated, does it merely simulate the 'natural' phenomenon, or is it an example of the phenomenon itself? Artificial light certainly is light, but on the other hand, artificial flowers are not themselves flowers.

One can approach this question in two ways. In seeking to determine whether an artificial X is an X, one can either look at the X, or at the artificial X. The philosophy of AI has tended to take the first approach, and its practitioners have tended to examine the properties of the X (intelligence), judging the artificial X on its ability to 'measure up' to standards discerned in the natural X. There is nothing inherently wrong with this approach, of course. If interesting and defining properties of the X can be discerned, then the question can be settled merely by proceeding to an examination of the artificial X to see whether it possesses those properties. The problem in the case of intelligence is that no clear set of such properties has been isolated and agreed upon. The viewpoints of Dreyfus and McCarthy, Searle and Pylyshyn, Nagel and Dennett seem irreconcilable, even after a couple of decades of discussion.

This suggests a failure of the 'top down' approach in this field. Obviously, one can't always expect philosophy to issue determinate answers to relatively nebulous questions such as this, but one would expect something like progress to be made. However the three major survey anthologies of papers on the topic (Anderson 1964; Haugeland 1981; Boden 1990a) have not shown any evidence of the debate's having moved particularly far.

The way I have described the impasse clearly suggests that I am going to favour the alternative 'bottom up' approach to this question about AI. But what does that entail? The idea is that the philosopher who takes this road will examine the properties of the artificial X, and seek to pronounce his judgment on that basis. Obviously, there is less of a clear cut distinction here than my description suggests. The top down philosopher needs to examine the artificial X before he judges it; the bottom up philosopher needs to examine the natural X at least in some respects if he wishes to discuss the artificial X in the context of Xs in general. But note that the properties of the debate will subtly alter as a result of the change in perspective from top down to bottom up. The bottom up philosopher will examine how the artificial X operates in its context; what are minor problems for the top down philosopher may well turn out to be major stumbling blocks from the bottom up point of view. This, of course, will tend to mean a shift in focus from the traditional interests of the philosopher to the rather more arcane technical literature of the artificial X. In the case of intelligence, the bottom up philosopher will turn from close readings of Descartes and de la Mettrie, Dennett and Searle, to examination of the less well-known results of such scientists as Lenat, Clancey, Newell, Rumelhart and McClelland.

The result of this switch should be beneficial; it should mean that philosophy could provide an extra influence on the work in this area, in the way that jurisprudence can influence the law or that the philosophy of quantum mechanics has influenced theoretical physics. And on the other hand, traditional philosophical questions could be illuminated by the new light coming from the field of AI. Obviously, this new input will need some interpretation; because AI is not discussed explicitly from the point of view of the philosophy of mind, the process of discovery of the relationship between AI and positions in current philosophy of mind is not so simple. But then this may not be a bad thing; the relationship between AI and the philosophy of mind is not so clear either. Any philosopher who makes relatively quick and easy identifications between positions in the two contexts is likely to be oversimplifying somewhere along the line.

Top down philosophy of AI tends to initiate discussion from the premisses of various thought experiments. Suppose we came across a room in which someone blindly followed explicit instructions on cards, such that his output was a perfect conversation in Chinese. Suppose someone made a molecule-for-molecule replica of you. Suppose we knew the state of Einstein's brain at the moment of

his death, and had sufficient theory of neurophysiological cause and effect to extrapolate forward. All these thought experiments, of course, have the potential to be informative, and crystallize important intuitions about the mind and its relation to the brain. But it is difficult to move from an appreciation of the issues raised in these thought experiments to issues raised in day to day AI research. We do not have the technology to build molecule-for-molecule replicas — and even if we did, we probably wouldn't. The production of a series of instructions for carrying out Chinese conversations would involve a colossal amount of work. The neuroscience involved to produce a conversation with the deceased Einstein's brain is so far beyond present capability as to be unimaginable in detail.

The result of the unrealistic nature of these philosophical sojourns into science fiction is that it is difficult to determine exactly how these thought experiments should impress us. It is a legitimate activity to hold the rest of the world constant while hypothetically musing on what might have been; however, so much of the world — in practical terms at least — must be altered to accommodate the science fiction scenarios, that what follows from what in those contexts is not at all obvious (witness the very large scale of disagreement here).

Far better, I would like to claim, to base philosophical discussion firmly on the bedrock of current practice. In the context of the philosophy of AI, this involves a firm grasp of the nature of research in AI. This, of course, does not imply the paternalistic view that only scientists are qualified to talk philosophically about science; it does, however, imply that one should have a practical account of the technology under discussion ready at hand. One should know what one is talking about. Interestingly, it also implies that 'purely' technological debate may well have important philosophical consequences.

This leads us to the most basic point about technology, which is that the benefit of a machine comes from its *complementing* human activities, not in its *imitating* them. A man can dig with his hands, but a shovel, though analogous to a hand in a number of ways, differs crucially: it is much harder and more resistant to stress; it is made of metal; it is a flat, curved plane, with nothing analogous to human fingers. A shovel might be an 'extra hand', but to judge its performance by standards set by the human hand would be ridiculous. A shovel can do one thing well — dig — and in virtually every other context is a less impressive performer than the human hand. So much is obvious. However, there are a

number of examples of philosophical argumentation taking a line that seems to require that a computer do *everything* as well — or as badly — as the human mind or the human brain. Since AI systems are very rarely created with that intention, that means that the argument has been effectively prejudged from the outset. Instead, the ideal relationship between AI and philosophy ought to be a symbiotic one, like that between Don Quixote and Sancho Panza; AI as Sancho should keep Quixotic philosophy's feet on the floor, while philosophy should provide a lead with respect to the interesting issues AI should be exploring. The result may well be a discipline moving onward in as comic a way as the original Quixote and Sancho, but the rate of progress should be increased. Be that as it may, to indicate some of the areas in which mismatch between 'pure' philosophical discussion and the realities of ordinary AI endeavour has already arisen, we shall briefly examine three examples of philosophical debate and its (in)applicability to AI.[1] Then, bearing these examples in mind, we will discuss how AI should be defined. This will complete our introductory task.

1.2 Three Examples of Philosophical Debate About AI

1.2.1 Example: The Turing Test

Our first example concerns an old chestnut: the Turing test (Turing 1950). The Turing test is well enough known for us to gloss over the details; the idea is that one important characteristic of intelligence is that the performance produced by application of the intelligence is indistinguishable from comparable human performance. But when the properties of machines are seriously considered, it turns out that the Turing test is going to give counterintuitive answers. It will tend to suggest that some performance that might normally be regarded as the product of intelligence is unintelligent, because too dissimilar to comparable human performance, and, conversely, that all sufficiently clever mimicry is the product of intelligence, whatever the intuitions might say. Now, the mere existence of contrary intuitions should not, by itself, be decisive, and certainly there are philosophers who take the supposed intuitions about the mind much too seriously. Even so, a test for intelligence should remain at least roughly congruent to general 'folk' concepts. And it it important to note that this is not because of a failure of the setup of the test itself — no tightening of the

[1] It has recently been argued that a similar mismatch appears between business practice and the field of business ethics. See (Stark 1993).

regulations will do the trick. The problem is that the test tends to ignore factors resulting from facts about the operation of machinery.

One point we should note here immediately is that Turing himself is very clear that his test is a *sufficient* but not a *necessary* condition for intelligence. Hence, strictly, failure to pass the test would not entail the unintelligence of the machine, though some writers do take the passing of the test to be both necessary and sufficient for intelligence (Harnad 1991). Even so, there are not many tests for, or general definitions of, intelligence which have gained wide approval. Therefore failure to pass the Turing test could reasonably be taken as fairly strong evidence that the machine is unintelligent (Hauser 1993c). The converse problem, when an unintelligent machine passes the test, is arguably more serious — if passing the Turing test is a sufficient condition, we get the unwelcome result immediately.

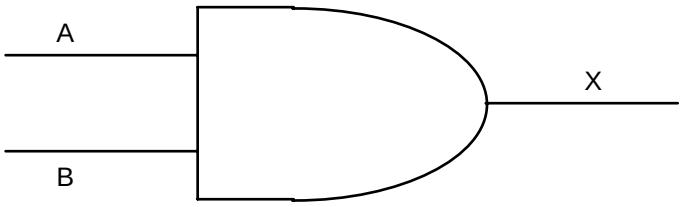
There are, then, two cases to consider where the Turing test simply 'gets it wrong'. The first case is the most often noticed, presumably because of prior prejudice against intelligent machines — the case where the test will pronounce a system intelligent, where it appears that the output on the basis of which the pronouncement was made was produced using totally mechanical means. We can easily demonstrate this with an example, that of arithmetic.

The powers of arithmetic of computers and calculators are well-known. They get the right answer in the blink of an eye. Furthermore, they make interesting errors which many a human would make — sometimes they even give up the ghost and output an error. The performance is arguably only trivially different from the human performance (except in terms of speed — and not always even then). In fact, one can imagine circumstances in which the performance is literally identical. The test judge asks the human to add 32 and 57; the human picks up the calculator and presses the following buttons in sequence: '3', '2', '+', '5', '7', '='. Now the judge 'asks' the calculator to add 32 and 57 — but asking the calculator to do this simply *is* pressing the sequence '3', '2', '+', '5', '7', '='. Hence it is arguable that the test should claim that, as far as calculation goes at least, such systems are intelligent.

But surely the computer's method of calculation is of interest here; the arithmetic unit of a computer is hard-wired in. Let us take a moment to consider how such calculations are actually performed. Computers can be regarded as being made

up of a small number of primitive elements at the *digital logic level* (Tanenbaum 1984). In a digital circuit, only two possible values are allowed. Usually in a computer's digital circuits, a voltage between 0 and 1 volt represents one of the values (a binary 0), and a voltage between 2 and 5 volts represents the other (a binary 1). These values can be manipulated by *gates*, arrangements of transistors which usually take one or two inputs and give an output that (when the output's voltage is interpreted as one of the binary signals) is a function of the input(s). Tanenbaum (1984, pp.59-61) discusses the way that gates actually work. Some gates and their 'truth tables' are given in Figures 1.1-1.3 (note that the different shapes of the gates are conventional representations, and not of any physical significance).

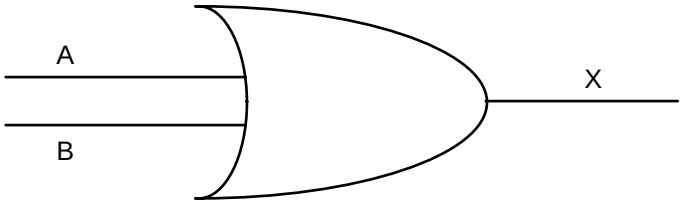
So, for example, in Figure 1.1, if the voltages of both A and B can be interpreted as binary 1s (i.e. their voltages are both between 2 and 5 volts), then the voltage of X will also be interpretable as binary 1. Otherwise, X's voltage will be in the range of the binary 0 (with the proviso that it is indeterminate what will happen if the voltage of either A or B is between 1 and 2 volts, or greater than 5 volts). Hence the gate is an AND gate.



AND

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

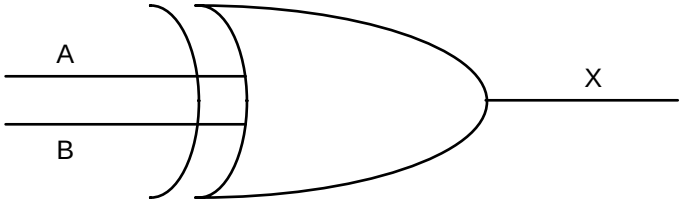
Figure 1.1: An AND Gate



OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Figure 1.2: An OR Gate



XOR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Figure 1.3: An XOR Gate (Computes the Exclusive-Or Function)

These gates can then be combined to produce circuits that can perform arithmetic operations. Figures 1.4 and 1.5 show the circuits that enable the computer to do addition.

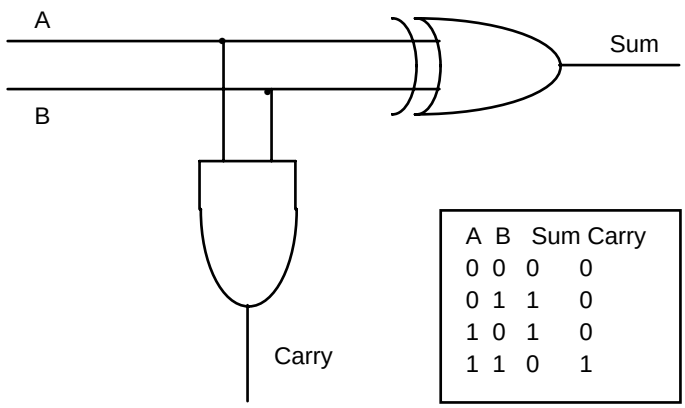


Figure 1.4: A Half-Adder

An array of these circuits can perform binary addition. For example, if one wished to add two 16-bit numbers (i.e. numbers between 0 and 65,535), one would have the half-adder (Figure 1.4) representing the rightmost (unit) digit, and fifteen versions of the full adder (Figure 1.5) each representing one of the other fifteen. A smaller example will give the idea of how these adders work. Suppose we wish to add 011 and 101 (two 3-bit numbers). We need one half-adder and two full adders. The rightmost (half-) adder takes as input the two unit digits of our numbers (i.e. 1 and 1). For the *sum*, 1 and 1 both pass into an XOR gate, and therefore 0 is output. For the *carry*, 1 and 1 pass through an AND gate, which therefore outputs 1. This circuit is linked up to the next (full) adder (which will deal with the middle digits) by the *carry* output of the half-adder which becomes the *carry in* input of the full adder. Hence we have, going into the middle adder a 1 (from the first number), and a 0 (from the second number) as our A and B, and a 1 as *carry in*. To compute the *sum*, the A and B (1 and 0) pass through an XOR gate, which outputs 1. This output becomes one of the inputs to another XOR gate, the other input being the *carry in* (1). Hence the *sum* output is 1 XOR 1, which is 0. For the *carry out*, A and B are passed through an AND gate, to give 0 (1 AND 0 = 0). Next, the output of the first XOR gate (which computed A XOR B), 1, is passed through another AND gate, together with the *carry in*, 1, to give 1. The outputs of these two AND gates then pass through an OR gate, which then outputs the *carry out*, which is 1 (0 OR 1 = 1). This *carry out* becomes the *carry in* of the leftmost circuit, together with the digits 1 and 0, to give a *sum* of 0 and a *carry out* of 1. This *carry out* becomes the fourth digit (since we have no other inputs). Hence, our 3-bit adder has taken as input 011 and 101, and output 1000. Converting binary to decimal, that is input 3 and 5, and output 8. The arrangement of adders, inputs and outputs is given in Figure 1.6. This type of addition circuit is called a *ripple-carry adder* — the carries

have to 'ripple' from right to left along the array of adders before the sum can be completed (faster adders that avoid this delay can also be built (Blaauw 1976)).

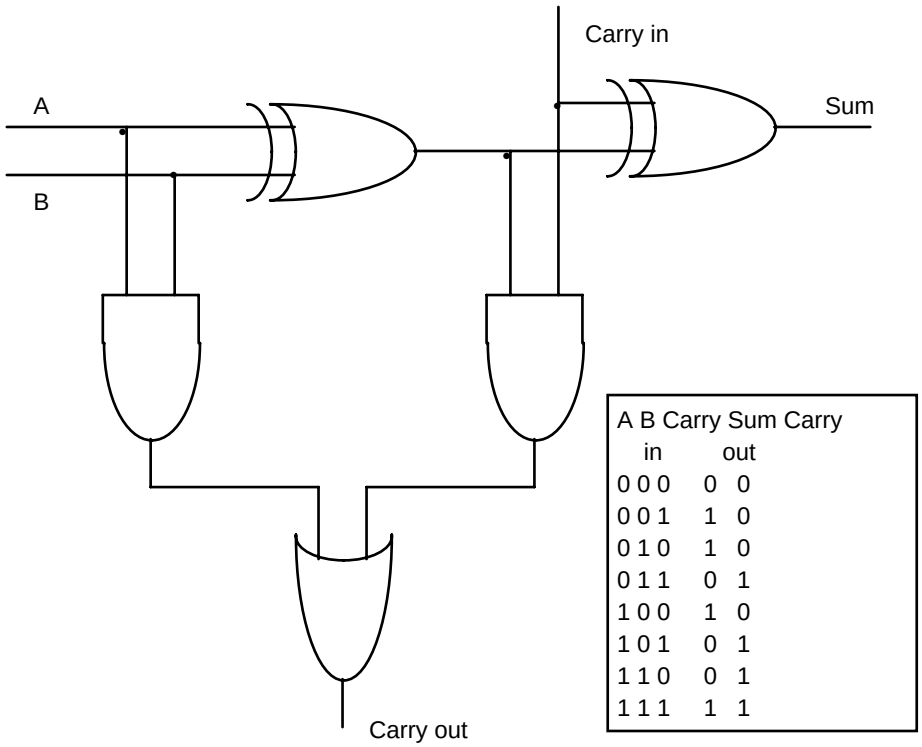


Figure 1.5: A Full Adder

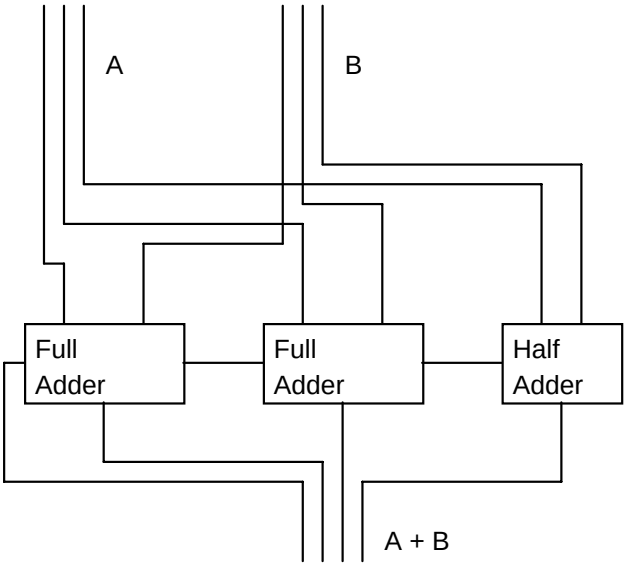


Figure 1.6: A 3-Digit Ripple-Carry Adder

There does seem to be a consensus (but see Hauser (1993ab) as an example of an opposing view) that artefacts which use such mechanical methods should not

have intelligence ascribed to them — even though their output can be easily interpreted as addition. Hence, few would claim that the passing of the Turing test with respect to arithmetic should be taken as the only evidence that intelligence is 'in use'. Of course, I am appealing to intuitions that a ripple-carry adder itself is not a mechanism that is likely to fall under any definition of intelligence (although it might be the case that a ripple-carry adder could be a *component* of some intelligent system).

But there is another case where the Turing test misses the mark, and this is the less visible problem — the case where the test fails to ascribe intelligence to systems that are at least arguably intelligent. Or, rather, the test has a built-in bias against artificial systems. Being artefacts, AI systems are built for a reason. Some machines are built or programmed as 'pure research', of course. However, most research in the field is driven at least partially by commercial considerations. The point here is that a commercial system — for example, a knowledge-based system such as we will be discussing in this thesis, or a face or voice recognition system, or a theorem prover, or whatever — is intended to perform some task, within some corporation or organization, where its operation is commercially viable. But this means that the machine must have some differential advantage over a human who can perform some comparable task. This advantage might just be cost, of course; it will simply be cheaper in some cases to use a machine to do some task than it would be if a human were used. An example of this might be a face recognition unit used as a 24-hour security system where it is cheaper to run the system than to pay the guard. However, in other cases, the advantages will be other than simply the gain of equivalent performance for lesser cost — the commercial differential will stem, at least in some cases, from a *difference* in performance. So the artificial face recognition system might beat the security guard for a number of performance-related reasons such as: the artificial system cannot be bribed; is less likely to bow to apparent authority (e.g. someone disguised as a general); cannot be threatened; can memorize 4,000 faces in an hour. This result is not surprising, of course; artificial systems are built and used to complement human abilities, not to mimic them. But it does follow that the Turing test will fail to be a reasonable yardstick for intelligence (if AI systems are to be allowed to compete here at all) because the Turing test legislates *against* anything that *complements* human performance, and *rewards* mimicry. This, in effect, is French's charge (1990). He has a problem with the Turing test's assumed definition of intelligence; he notes, rightly, that the definition cannot be *general* because of the importance of the

conversational abilities required by any system that passes the full version of the test as described by Turing. We are interested in a weaker charge, that the test actually rules out virtually all existing and foreseeable pieces of software, because such software will, in the nature of the case, not only fail to reproduce such conversational abilities, but in practice will deliberately eschew such skills.

This result actually is generalizable. Any test which purports to be a simple test for intelligence such as the Turing test, or a component of such a test, should at least be developed with an eye to the context in which software development takes place. Three brief examples show that this is not always the case. Some thinkers have supposed that reproducibility is an important factor. But some people are unable to reproduce; furthermore, a reproducing machine would not really be very useful. Even so, some software is able to reproduce itself — this is how computer viruses work, but no-one would suggest that that ability adds anything to the *intelligence* of the viruses. Other philosophers (Harnad 1990) have talked of the *Total Turing Test*, a version of the Turing test where the machine has to move around in the world and interact with it. This ignores the fact that virtually no machine is developed to do that, save in an abstract way (but cf Brooks 1991). Most AI systems are designed to fill computational niches in organizations and corporations, but this is the extent of their situatedness (O'Hara and Shadbolt forthcoming discuss the importance for psychology of this type of contextualization). Still other commentators (e.g. the connectionists) talk about graceful degradation, the ability to produce decent performance, albeit at a decreasing level, in unfamiliar situations. Again, this ignores the obvious fact that most AI systems are intended for a single task, and that graceful degradation in those circumstances has no utility at all. In fact, in many applications, graceful degradation is a negative attribute. It is often important not to use computers in circumstances where their output is not entirely reliable, whereas the temptation to use a computer in inappropriate circumstances may be large when the performance only degrades partially.

We have discussed two cases where the Turing test arguably fails to produce the right results about intelligence. It is plausible that in each case, a *different* notion of Turing test is in operation. In the first case (where a calculator is pronounced intelligent), the test is really dependent on IO (input-output) relations, with perhaps some small set of performance-related criteria (e.g. reaction times) as supplementary. In the second case, however, emphasis is laid on characteristics that are arguably essentially *human*, which we have no reason *a priori* to assume

are characteristic of intelligence generally. Not only that, but study of the current state of technology will show that such characteristics are generally ignored by designers of 'intelligent' systems, for the reasons outlined above. Hence, whichever version of the Turing test you buy, only one of these criticisms applies to it. Nevertheless, our main point survives: in each case, reflection on the nature of the computers (and the use of computers) leads us to suppose that highly counterintuitive results will follow from the use of whichever version of the Turing test is chosen (and therefore that informed reflection on the actual real-life *use* of computers should lead us to reject the Turing test as a test for intelligence).

1.2.2 Example: The Problem of Induction

Our second example of the inappropriateness of the top down approach with respect to the philosophy of AI concerns induction. The problem of induction is well known, hinted at by Locke, and stated explicitly by Hume. Many scientific/empirical generalizations of the form $\forall x.Fx$ are based on a finite quantity of, typically observational, experiments. Yet of course, we cannot *deduce* the universally quantified law from the conjunction of the finite quantity of instances (unless — the rare case — we have independent grounds for supposing that we have observed all possible instances). Repeated observation is not even *allowed* as a basis for the assertion of the generalization in many sciences (e.g. mathematics, where Goldbach's conjecture, for instance, remains a conjecture, not a theorem).

Investigation of the surrounding issues in philosophy has tended toward the justification of so-called *ampliative* induction (i.e. induction that attempts to establish truths that extrapolate beyond the data). So, for instance, philosophical questions may be asked about the type of evidence that is required to establish an inductive hypothesis; Bacon, Mill and Keynes argued in favour of a type of ampliative induction known as *variative* induction, which claims that the inductive hypothesis is better supported if the instances cited in its support are drawn from a relatively wide variety of types, while the opposing position, *enumerative* induction, championed by Carnap, suggests that mere multiplicity of supporting instances is the relevant factor. An orthogonal position, *Bayesianism*, stresses the subjective gradation of a reasoner's belief, with the result that the variative/enumerative controversy can be sidestepped. A second set of questions concerns the possibility of application of Pascalian numerical

techniques to the analysis of induction. A third set of questions concerns the attack on Humean scepticism (and related paradoxes such as the paradox of the ravens, the 'grue' paradox and the lottery paradox).[1]

All this is, of course, very important, and philosophically and logically deep. But note how this work all stems from the work of Bacon, Pascal and Hume. The concentration on the philosophical literature produces a corresponding concentration on the problems that absorbed the great philosophers (Bacon in particular would not have approved of that!); the development of rules for the manipulation of evidence takes place, not against the background of actual scientific practice, but instead as a means of minimizing the departure from the purity of truth-preserving manipulation of premisses in a logical argument.

However, induction is also a technological problem in AI. The field of *machine learning* (ML) is an area of research into the possibility of getting machines to discover inductive generalizations. Sets of observations are described in some canonical way (the *case data*), and general rules are suggested and tested by the machine. Clearly, all three traditional sets of questions are extremely relevant to this practice. Each rule induced by the system will be associated with a measure of confidence or certainty in the generalization; if variative induction is required, then some measure of variety of the case data will have to feed into the calculation of the confidence (if not, some measure of the number of cases will suffice). Most ML systems are numerically based, so the exploration of Pascalian probability analysis is clearly relevant. The various inductive paradoxes are problematic for the output of ML systems as well as for the output of more familiar inductive practices.[2] Yet study of the actual practice of ML

---

[1]The straightforward scepticism trades on the logical possibility that one could build up a massive body of favourable evidence for a hypothesis, while being unable to rule out the logical possibility of one countervailing instance, which would, of course, falsify the universally quantified hypothesis. The paradox of the ravens (Hempel 1945) concerns the relative usefulness of positive evidence. Intuitively, it should be the case that any evidence which is entailed by the hypothesis should support the hypothesis. But then the observation of a white handkerchief should support the hypothesis "All ravens are black", by virtue of its support for the logically equivalent hypothesis "All non-white things are non-ravens"; this seems counterintuitive. The 'grue' paradox (Goodman 1954) depends on generalizing from noticing that all observations of green things are observations of grue things (where something is grue iff it is green until the year 2100 and blue thereafter), and therefore any hypotheses involving the notion of greenness are supported to precisely the same degree as the analogous hypotheses with 'grue' substituted for one or more occurrences of 'green'. Yet we do not take grue-like predicates seriously in our actual inductive practice. The lottery paradox (Kyburg 1965) trades on the idea that it seems to follow that, since each lottery ticket is practically certain not to win the raffle, no lottery ticket will win the raffle.

[2]Though in this case it should be noted that, as with most inductive practices, one can ignore such paradoxes and just concentrate on inducing rules that work.

reveals a further serious problem that is addressed much less often by the philosophical community, the problem of *hypothesis generation*. What the ML system must do is to search for possible rules in a space of possibilities that will grow as the case data are described more elaborately. For example, suppose the algorithm works by inducing over a range of cases. If each case consists of an object, with, say, the values of m associated attributes, then in the case of simple rules (i.e. rules which relate single atomic propositions as antecedent and consequent) concerning binary attributes only (i.e. attributes with two possible values), there will be 2m(m - 1) possible relationships to check. So, to give a small example, suppose that the domain of the cases is football teams, and the data have three binary attributes: *successful?*, *well supported?* and *good to watch?*. Then the ML system will have to search for twelve possible rules:

All successful teams are well supported.
All unsuccessful teams are well supported.
All successful teams are good to watch.
All unsuccessful teams are good to watch.
All teams that are good to watch are well supported.
All teams that are not good to watch are well supported.
All well supported teams are successful.
All well supported teams are unsuccessful.
All teams that are good to watch are successful.
All teams that are good to watch are unsuccessful
All well supported teams are good to watch.
Well supported teams are never good to watch.

It can easily be seen that the complexity will increase if conjunctions are allowed in the antecedent, or disjunctions in the consequent. But the main problem here is the restriction to binary valued attributes. An ML system might possibly get away with disallowing conjunctive antecedents and disjunctive consequents, but ultimately will have to support multi-valued attributes.

The point here is that there is an 'unknown problem' of induction. The justification of induction is a serious question; but, for ML purposes, more serious is the problem of *selecting hypotheses to test from the vast space of possible hypotheses*. This space must somehow be cut down, so that inductive hypotheses are selected with some kind of prior probability of ampliative success. The greater seriousness of the latter problem in ML stems, of course, from a prosaic technological problem; if the system doesn't run, its output cannot be tested, and so therefore if the system takes arbitrarily large amounts of time to process detailed cases, the pressing problem is not the justification or validation of the output, but the production of output *in real time*. This is not a problem that philosophers have often treated — indeed, philosophers have rarely considered the problems of real time operation. As Cohen (1989) puts it:

> Bacon remarked that knowledge is not communicated in the same order as that in which it was discovered. But he drew no sharp distinction between the concept of induction as a process of research or gradual discovery and the concept of induction as a pattern of proof or of graded justification, though the former is concerned with temporal sequences of events and the latter with timeless relations between propositions. Mill had approached this distinction by 1851, when he claimed, in the third edition of his *System of Logic*, that even if his inductive methods were not methods of discovery they were at least methods of proof. And since Mill's day most writers on induction have concentrated on the logic of inductive proof rather than the methodology of inductive discovery. (Perhaps the only systematic exceptions to this tendency today are computer scientists writing about the construction of expert systems.)
> (Cohen 1989, p.11)

When we talk about knowledge acquisition for expert systems later in the thesis, we shall see how a main problem with the evaluation of expert testimony is that most expert testimony is intended implicitly to be justificatory of the output, rather than a description of the actual process of discovery. Interestingly, a similar problem can be seen in the philosophy of science, where, for example, a theory like Popper's falsificationism, though it claims to have methodological repercussions, is mainly concerned with 'what follows from what', i.e. Cohen's timeless relations between propositions. Fortunately, in the philosophy of science, the problem is less pressing, since scientists are relatively good at suggesting useful hypotheses. ML systems, in contrast, need to have that ability programmed into them.

This, of course, is a serious problem for ML. However, it also turns out to be a problem for the philosophers of justification. Standardly, Bernoulli's law of large numbers is used to show that, as more instances are observed, the formula learned will be more probably true (Cohen 1989, pp.21-2). However, Bernoulli's law only applies to a given formula; in other words, if there is some independent way to secure a hypothesis in which we have confidence, then there is no problem. But this is not the standard position in ML, and, worryingly, it is at least possible that situations will arise in other contexts where no given inductive hypothesis is available. Bergadano (1993) shows that as the set of possible inductive hypotheses grows, the accuracy of the estimates of the probability of

those hypotheses will diminish. Although an increase in the number of examples will tend to increase accuracy, it is also true that as the language for describing the domain gets richer (which will automatically increase the number of *possible* hypotheses), the confidence we can have in our judgments will decrease. Of course, in some circumstances this will not matter too much. For instance, if the set of training examples contains a large proportion of all possible examples, then increasing the number of possible hypotheses will probably increase accuracy (as there will be more of a chance of selecting an applicable hypothesis). However, since the usual situation is that there are infinitely many possible examples not contained in the training set, this is little comfort.

The problems that Bergadano explores in the analysis of induction therefore turn out to be relevant to the philosophical discussions to which we have alluded. Indeed, the relevance of this ML problem is greater even than the above argument suggests — Bergadano (1993, pp.39-47) explores the possible solutions to this new inductive problem from within various standard philosophies of inductive inference (for the record, according to Bergadano, Carnap's inductive logic framework seems more promising than both the classical probability approach of uniform convergence and subjective Bayesianism). Hence it might be the case that this ML problem could help adjudicate between various rival accounts of induction.

So, whatever the details of the disputes over induction might be, we can say that by the analysis of the requirements of a particular functioning technology, ML, we discover (a) that the standard philosophical accounts of induction, while clearly relevant for ML, do not fully describe the inductive problem as it presents itself in ML; (b) that in fact the ML problem is arguably a special case of a ubiquitous problem of general philosophical interest; and (c) that it turns out that the ML problem can be seen as a method of judging between a number of competing philosophical accounts. Again the bottom up approach to philosophy proves useful in ways unavailable to the top down approach.

1.2.3 Example: Narrow versus Wide Content

Finally in this section, we can briefly consider a third example: the ongoing debate about 'narrow' and 'wide' content, which has been taken to impinge on debates in the philosophy of AI. The narrow/wide content debate stems from Hilary Putnam's Twin-Earth thought experiment (1975), the gist of which is that

on a molecule-for-molecule replica of Earth, where the only difference is that for each molecule of H_2O on Earth, on Twin-Earth there is a molecule of a complex compound XYZ which is macroscopically indistinguishable from H_2O. Now consider a man M who thinks "I would like a glass of water." Then on Twin-Earth there is another man, Twin-M, who also thinks "I would like a glass of water." There is no relevant physical difference between the two men. However, M is having a thought that refers to H_2O, whereas Twin-M is having a thought that is related to XYZ. Hence, the argument goes, the two men, who by hypothesis have no relevant physical differences between them, must be thinking different thoughts, from which it follows that the physical facts about M and Twin-M alone are not sufficient to establish the contents of their thoughts.

The argument is very controversial, and we needn't evaluate it here. What we are concerned with is the further argument that the Twin-Earth speculations are an important negative indicator for AI; that AI must somehow, if it is to develop as a discipline of psychological importance, defuse that argument. This argument, which can perhaps be seen most clearly in some of the papers in (Pettit and McDowell 1986), goes roughly along the following lines:

> What the Twin-Earth argument, and related arguments, show is that the content of at least some psychological states is not dependent, as might intuitively be thought, solely upon other states of the subject. The content of these states is essentially partially dependent on environmental factors. Therefore, the exact content of what I think is not necessarily up to me.
>
> Now, what gets postulated in AI is a view, roughly, that psychological states can be explained by the development of computer programs which, all being well, have the same IO relations as the human psychological faculties that they are intended to model, and satisfy other evidential standards as well (whatever they may be — see Chapter Four). There will be a supposed correspondence between the internal states of the machine, described at some level, and the psychological states which are the explananda of the endeavour.
>
> However, what the Putnam argument shows is that the psychological states can only be specified semantically in terms that are *broad* or world-involving. Unfortunately, computational states are fully

> specifiable using internal criteria: the environmental circumstances are irrelevant to the description of a computer. Hence computational states cannot fully model psychological states.

This argument can be attacked at several levels. At a basic level, the Twin-Earth argument depends on the intuition that meaning determines truth conditions — the idea being that two propositions, one of which is true and another false in identical circumstances, are necessarily non-identical. This is certainly a plausible line to take, but it *is* a suppressed premiss of the argument. At a wider focus, one could argue, for various reasons, that the Twin-Earth argument is simply unconvincing. Jerry Fodor (1987) attempts something of the sort; Tyler Burge (1986) also implicitly recognises the force of this sort of complaint by his attempts to produce a similar argument to Putnam's, a 'tighter' argument less vulnerable to 'irrelevant' criticisms. Alternatively, one could deny that the Twin-Earth argument is relevant to AI after all, by denying the premiss that computational states have to be described 'narrowly' (Wilson 1994). At a wider focus still, one could argue, with Andy Clark (1989), that even if the Twin-Earth argument goes through unimpeded, the relevance to cognitive science and AI is at best limited, since the relationship between mental or psychological states, and the 'inner' causes of those states, is doubly mediated, both by relations to the external world — as Putnam claims — and by a holistic method of ascription of psychological states which entails that one ascribes a *group* of psychological states to a subject on the basis of observed *series* of behaviours. These are all top-down arguments. As part of our bottom-up practice, we can add further evidence by considering the nature of cognitive science and AI, and considering what action it would be practical to take in the face of these arguments.

This additional evidence would also be relevant to another, related, debate, which can be seen as coming from what, in some sense, is the opposite (though still top down) direction. The *eliminativist* programme of research, led by the Churchlands (1979, 1986), claims that standard 'folk psychological' terms cannot be the basis of a truly scientific psychology, and that any scientific psychology would have to be based on discernible structures/processes in the brain. The folk psychological terms, revealed as having been referring to nothing (systematically), would then cease to be employed, rather in the way that terms such as 'witch' or 'phlogiston' have ceased to be employed, or else would undergo radical alteration, rather as the term 'humour' has. The standard responses to eliminativism have been to deny that folk psychology is intended as a theory,

even a proto-scientific one (e.g. Wilkes 1991), and to refuse to accept the post-elimination discourse as distinctively *psychological* (McCulloch 1986). Again, these are, I think, perfectly reasonable counter-arguments to the eliminativist claims. But in this section what I want to show is that such eliminativist arguments run counter to much of the spirit of AI. This, of course, need not be fatal to the arguments themselves; what I do want to establish, however, is how destructive such arguments can be when applied to the domain of AI.

The destructiveness comes because the eliminativist claims also lead fairly naturally from the psychological/philosophical claim, to a methodological claim. The methodological claim is quite straightforward: if psychology should be based around facts about the structure of the brain, then a machine intended for psychological modelling should be structurally similar. In other words, the current use of computers with von Neumann architectures (the standard form of computer architecture, not at all brainlike) should be stopped (for AI purposes), and instead some sort of computation based on neuronal structures should be used.

So, we have two arguments that supposedly cause trouble for AI: the Twin-Earth argument and its variants, and the eliminativist argument. They are motivated by different emphases, but have an analogous effect when applied to AI. The Twin-Earth argument claims that computational states, since they are specified without reference to the environment, are ruled out of the psychological court; the eliminativist argument says that (folk) psychological states, since they are apparently unrelated to the computational states in the brain, are ruled out of the scientific court.

The methodological variant of these arguments is to say that what AI should be doing is modelling the underlying computational structures of the brain in some way. So, the methodological variant of the Twin-Earth argument says that if a computer is to have psychological states, for example, then what should be attempted in AI is the production of a rich repertoire of behaviour using methods analogous to those used by the brain, since the brain has genuine causal powers which can be captured internally, 'narrowly'. Hence when the computer is transferred from context to Twin-context (or when we compare the states of a computer on Earth with those of its twin), its psychological states will vary in the same way as a human's. The position then would be roughly analogous to McGinn's (1982) view that internal states determine the narrow component of

content, and the environment completes the determination; intelligent computers' states would have the same bipartite content. The job of the computer scientist would be to look after the narrow content; the broad content would be provided by the context. Hence computer scientists shouldn't try to program computers with the whole content; they should instead seek to represent narrow content in their programming languages.

Similarly, the methodological variant of the eliminativist argument says that AI should not be modelling folk psychological entities, with their relations of rationality, etc., but instead should be focussing on modelling the logical structure of the brain. In each case, the claim is made that AI should cease to focus on the symbolic areas upon which it has traditionally focussed.

The obvious beneficiary of these views, of course, in the AI field, is the field of connectionism (Rumelhart et al 1986, McClelland et al 1986). We will not go into connectionism in any detail here, suffice it to say that a connectionist system is made up of 'nodes', each of which is a very small processor capable of rudimentary computation, connected in a directed graph by 'connections'. The connections carry signals to the nodes, which then act as inputs to 'activation functions', one of which is associated with each node. The activation function determines, on the input, whether the current node will send a signal or not to the other nodes with which it is connected. The association with the brain is clear: nodes are analogous to neurons, connections to synapses, and the activation functions model the readiness of a neuron to fire, given suitable signals from the neurons connected to it. The analogy with the narrow content debate is that the states of the nodes and their connections are like narrow content (invariant across contexts), while broad content is provided by the (context-sensitive) interpretations of the states of the output nodes. On the basis of these analogies, the methodological variants of both the Twin-Earth and eliminativist arguments often recommend the use of connectionism as a standard methodology in AI.[1]

Nevertheless, it is a big step to argue that a major industry/discipline such as AI should cleave to a single methodology on the basis of purely philosophical arguments. What I wish to suggest in the remainder of this section is that,

[1]Though not all. Brooks (1991) wants to develop robots which move about in the world without necessarily doing any representation of data at all - because this is more physiologically realistic. Edelman (1992) agrees with the eliminativists that the brain should be modelled; he, however, denies that the brain does any computing at all.

methodologically, the idea is a non-starter, on the basis of a brief examination of current AI practice.

The first point to note is that connectionism, despite the many attractive qualities of that methodology, will not do the trick. The details of this point need not be laboured here. All we need to note is that the architecture of this methodology fails to model the brain sufficiently well. For example, the brain consists of some 10^{11} neurons, each with approximately 1,000 meaningful connections with other neurons, making for a total complexity of the system of some 10^{14} connections. Connectionist systems, on the other hand, rarely number their connections even in the millions. Hence the difference in scale is sufficiently great to cause concern. As another example, the brain has relevant features that it is difficult for connectionist systems to model, structural features such as synapse-on-synapse connections, and features relevant to the characterization of the computation of the brain, such as the flooding of contiguous neurons with neurotransmitters, thereby affecting neurons that are close together in space (such contiguity is not necessarily associated with the patterns of connectivity of the brain). Conversely, many important techniques associated with connectionist computing have no obvious neural analogue (e.g. back propagation).

Now, connectionist systems have a number of interesting features, and are able to perform tasks in real time that standard symbolic systems are unable to perform, so it should not be deduced from this that connectionism should be abandoned as an AI methodology. However, it does seem to follow that, if the argument is that only a brain-like computational architecture will do for AI, more work needs to be done to establish that connectionism is such an architecture; at present, both connectionist systems and symbolic systems should equally be rejected on these principles. There is no gain to be made by insisting that a search begin for computational techniques that can operate on a genuinely brainlike architecture, either; the interesting systems that have explored this path still have very primitive computational abilities (Scutt 1994).

The second point to note is that the big advantage of symbolic systems is that they can be manipulated using languages that are sufficiently like natural languages to make the following of their semantics tractable. For instance, take the ability to program computers to do arithmetic. The semantics of the computational output are not isomorphic to the abstract structure that we normally call arithmetic. If we call the addition operator of the machine 'H', and

the addition operator over the real numbers '+', then we can see that there are large numbers of cases where H and + coincide. Obvious examples are simple sums: $2 + 2 = 4$ and $2 \text{ H } 2 = 4$. However, there are important differences. A computer is a finite machine, and so it can only represent finitely many numbers. Hence its data class *real* is finite, and will contain no transcendental numbers (it will, for example, contain approximations, to, say, 15 decimal places, of such numbers as π and e). Also, because the computer reals are finite in number, there is a largest and a smallest; this, of course, is not true of the real numbers. Furthermore, a computer can only represent numbers to some specified number of significant figures. Hence some problems will receive the wrong answer on that account: for example

1,000,000,000 + 0.000,000,001 = 1,000,000,000.000,000,001

but

1,000,000,000 H 0.000,000,001 = 1,000,000,000 (a so-called 'rounding error)

Finally, there is one symbol in the semantics of computer arithmetic that has no analogue in the real numbers, and that is $\perp$ (pronounced 'bottom'). $\perp$ is the result of any problem that is impossible to compute, either because the program goes into an infinite loop, or because the result is impossible to represent. So, for example, if L is the largest number that can be represented by a computer, $L \text{ H } 1 = \perp$ (whereas in ordinary arithmetic, $L + 1$ is the successor of L). Another example: 1/0 is undefined in ordinary arithmetic, while $1/0 = \perp$ for the machine.

The purpose of this digression is to show that, although all computers will depart from the desired arithmetic performance in some respects, the fact that we can give computer arithmetic a semantics is helpful. And we can generalize this result to any symbolic manipulations performed by machines. The ability to give a semantics to computational systems is useful in two ways. Firstly, of course, the semantics can be checked to make sure that the machine will give the output that is desired. This task, of course, gets increasingly difficult as the program gets complex, but the roughly familiar structure of most symbolic programming languages means that the task is generally not impossible (Gries 1981). And the second way in which having a semantics is useful is that the points where performance departs from the ideal can be monitored; in our discussion of

computer arithmetic, we could see where the computer would simply fail to get the right answers (at least beyond some rigorous degree of approximation). This, as can be imagined, is very useful indeed. However, whereas symbolic systems have a relatively simple semantics, the same is not true of connectionist systems. Virtually the only way to work out what even a moderately complex system will do it to run it on some data. Of course, the advantage of this is that, in many cases, novel solutions can be found for problems (often problems which seem intractable symbolically). But the flip side is that, if we wish to program a computer to do some task, if we can program it symbolically, we know exactly how close to the desired performance we will get, whereas setting up a connectionist network may well result in good performance in tests with no guarantee that that standard of performance will continue (or no indication of which areas are likely to produce unhappy results).

The third point that we should make against those who insist on a monolithic approach to AI and cognitive science is that there is a deep assumption being made by those who say that only structures sufficiently brainlike can be truly explanatory psychologically. We shall discuss psychological explanation in much more detail in Chapter Four; for now we just need to note that, for a number of purposes, it might well be the case that a symbolic program could provide an account at an interesting level of detail. It may not be 'the whole story', but to say that a standard symbolic AI program can *never* be of psychological interest seems extremely premature.

1.3 The Philosophy of AI

These preceding three examples are intended to show how a study of the practices of AI can lead to some philosophical insights. To reiterate, this is not to say that the philosophical discussions are wrong, or misguided, only that interesting input to the debates can be found in a discipline which has always had philosophical roots. The first example, of the Turing test, shows the relevance of the fact that it is essential that an AI system work; if it did not, then, no matter how philosophically kosher it was with respect to some doctrine, it would be completely useless. Because AI systems have to work, philosophical corners inevitably have to be cut. But this does not invalidate such systems philosophically; on the contrary, usually the result is a demonstration of the practical import (or otherwise) of the philosophical doctrine. Only by studying what computer scientists build can we see that the Turing test has a tendency to

rule out AI systems; AI systems, since they are intended to *complement* human intelligence, will always fail a test for intelligence that is built around *duplicating* human intelligence. This suggests that the Turing test cannot be a general (i.e. neutral on the question of machine intelligence) test of intelligence (as Turing himself understood). Our second example, a look at machine learning, shows that the *a priori* considerations characteristic of philosophy can sometimes overlook important practical realities. The attempt to build machines which can automatically induce indicates an undiscovered problem of induction; the problem of hypothesis generation is the 'real' problem, while the problem of justification can, to an extent, take care of itself. Thirdly, our example of narrow content shows that some philosophical distinctions, useful in fictional settings for the purpose of thought experiments, need not translate particularly well into methodological considerations.

In the remainder of this chapter, we shall briefly consider some deeper, more foundational problems than have been discussed above. In particular, since our top-level argument in this thesis is the value of AI to philosophy, we need a characterization of AI, which shall be the business of this section. In the final section, we shall introduce the philosophical problems to be tackled in the remainder of the thesis.

1.3.1 AI and Intentionality

So, what might be a first shot at a definition of AI? It has been described as a collection of various projects which attempt to use computers to produce output that, if produced by a human, would be assumed to have required intelligence. This is a useful short characterization, but not an adequate one for two reasons. Being closely related to the Turing test definition of intelligence, it will fall foul of the two arguments produced in §1.2.1 above. Arithmetic calculations would certainly require intelligence if performed by a human, yet the simple hard-wired arithmetic circuits described in §1.2.1 stretch even sympathetic intuitions a long way. And conversely, much AI deals with rather boring and mundane (to an unreflecting human) abilities; as Margaret Boden puts it:

> [AI] is sometimes described as the study of how to build and/or program computers to enable them to do the sorts of things that minds can do. Some of these things are commonly regarded as requiring intelligence: offering a medical diagnosis and/or prescription, giving

> legal or scientific advice, proving theorems in logic or mathematics. Others are not, because they can be done by all normal adults irrespective of educational background (and sometimes by non-human animals too), and typically involve no conscious control: seeing things in sunlight and shadows, finding a path through cluttered terrain, fitting pegs into holes, speaking one's native tongue, and using one's common sense.
> (Boden 1990b, p.1)

Note the jarring (though not contradictory) nature of the two arguments against this first characterization. The first seems to rule out the effects of simple hard-wired computer circuits; the second seems to rule in the effects of what we might call hard-wired capabilities in humans, those over which we have no conscious control. This clash is, I think, indicative of the many problems that exist in the characterization of intelligence, and similar relevant notions in the philosophy of mind and AI, such as intentionality, cognition, symbol grounding, etc. etc..

What we would like to do is to discuss AI without entering into this area of confusion, both quagmire *and* minefield! In particular, there is no need to enter into questions of which faculties are 'caused' or 'premised upon' intentionality. Intentionality is the notion of being 'directed towards' events or things in the world. We uncontroversially 'have' intentionality, since we uncontroversially can think 'about' birds, bees or whatever. Arguments can ensue about dogs and cats; do they think 'about' things? There appears to be a majority in favour of at least a qualified 'yes' to that question. No such consensus can be determined about machines.

But the existence of the notion of intentionality does seem to some to offer a nice short cut to the establishment of certain philosophical truths about machines and minds. If we knew whether or not machines 'had' intentionality, then we could settle many of the philosophical questions that our developing technology seems to be posing. Hence all we need to do is look for intentionality, right?

Wrong! In what sense do computers do arithmetic? In the sense that they produce output that can be *interpreted* as arithmetic. This notion of being interpreted as arithmetic is crucial here. In fact, humans do arithmetic in the same sense — we do arithmetic because of the way that our output can be interpreted as arithmetic. Certain mistakes are admissible — forgetting to carry a

1, for example — while others — apparently random guessing, say — are not. And this is a result we should expect. Intentionality, the sense of being 'directed towards', is a semantic notion, as is interpretation. But this should lead us to suspect that intentionality is not at all a *property* of a system, in the way that, say, height is a property of a man. Intentionality, being a semantic notion, is at least in part imposed from without. The question should now become: under what circumstances should we regard a system's output to be directed towards some external thing? But note that this is not necessarily going to be decidable simply by examining the system itself. In particular, the question may at least partly rest upon the way in which we make fresh decisions in the fresh situations envisaged by the thought experimenters. In this context, we can see that the novelty of Searle's classic 'Minds, Brains and Programs' (1980) is not its discovery that syntax does not determine semantics, but in its assumption that this is *news*.

1.3.2 Two Views of AI

So, let us ignore the question of the relationship between human and animal or artificial intelligence, conceived of as a question of determining which systems' outputs have semantics and which don't (intrinsically), since phrased this way, the question is practically impossible to answer, though easy to prejudge. If we look instead at current AI practice, what we will see will be two discernibly different forms of AI, which we might term the official version and the unofficial version. The official version is promoted by Margaret Boden, seeing AI

> ... as *the science of intelligence in general* — or, more accurately, as the intellectual core of cognitive science. As such, its goal is to provide a systematic theory that can explain (and perhaps enable us to replicate) both the general categories of intentionality and the diverse psychological capacities grounded in them. It must encompass not only the psychology of terrestrial creatures, but the entire range of possible minds. It must tell us whether intelligence can be embodied only in systems whose basic architecture is brainlike (involving parallel-processing within networks of associated cells), or whether it can be implemented in some other manner. And, 'computers' having dropped

> out of the definition, their especial relevance to such a science must be proven.
> (Boden 1990b, p.1)

This, though it may be a unified project in its own right, cuts across the field of AI as it sees itself. For example, most philosophy of AI would fall under this rubric, of course. But I wish to maintain that the philosophy of AI is philosophy, not AI (as the philosophy of logic is not logic, and aesthetics is not art). On the other hand, the type of AI system that I intend to focus on in this thesis, the *knowledge-based system*, would probably fall outside the definition. Most knowledge-based systems add little to current knowledge, or theory, of intelligence in general, tending instead to be used to solve hard problems in small, well structured (typically terrestrial) domains.[1]

Another problem with the official definition is that it does, as Boden says, write computers out of it. Whereas AI, however it is conceived, is certainly a science to do with the application of computers to problems. Even those approaches that, for example, look toward the brain as the embodier of the important properties that lead to human intelligence (McCulloch and Pitts 1965; Churchland 1986; Rumelhart et al 1986; McClelland et al 1986; Clark 1989) abstract the brain's *computational* properties from its many others. Searle's point against the 'many mansions reply' to his Chinese Room experiment (1980, pp.80-1), that 'the interest of the original claim made on behalf of artificial intelligence is that it was a precise, well-defined thesis: mental processes are computational processes over formally defined elements', is well made.

On the other hand, there is the unofficial view, which is based more on a census of what AI researchers actually *do*. This view will maintain, pragmatically, that AI is basically a loose collection of research projects, the general focus of which is problem-solving using computers. An added dimension to this is that certain computational *methods* have proved useful in these projects; further work goes on to examine the possibilities of using these methods in novel situations. On this view, philosophical interest in AI should focus on the significance of these

[1]However, it could be the case that even KBSs could be viewed as small experimental, empirical studies into the ability of artificial systems (specifically: computer programs) to produce the sort of 'intelligent' output that we are interested in. But even if they are so viewed, this is a very small role to play. If we are to take current practice seriously, we should note that a definition which entails that KBSs have such a small role belies the fact that they are easily the most common locus of AI research and development.

pieces of research, conceived both individually, and as part of a global AI research programme.

However, few philosophers have bothered to investigate particular research projects. On the contrary, the philosophy of AI all too often focuses on either science fiction thought experiments such as the Chinese Room, or other, more low-level, discussions of the neo-Wittgensteinian kind, designed to bring out the 'logical grammar' of mind, thereby discovering the philosophical significance of AI without necessarily having to know anything about it! In other words, the starting point of most philosophical discussion of AI is the work of other philosophers. Nothing wrong with that, of course, except that people in AI who wish to see what *they* have done in philosophical (as opposed to computational or managerial) terms are usually obliged to do that philosophical research as well. There is nothing particularly wrong with this either, except that the opportunity to cross-fertilize in the two disciplines is lost.

I wish to contribute to the redirection of the philosophy of AI towards the problems that are set by AI *as actually performed*, as opposed to problems to do with, for example, perfect artificial replicas of people. These problems are just too remote from those confronted by AI in the unofficial sense. When practitioners of AI meet a philosophical problem, it is usually much more specific — for example, someone may wish to discover what the epistemological or psychological status of a new taxonomy of supposedly generic problem-solving 'types' is (Chandrasekaran and Johnson 1993). For such practitioners, the question is: which philosophical theories/arguments will be valuable in that restricted context? What is certain is that no-one who did not know about that restricted context would be able to answer the question.

For example, take a typical question in the philosophy of AI: Can machines think? I don't want to worry now about the meaningfulness of the question, particularly. What I want to highlight is the way that it is impossible for AI in the unofficial sense to provide any input to that question. If I went out tomorrow and built a machine that was up to the standard suggested in the various thought experiments or science fiction films, then that would still have no effect on the philosophy of mind.[1] The *existence* of the machine used in the various thought experiments is conceived to be irrelevant to those thought experiments. Those

[1]Conceptually at least. Technology often changes cultures (and therefore the nature and focus of philosophical arguments) in dramatic ways, and I certainly think that a machine of that capability would have an effect of the greatest magnitude.

that wish to show that machines might be able to think tend to have an argument of this sort:

> If a machine had property P then it would be able to think (for reasons r, s, ...);
>
> It is possible that a machine might have property P;
>
> Hence it is possible that a machine could think.

On the other hand, those who wish to show that machines couldn't think tend to have a counterargument of this sort:

> (Even) if a machine had property P, it would not be able to think (for reasons t, u, ...);
>
> Allowing the possibility of a machine's having property P is maximally generous to the 'machines-can-think' brigade;
>
> Hence it is indeed the case that it is impossible that machines could think.

Either way, my building a machine with property P does not affect the argument.

But, as we have outlined above, there is always the bottom up way of approaching the philosophy of AI. If one approaches from the top down, then one tends to get argumentation that never intersects with the concerns of AI in the unofficial sense, that is simply too abstract to have any effect in the field of AI itself. This cannot be right — the philosophy of X should, one hopes, have (beneficial) effects on X. The alternative approach is to enter the philosophy of AI by way of AI. We should be examining developments in unofficial AI, and pronouncing on the philosophical importance of these developments. We would not be dealing, at least initially, with such highly abstract questions as whether, in principle, machines could ever think. Instead we would focus on such questions as: *if a machine had property P, could it produce moderately complex output in real time?* or *how would, or could, property P be instantiated in a machine?* We would, by this process, be discovering a little more about what the abstract philosophical questions could mean, on the one hand, and on the other, we would be discovering a little more about our technological world as it stands.

1.4 Example: Modelling Expertise

In the light of that, in the remainder of this thesis I would like to examine the contribution that unofficial (= as it is practiced) AI can make to Boden's project of official AI (i.e. the study of intelligence). In particular, I should like to look at the ways in which technologically and commercially based AI modelling theory can contribute towards psychological explanation. I shall take the notion of a knowledge-based system, a type of AI system which attempts to mimic or simulate expertise in various more or less well-structured contexts, and try to show how some recent developments in knowledge-based system development methodologies are in fact more informative about the psychology of expertise than many (including some workers in the field) have supposed. I introduce knowledge-based system development theory in the following chapter. Chapters Three and Four set out a philosophy of explanation; Chapter Three discusses scientific explanation in general, while Chapter Four is concerned with the particular case of psychological explanation. Chapter Five will bring together all the threads for the main argument.

Chapter Two: Of MYCIN Men: Knowledge-Based System Development Methodologies

> "What about the secret weapon? From the way you behaved when you took off to intercept the Jap formation I gathered you had an idea about it."
>
> "I've more than an idea," answered Biggles. "I know what it is. Only it isn't a weapon. I'd call it a trick."
>
> Capt. W.E. Johns *Biggles in the Orient*

2.1 What Knowledge-Based Systems Are

Knowledge-based systems (KBS or *expert systems*) are computer systems that attempt to store large amounts of data (or *knowledge*[1]) about a particular domain, organized in such a way as to enable them to solve problems and make inferences in that domain.[2] As such, a KBS can be divided into two separate components: the *knowledge base* (KB), which contains the pieces of knowledge that the system uses, organized and represented in a *knowledge representation language*; and the *inference engine*, which is the part of the program which manipulates the pieces of knowledge in order to produce new pieces of knowledge. The person responsible for the construction of a KBS is called a *knowledge engineer*, and the process of KBS development is often called *knowledge engineering*.

This section will introduce the ideas in and behind KBS technology, to provide background for the reader who is not versed in the field. In our opening quotation, Biggles has just discovered how the enemy are bringing down the Allied aircraft. Once the mechanism was uncovered, its simplicity led Biggles to

---

[1]In the field of knowledge-based systems, the term *knowledge* is not used as it is in philosophy. In particular, that p is a piece of knowledge does *not* imply that p is true! For example, a KBS may contain a rule 'if p then q'. This may then be treated as a material conditional, even though it may be consistent to assert 'p and not q'. The justification for this is that 'p and not q' may be true only in very rare or extreme conditions, and so little harm is likely to be done to the safety of the KBS by the inclusion of 'if p then q'. As another example, a KBS might suggest a proposition p as a hypothesis (e.g. in a medical system, p might be 'x has pneumonia'). Even though the hypothesis is not being asserted, it still counts as a piece of knowledge in this sense.

For an overview of the issues surrounding the use and organization of knowledge in AI see (Reichgelt 1991), and, for more technical discussion, see (Ramsay 1988) and especially (Genesereth and Nilsson 1987). Throughout this thesis, I shall use the term 'knowledge' in the AI sense, as opposed to the philosophical sense, unless otherwise flagged.

[2]For an historical review of KBS research see (Shadbolt 1989). For discussions on the techniques used in KBSs (and their theoretical foundations) see (Jackson 1986; Giarratano and Riley 1989).

denounce it as a trick. The operation of KBSs is so apparently simple that many (e.g. Dreyfus et al 1986) are drawn to a similar denunciation. Here's how the trick is done.

2.1.1 Brief Historical Notes[1]

Expert systems first appeared as a distinct field of AI research round about the late seventies, after AI itself had been around for perhaps twenty years. Which project produced the first KBS is, as is usually the case with technological innovation, a matter of some dispute. Contenders include MYCIN (Shortliffe et al 1973; Shortliffe 1976), a system to assist doctors in the selection of appropriate courses of treatment for patients with bacteremia, meningitis and cystitis, and which is still highly influential, DENDRAL (Buchanan and Feigenbaum 1978), which inferred the molecular structure of various substances on the basis of the results of mass spectrometry, and MACSYMA (Martin and Fateman 1971), which selects techniques of analysis for mathematical problems (and which is still in widespread use today).

By the 1980s, KBSs were becoming commercially viable, as opposed to being large, academically-based systems, and their output was being acted upon, rather than being tested against actually prevailing conditions and decisions. In 1987, Alan Bundy was able to remark that

> the U.K. expert system community has been very successful in the development of small scale, commercial, rule-based, expert systems. A typical example is a fault diagnosis system for a piece of specialised hardware, consisting of a set of less than 100 rules, running on a P.C., in one of the many commercial shells. Part of the success consists in the unexpected (to me anyway) discovery of a large number of commercially interesting problems which yield to such a simple mechanism.
> (Bundy 1987, p.3)

while in the States, a year earlier,

[1]This section is deeply indebted to (Shadbolt 1989). For a history with special reference to the KADS, Generic Task, and GDM methodologies, see (O'Hara and Shadbolt forthcoming).

> we are beginning to see the development of systems designed ... for sale as finished products, at times allowing an organization to move into a new line of business. ...
>
> We're going to see many "mundane" applications. Despite the image of AI as high technology, it may be that the most commercially significant applications are hardly the stuff of science fiction. Campbell's Soup, for example, was faced recently with the imminent retirement of someone with 44 years of experience in running one of its soup production lines. Rather than lose that body of skill, they worked with the expert to capture what he knew about that specific task and embodied it in a rule-based system. Running a soup cooker well is far from dazzling high technology, but it does save significant time and money. I think we're going to see many more of these apparently mundane but in fact quite important applications.
> (Davis 1987, pp.27-8)

And now KBS technology is firmly established commercially, performing mundane tasks as predicted.[1]

A KBS contains a great deal of knowledge about a small domain (and virtually nothing else[2]). When that knowledge is manipulated successfully, then the problem-solving behaviour of an expert in that domain can be mimicked by the machine. Compared to the experts, KBSs do surprisingly well (though see Dreyfus and Dreyfus for the opposite view (Dreyfus et al 1986; Dreyfus 1987)[3]). Obviously, in other domains (even related domains), a KBS will be worse than useless (although the problem-solving *strategies* of the KBS may be reusable — see §2.2.2).

---

[1]They don't come any more mundane than controlling the activated sludge process in sewage management (Williams et al 1989)! *Research and Development in Expert Systems VI* contains a number of other examples of papers describing commercial expert systems; see in particular (Bolger et al 1989; Chierici et al 1989; Vadhwana 1989).

[2]Doug Lenat's CYC system is intended to provide a KB which will hold other sorts of 'common sense' knowledge for KBSs (Guha and Lenat 1990).

[3]Dreyfus and Dreyfus do not help their case, however, by concentrating on computer chess; the field of computer chess is not usually thought of as being in the realm of AI. Furthermore, after coming out in favour of neural nets in (Dreyfus 1987), they decided that neural net technology was just as hopeless as standard symbolic AI (Dreyfus and Dreyfus 1988). The suspicion is that they are opposed to any technology as it becomes feasible; they are subscribers to the doctrine that, in matters psychological, clarity is a sin, and the best psychologies are obfuscatory.

We should just make a terminological point here. The terms *knowledge-based system* and *expert system* are used more or less interchangeably. However, to call a system an 'expert system' does imply that the system is equivalent to, or can do the job of, an expert in the domain, in some sense. In actual practice, though, the performance of such a system is often not up to that standard, and, more to the point, the envisaged role of such a system is usually much less ambitious. The term 'KBS', on the other hand, conveys the governing principle of such a system, which is that the inferential power of the system is based not so much on the techniques used to perform inferences, but on the large amount of knowledge stored in the KB. The term gives no false hints about performance and role; in fact, such systems are often used as assistants, or teaching aids, embedded in human and corporate contexts, rather than as decision makers or substitute experts. Hence we shall use the term KBS from now on.

Having established what KBSs are historically, in the rest of §2.1 we shall look at some of the relevant issues in KBS development. We begin by looking at the representation of knowledge in the system. Then we will look at some requirements of standard KBSs; the requirement that the system be verified, and the requirement that the system explain its own output. Finally, we look at how knowledge is acquired for KBSs bearing in mind these requirements.

2.1.2 Knowledge Representation

Knowledge Representation is the study if ways of storing and organizing knowledge in a computer for use in a KBS; representation is usually done through the medium of a *knowledge representation language* (KRL). In this section, we discuss two views of representation (two *types* of KRL) only; logic-based representation, and frame-based representation. This ignores the field of connectionist, or PDP systems which put forward an alternative view of information storage. The use of connectionist principles in KBSs is a burgeoning area of research at the moment (Lacher 1993), but in this thesis we restrict our attention to more mainstream KBS paradigms.

Logic-based knowledge representation languages (KRL) are what one would expect — languages that are based around logic. Hence, the language will have something like the expressive power of the language of first order logic.[1] Then,

[1]And indeed of other logics than FOPC. There are a number of systems that use the languages of modal logic for, e.g. temporal reasoning, or of dynamic logic, or equational logic.

for example, the expert knowledge can be encoded as a number of if ... then ... rules, and the data upon which the KBS is to make its inferences can be matched against and unified with[1] the antecedents of those conditionals — when they match, the consequents of the conditionals can be concluded. This mode of inference is called *forward chaining*. Alternatively, if the KBS needs some particular piece of information, that piece of information can act as a goal to be proven, and be matched against the consequents of the conditionals in the KB; the antecedents of the conditionals that match can then be set up as intermediate goals, and so on until a goal is reached which the system knows is satisfied. This mode of inference is called *backward chaining*.

Logical propositions are generally represented and stored independently of each other. However, inferential demands are often made in KBSs that need knowledge to be arranged in 'chunks' (Minsky 1975). One common type of chunk in use is the *frame*, and frame-based representation and reasoning is now well-established and widespread. Each frame has a number of *slots*, each of which is filled with the value of the parameter symbolized by the slot-name. Frames can be hierarchically ordered into structures, so that frames lower down the hierarchy can *inherit* values and slots from frames above. For example, we might have the following frame declaration. I have invented a little standard language for this declaration, where slot names are underlined and to the left of a colon, with the values given to the right of the colon.

```
frame: car
   no-of-wheels: 4
   size: small
   subframes: beetle ferrari
   superframes:

frame: beetle
   speed: slow
   nationality: germany
   subframes: my-beetle
   superframes: car

frame: ferrari
   speed: fast
   nationality: italy
   subframes:
   superframes: car
```

[1]Unification is the process of checking that the variables, constants, and function terms in a pair of wffs can be substituted consistently. For example, if we had a rule *FOR ALL X,Y,Z, IF P(X,Y) THEN S(X,Y,Z)*, and we knew that *P(0,1)*, then we say that *P(0,1) unifies* against the antecedent of the conditional by the substitution {X/0, Y/1}, and therefore that we can deduce *FOR ALL Z, S(0,1,Z)*. However, if the rule was *FOR ALL X,Y,Z, IF P(X,X) THEN S(X,Y,Z)*, then the antecedent of this would *not* unify against *P(0,1)*, because there is no substitution for *X* that would produce the latter proposition. Producing a unification algorithm for a computer is non-trivial.

```
frame: my-beetle
   colour: green
   year: 1974
   subframes:
   superframes: beetle
```

which would give us the structure shown in Figure 2.1. Then from this hierarchy, we can deduce about `my-beetle`, not only that it is green, but also, for example, that it is German, and that it has four wheels — it *inherits* this information from the hierarchy of superframes above it.

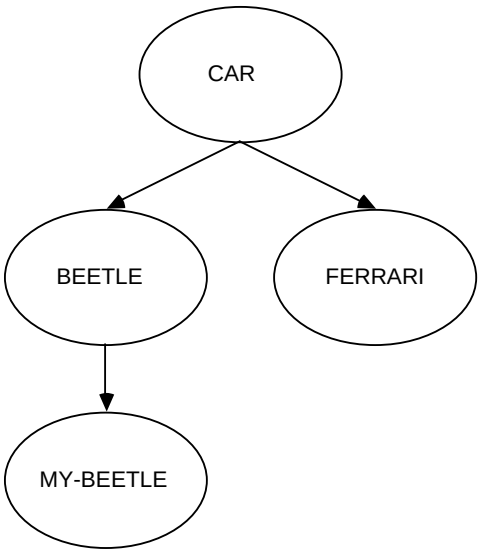


Figure 2.1: A Frame Hierarchy

So basically, we have two types of representation, and they correspond to two types of inference. Logic-based KRLs use logical inference; they try to *deduce* a desired conclusion from premisses. On the other hand, frame-based KRLs use *inheritance* as their mode of inference; here an individual is placed in a class from which it inherits properties that can be asserted of it.

Logic has two main advantages over frames. Firstly, logic has a semantics, whereas frames do not. That is to exaggerate the difference between them; since logic-based theorem provers generally have restrictions on how long they will search for a proof, some valid deductions cannot be performed. Nevertheless, the existence even of a partial semantics for logic means that one is relatively clear what can be deduced from what, and what the truth conditions of anything in the KB are. Whereas frame-based semantics are notoriously difficult to pin down. A famous, or infamous example, is the Nixon diamond, shown in Figure 2.2. The structure tells us that Republicans are bellicose, and that Quakers are not, but,

when faced with an object *Nixon*, which is both a Republican and a Quaker, and which therefore inherits the *bellicosity* slot, we do not know what to say. Because Nixon is a Republican, he should be bellicose, yet because he is a Quaker, he shouldn't. This type of problem is called an *inheritance conflict*. Of course, computers are machines whose behaviour is determinate, and hence in any *implementation* of this frame structure, there will be a determinate answer to inheritance conflicts such as Nixon's bellicosity; however, these answers will depend on irrelevant factors, such as which of the two frames *Republican* and *Quaker* was declared first, and whether new frame declarations override older ones. The frame structure itself will not provide the answers.

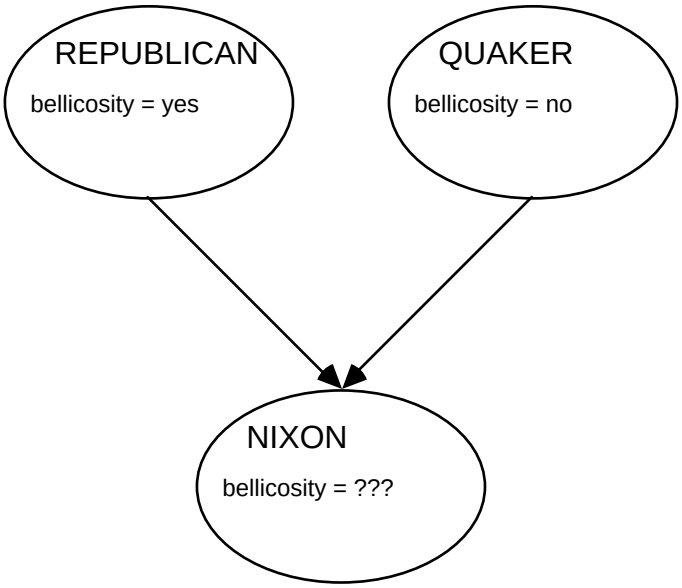


Figure 2.2: The Nixon Diamond[1]

The second advantage of logic is that logical languages are very expressive. Logics exist for a number of areas of discourse, for example, modal logic, temporal logic, epistemic logic (though the usual logic used by KRLs is first order predicate logic). And also logics can express information that is partial or incomplete, by using disjunctions and the existential quantifier. Frame-based languages have difficulty in expressing this sort of information. It is not clear, for example, how to express the information that *there is at least one Republican*, if there are no known examples of particular Republicans.

[1]Called the Nixon *Diamond* because the Republican and Quaker frames are each subframes of a general Object frame.

On the other hand, frame-based languages have two advantages over logic. Logical systems (theorem provers) are very inefficient and slow. Frame systems, on the other hand, have the effect of grouping related knowledge, which cuts down the search space for relevant information. If a logic system needed to discover the value of *X* in *(bellicosity Nixon X)*, it would have to search through the KB in order to match the form of the query against the propositions held in the KB, whereas the frame system can go straight to the *Nixon* frame and read the answer off, either from the frame itself, or from the frame hierarchy above it (assuming the inheritance conflict has some resolution). So frames can be seen as useful ways of organizing knowledge taking the *context* of that knowledge seriously.[1]

Furthermore, frames are better able to express the type of knowledge that is not logically valid, but enables us to make defeasible deductions. For example, given that x is a car, we can deduce that x has four wheels. But this is not logically true; it is not even accidentally true, since some cars have three. Hence the material implication symbol will not do, to express this sort of rule of thumb. If we did try to use material implication, we would end up with the following KB.

```
∀x.car(x) ∅ no-of-wheels(x) = 4
car(reliant-robin)
no-of-wheels(reliant-robin) = 3
```

This KB is inconsistent. However, a frame structure enables a local value to override the more general value (the *default* value). Hence, if we added the following two frames to our frame-based KB above

[1]Actually, logic-based languages and frame-based languages perform pretty much the same when the answer is given in the frame. Where frames come into their own is when inferences have to be made using inheritance. For example, suppose we wanted to know how many wheels *my-beetle* has. In the frame system given above, the inferences would go like this: look for *no-of-wheels* slot in *my-beetle*. There isn't one; look for *superframes* slot. Go to the superframe; look for *no-of-wheels* slot; there isn't one. Look for next superframe; look for *no-of-wheels* slot; the answer is 4. In an equivalent logic-based KB, the inference would go: look for any unquantified statements in KB that have *my-beetle* as a term; then look for those in which the predicate is *no-of-wheels*. There aren't any. Now go through all statements that assert that *my-beetle* is an instance of something. We find *beetle(my-beetle)*. Now go through all universally quantified statements with *beetle* as antecedent; then go though all the consequents of those statements looking for one with *no-of-wheels* in the consequent. There aren't any. Now go through all those consequents again. Each time there is a predicate in the consequent, look for all quantified statements which have that particular predicate in the antecedent; then look for *no-of-wheels* in the consequents of those statements. Hence eventually we find that all beetles are cars and then that all cars have four wheels (and hence that *my-beetle* has four wheels). But in a complex database, that could be a very tough search.

```
frame: reliant-robin
   no-of-wheels: 3
   subframes: not-my-car
   superframes: car

frame: not-my-car
   subframes:
   superframes: reliant-robin
```

and we wanted to know `(no-of-wheels not-my-car)`, the inference engine would first look for the `no-of-wheels` slot in `not-my-car`, fail to find it, move up the hierarchy to the next superframe, `reliant-robin`, where it would discover that the number of wheels of `not-my-car` was 3. If, however, we wanted to know `(no-of-wheels my-beetle)`, the inference engine would have had to go up as far as `car` to find the governing `no-of-wheels` slot, and would then give the answer that `my-beetle` has four wheels, the default value. Various logics of default reasoning have been proposed (Reiter 1980; McCarthy 1980), but none are entirely satisfactory. The reason is pretty obvious; the standard semantics of logic simply won't cover defeasible reasoning. After all, *for all x, if x is a car then x has four wheels* taken as a universally quantified material implication is simply false.

Current research into knowledge representation formats is beginning to indicate that a suitable compromise is to have a hybrid KRL that will include both logic and frames, and possibly special purpose modules to carry out specialized inferences. That way, the efficient frame-based module can be used to deal with whatever inferences it can deal with, and then pass anything that it can't deal with over to the theorem prover, thus retaining most of the advantages of each, while minimizing (though not eradicating) the effects of the disadvantages. One example, VITAL-KRL, is reported in (Domingue et al 1993).

2.1.3 Verification and Validation

KBSs are now used very frequently indeed, often in circumstances where lives or money are at risk. Hence, an important research issue is the *verification and validation* (V&V) of KBSs. Many techniques are used for V&V, often having been invented for a single application. Generic tools, approaches and languages for V&V are only now just being brought out (Benbasat and Dhaliwal 1989; Dhaliwal and Benbasat 1990).

Validation is the process of determining the degree to which some artefact actually performs its intended task (Nunnally 1967). Hence, in the case of a

KBS, there will be some requirements specification, and the validity of the KBS will be the degree of homomorphism between that KBS and the specification of the target system (Vandierendonck 1975; Adrion et al 1982). *Verification* is the demonstration of the consistency and completeness of the software, i.e. the fact (if it is a fact) that the KBS can compute all that it is supposed to compute, that it can't compute anything else (if it can, then memory is being wasted), and that what it computes is indeed correct (Adrion et al 1982). The difference between verification and validation is, intuitively, that verification shows that the program actually fulfills its internal requirements — i.e. that the software is computationally capable of performing the low-level tasks that are intended — while validation shows that the program as specified solves the customer's problem. If we imagine a project with a problem to be solved (written in English), a specification of a program (in some algebraic language such as Z) and a program (in some implementable programming language such as Lisp), then validation will show that anything conforming to the Z specification will solve the English problem, while verification shows that the Lisp program indeed conforms to the Z specification. A final process, *evaluation* is the process of determining whether or not the system is actually of any use. Obviously this is related to the outcomes of verification and validation, but also will involve investigating whether the KBS's user interface is any good, or whether the KBS will take forever to make its inferences, or whether the system explains its decisions sufficiently clearly, or whether the output is understandable by someone who is not computer literate (Gaschnig et al 1983; Gaines 1988).

Hence, when discussing KBSs, we should remember that these are the dimensions along which they are *meant* to be measured. The standard philosophical thought experiments of perfect robots á la *Blade Runner*, not even aware themselves that they are not human, or of molecule-for-molecule replicas, perhaps had better remain science fiction, as least until we have developed a way of measuring real systems such as KBSs against them. Obviously, a KBS will fall well short of the android in many things, but not (necessarily) in the one thing that it was designed to do — infer and communicate knowledge about a domain.

2.1.4 Explaining KBS Output

There is another way in which the knowledge a KBS contains has to be available. Not only does the structure of knowledge in the KBS have to be

available for the purposes of verifying and validating the system, but the decisions that the system makes must be open to scrutiny. After all, if the KBS outputs a decision that the patient's leg be sawn off, a properly prudent doctor would wish to know the reason why. The explanatory knowledge is important for the understanding of the system's decision, and is also useful in debugging.

The first serious explanation system was developed as part of the MYCIN programme (Davis 1982). The idea behind this development was to give a trace of problem-solving at the implementation level. In other words, if the system had fired rules R_1, R_2, ... R_n at runtime in drawing its conclusion, these rules would be presented to the user in a chain to show how the system reached the conclusions it did. Hence the question "why?" was interpreted automatically as "which rules need this datum, and what are their consequents?"

When a rule trace is given as an explanation, the result is that there is no particular difference between V&V and explanation — both will involve the production of an implementation level account of the problem-solving processes coded up in the machine, and the comparison of that implementational account with some sort of ideal or model account. But explanation and V&V have very different requirements. In V&V, the knowledge engineer has to make sure that the particular implementation of the system can meet the user's requirements, whereas what an explanation has to do is to justify the KBS's output for its intended audience. However, most of the uninitiated find KRLs extremely unfamiliar and unintuitive, and therefore when rule traces are presented (especially if this is done in the KRL directly) they find the output difficult to understand and correspondingly difficult to critique.

Hence, the explanation facilities of KBSs must go beyond rule traces. Swartout's Xplain system (1983) associates deeper justifications with the rules that it uses. In particular, underlying the top-level knowledge, *compiled knowledge* in the jargon, are two types of *deep knowledge*: a domain model, and 'domain principles', which are representations of problem-solving control strategies. Hence the content of each rule can be explained in terms of the deep domain model, while the domain principles can explain why the system behaved as it did.

For example, suppose a macroeconomic policy KBS outputs that a tax cut is required. Then it is reasonable to ask why that output is produced. The KBS

might then reproduce the rules it used to come to that decision. So, in this example, the KBS might say that a tax cut's preconditions are high inflation and trade deficits, and current conditions include those factors. This is an explanation of the first type, such as might be found in MYCIN. However, the user might not be interested in that sort of explanation. Although he will be interested in the trigger conditions for an assertion, he may feel that such an explanation is not very deep (and therefore that his scepticism remains with respect to the output), or that the policy is unsound in the particular conditions that obtain, or even that the apparent input-output relation is ill-founded. Hence he might want to ask why a tax cut is a good idea for shrinking trade deficits. But a system whose explanatory powers were only of the first type would be stymied by this question.

What should be possible for explanatory systems is that they can derive explanations from a deeper domain understanding. This was Swartout's aim in producing Xplain. So, in our small example, when asked why the tax cut is recommended for shrinking trade deficits, it might answer that tax cuts can be expected to encourage savings, stimulate investment, and increase production, which will then decrease prices, increase exports, make domestic goods relatively more attractive to foreigners, and thereby tend to reduce the trade deficit. With all this knowledge, it could have derived the tax cut policy directly (as we say, from 'first principles'). However, the computation is quicker if the short cut, from high inflation and trade deficit to tax cut, is simply given as a rule for the system. But even so, although the long form of the reasoning is slow and therefore inappropriate, it can be used to generate explanations for the system's user.

A third innovation in explanation came when William Clancey noted that KBSs generally perform tasks that are describable at a higher level of abstraction than the rule-based goal-subgoal level. In other words, the "why?" question which prompts justification of KBS output is often to be interpreted strategically, as opposed to the more nuts-and-bolts level at which MYCIN takes it. NEOMYCIN (Clancey 1983) works in the same domain as MYCIN, but represents its diagnostic task explicitly. Its operators include such items as *establish hypothesis* and *explore and refine*, and thanks to that extension of the expressive powers of MYCIN, it can provide explanations at that level of abstraction.

This means that the system can reason about its own goals and strategies, as well as about the relationships in the domain. So, to continue the tax cutting example, if the user of the system wanted to know, not just why a certain aim was being pursued, but what is the reasoning behind the adoption of the aims, he might ask why increased tariffs are not considered as a way of reducing trade deficits, and receive the answer that tariffs have associated with them political costs, and the strategy is to consider politically easier plans first.

Hence we note that, because of the importance of the requirement that KBSs give good explanations of their reasoning, KBSs will tend to have to contain much more knowledge than is required simply to produce reasonable output. Furthermore, KBSs will tend to have very deep knowledge encoded within them about the domain, and also about useful problem-solving strategies in that domain (Chandrasekaran et al 1989).

2.1.5 Knowledge Acquisition

And so we see that, for various reasons, what goes into a KBS might be more than would normally be expected. The knowledge representation languages and techniques that we discussed in §2.1.2 are contrived for the purpose of encoding information into a machine; hence they are not particularly natural formalisms for an expert to express her expertise. Furthermore, the requirements of V&V (§2.1.3) and explanation (§2.1.4) mean that a very complete picture of the domain, and of the way that the KBS will manipulate objects in the domain, must be elicited and put into the KB. Hence, it is not sufficient merely to interact with an expert, interrogate her, and fill the KBS with the elicited information. Getting information out of an expert and into a machine is a long and arduous process, called *knowledge acquisition* (KA). Indeed, the process can be so arduous that the *knowledge acquisition bottleneck* is often referred to, since the important business of coding up the KBS cannot get going properly until the knowledge that will fill the KB has been elicited. The problems that the KA process have to overcome include the following.

• The representations that are adequate for the machine (e.g. logic-based languages, frame-based languages) are usually not familiar to an expert. For example, when an expert uses an 'if ... then ...' construction, it is not usually material implication that she is using. More frequently, she won't use implication-based modes of expression (which are ideal for KBSs) at all. Hence

she generally cannot express her expertise in a particularly congenial way. With the help of the knowledge engineer, she has to 'translate' her expertise into a machine-friendly language. The problem *there* is that the translation itself has to be verified, which is no easy task, since there is no-one with the *overarching* expertise in both the domain itself and knowledge engineering to do that.

• Experts are generally quite poor at explaining their own problem-solving. This is not too surprising; experts need to explain their results largely for the purpose of justification, for example in a court of law. Hence they need to show that hypotheses are useful and fruitful in their explanations; this usually involves a fair bit of *post hoc* rationalization, cutting out the dead ends and useless hypotheses generated along the way, which, of course, are completely uninteresting for any justificatory explanation. However, for the purposes of knowledge engineering, the process of hypothesis generation is itself highly interesting and important, so the knowledge engineer needs to know which hypotheses get generated and why, even if in an individual case the hypotheses generated are useless; this is the very knowledge that the expert has been taught to suppress!

• There needs to be a way of allowing V&V of the knowledge acquired to take place. If the target system goes wrong, there are a number of ways in which this might happen. The fault may simply be a programming error (e.g. something as simple as a typographical error), or a more substantial design fault, where the programming is a perfectly correct encoding of a faulty design. But the fault may also be in the knowledge acquired during the KA process. Since KBSs may deal with life or death situations, it is essential that errors of this sort should be isolable and curable.

• Much KA is done using a number of experts, rather than a single one. It could be the case that no one available person has all the necessary expertise to cover the requirements of the system, or that the top expert in the domain is too expensive for much use during KA, and therefore that assistants have to fill in the gaps, or simply that the target KBS is specifically intended to synthesise the expertise of a number of experts. This situation raises a number of problems. The knowledge engineer has to make sure that the various experts are using terms in the same way. Often in the case of disagreement, it is not clear whether experts agree on the usage of terms and disagree on the properties of the associated concepts, or vice versa (or indeed agree on neither).

In order to tackle these problems, there is a field of software development and psychological research which we discuss below in §§2.2.4-2.2.5. Various software tools are designed with interfaces that are supposed to allow and encourage the expert to express her knowledge in a more or less familiar way, while underlying the interface a special purpose KRL will encode that knowledge in a suitably machine-oriented formalism. §§2.2.1-2.2.2 set out the knowledge modelling methodologies that tackle the problems of V&V and of multiple expert testimony.

2.2 Model-Based Knowledge Acquisition and KBS Development

In this subsection, we examine a research programme in AI that is very current and controversial indeed, the idea of *model-based KBS development*. The work here is, I think, of philosophical as well as engineering interest, but the main reason for including it here is that we should have a firm grasp of the principles underlying *conceptual modelling* and *knowledge acquisition* in the field of KBS development. §2.2.1 will describe (in general terms) the model-based approach to KBS development. §2.2.2 will describe one particular model of expertise called the *conceptual model*, with examples of what a conceptual model might look like, together with discussion of the generic portions of these conceptual models which are supposed to hold across various problem-solving domains. §2.2.3 discusses two ways in which modelling can be utilized as a KBS development methodology; §2.2.4 will examine the KA techniques used in constructing a conceptual model, to indicate the sort of knowledge they can uncover, while §2.2.5 discusses some psychological research that has been performed on these KA techniques. Ultimately, it will be our contention that the conceptual model of the expert might be termed a *competence model* of the expert. That, together with our contention that a competence model can function as a psychological explanation of the behaviour it models will then imply that the conceptual model can function as a psychological explanation of the expert's (or experts') problem-solving behaviour. Then, to the extent that the various stages of model refinement preserve the structure and content of the conceptual model, it should be possible to assert that the implementation (the system) also functions as an explanation of the problem-solving behaviour.

2.2.1 KBS Development as Model Refinement

The development of KBSs is increasingly coming to be seen as an incremental process, with an associated 'life cycle'. This life cycle will be made up of various stages, each of which results in a 'deliverable' — either a document (e.g. requirements specification) or a piece of software (e.g. prototype) — that can be evaluated. A related view is that KBS development should be seen as a process of *model refinement*, as shown in Figure 2.3.

The process begins with the expert, or experts. These are generally the people who actually perform the real-world task to be modelled. Suitable areas for KBS development include areas where expertise is scarce, or not distributed evenly (for example, most AIDS cases occur in Africa, while most AIDS experts are in America), areas where human experts would be at risk or where the need for their expertise would be possible but unlikely (for example, a respiratory disease diagnostician on a space shuttle would be expensive to employ and insure, while her expertise might well never get used at all), or areas where the expertise could be put to better use (for example, a KBS could be used for training novices, to free the expert to perform the task as opposed to teaching it). The area in which the expert's expertise is operative is the *domain*. Common types of domain in commercial KBS development include medical domains (e.g. lung disease diagnosis is the domain of PUFF, the monitoring of intensive care patients is the domain of VM, while anaesthetic management instruction is the domain of ATTENDING), electronics domains (e.g. PALLADIO's domain is the design and testing of new VLSI circuits while IN-ATE's domain is the diagnosis of oscilloscope faults), and chemistry domains (TQMSTUNE keeps triple quadruple mass spectrometers tuned, and CRYSALIS interprets the 3-D structure of proteins). Giarratano and Riley (1989) discuss useful guidelines for selecting a domain for KBS development.

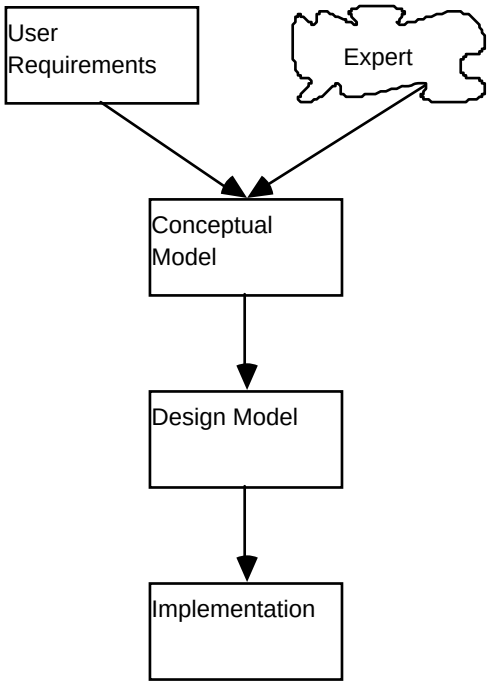


Figure 2.3: KBS Development as Model Refinement

The developer of the KBS, the *knowledge engineer*, interacts with the expert(s), on the basis of the content of a document called the *user requirements specification*,[1] to produce a model of the expertise that would be needed to perform the tasks specified in the user requirements. This model is called the *conceptual model*, and is intended to be a model of expertise only. This will also, in some sense, be a model of the domain, but will not be dictated by the domain structure. So, for example, it might be the case that there is some particular hierarchy of diseases in some medical domain. However, the structure of domain objects in the conceptual model is more likely to respect the similarity with respect to the observable characteristics of those objects rather than the scientific classification. Hence, if two diseases, aetiologically very distinct, have very similar symptoms, then they are likely to be grouped together (although, as the discussion of explanation above implies, the system may have to store the aetiological structure as well for the purposes of deep explanation of its output).

[1]I.e. a specification of the requirements that the person/corporation that will ultimately use the system have for the system. He who pays the piper calls the tune! This document will basically be a guide for the knowledge engineer, specifying which problems are to be solved, by whom, and what role the target KBS will play in the solution of those problems (e.g. will the KBS solve the problems on its own, advise an expert, advise a novice, or simply 'walk through' training examples with a class of trainees?), together with other stuff about the type of machine the KBS will run on, the computer language(s) preferred for the implementation, maximum cost of the project, and so on.

This conceptual model is then transformed into a *design model*, which is a model of the final system. The aim of this transformation is to preserve as much as possible of the content of the conceptual model, while introducing new constraints on the model determined by the constraints on the target KBS (e.g. constraints on the efficiency of the algorithms used in the computation, constraints on the amount of memory available, and so on). The design model is then transformed into the final *implementation*.

This is a very rough description of a complex process, upon which there are several views, constituting a research programme that is very much current. There are various views on how abstract the various notions of models are. For example, the KADS methodology (Wielinga et al 1992) divides the conceptual model into two: a *model of expertise*, which is a statement of the knowledge required to solve the various problems, and a *model of cooperation*, which specifies which agent (e.g. the user, the KBS) will perform which task. On the other hand, the VITAL methodology (Shadbolt et al 1993) has a single conceptual model, but specifies that what we call here the design model should be divided into the *functional design model* and the *technical design model*, where the functional design specifies the functionality that the machine must have available to it to solve the problems that it will need to solve, and the technical design model describes the architecture of the machine and specifies *how* the functionality specified in the functional design model can be achieved. Further, the arrows in Figure 2.3 might be seen as implying some sort of temporal ordering between construction of the models. Indeed, they do correspond to some rough temporal ordering, but there may be feedback loops if, for example, the design model turns out to need more expertise than was initially gathered in the conceptual modelling stage (Motta et al 1994). More drastically, if the aim of the KBS development project was to rationalize some area of problem-solving in a corporation, then the very process of development itself can often result in a restructuring of a corporation or revision of its aims. The result of that can be that a model in the later stages of development can cause a revision of the original requirements specification!

A first key question is the methodology for the move from expert to conceptual model. This is the process called *knowledge acquisition* (KA). How does the knowledge engineer get the knowledge out of the expert and into the model? The first thing to note is that the question, phrased thus, is rather misleading. The KA process was, certainly, traditionally viewed as a process of extraction of 'nuggets'

of information from an expert and transferral of those nuggets into the KBS. The task of the knowledge engineer then would be to ask the expert which rules were applicable in various circumstances, and then translate the natural language answer that he receives into an appropriate formalism. However, it was pointed out by several authors (Hayward et al 1987; Wielinga and Schreiber 1989; Morik 1989) that that was not really good enough. For instance, it is essential, if that extraction process is to preserve the expert's problem-solving behaviour, that the expert and the knowledge engineer must agree on the meanings of the terms used by the expert, and that that agreement must also be in accordance with (i.e. representable in) the formalism used in the encoding process. Hence the KA process must at least include a period in which the knowledge engineer comes to share the viewpoint of the expert (and possibly change the expert's view as well — see §2.2.5 below).

A further point is that building a KBS is obviously a practical project, not generally intended to demonstrate psychological points. Hence, the conceptual model that results from the KA process does not always respect the expert's view of her own problem-solving. The expert's behaviour is always a good starting point for the construction of models, but there will be many reasons for departing from that paradigm. Firstly, the expert's own behaviour may well be flawed in some respect, of course. This possibility is discussed below in §2.2.5. But even if the expert's behaviour is ideal for the problem-solving she is employed to do, there are still factors that typically result in the conceptual model (or models at a later stage of KBS development) departing from her practice. For example, there are inherent differences between the capabilities of machines and people. A diagnostician may well be able to tell what is wrong merely by looking at her patient — clearly a machine cannot reproduce this sort of diagnostic acumen (and will not in the foreseeable future). On the other hand, a machine is capable of feats of memory storage and calculation speed that most people can only dream of. Barthélemy et al (1988) describe an experiment for the KADS project where a small KBS was being developed to configure moulds. The experts only generated a small set of possible solutions; however, the KBS was able to generate all possible solutions to the problem, and search through that enlarged set while remaining well within the memory parameters set at the beginning of the experiment. Hence the aspect of the expertise that enabled the search space to be pruned without compromising the final solution could simply be ignored. Furthermore, most KBSs are quite small scale projects, and would not attempt to equal the performance of an expert; hence a model with the breadth of a full

scale psychological model is rarely required. The moral is clearly that we need to be clear about the extent to which such systems and models are psychologically explanatory.

2.2.2 Conceptual Models

A conceptual model is a set of sentences (often in a formalized language) describing expert problem-solving. But a conceptual model need not be *simply* a model of an expert. The knowledge engineer has his requirements specification to satisfy, and will therefore take knowledge from whatever source he can in order to fulfill his own task. Typically, there will be a number of experts with whom he can build a model (another source of information will be articles and textbooks where appropriate). But great care has to be taken with, for example, a pair of experts who use the same term in differing ways (or who use different terms for the same object). The conceptual model, then, will be something like a description of the knowledge that an expert may require in order to solve problems in the domain. Hence, to an extent, the conceptual model is to be seen as a model of the expert (or the expert's expertise), but this story needs to be augmented. The 'expert' in question is often an abstraction from the several experts employed in the KA process, or sometimes an abstraction from the various (perhaps idiosyncratic) capabilities of a single expert. Equally, it is possible to view the conceptual model as 'abstract descriptions of the objects and operations that a system should know about' (Wielinga et al 1992, p.12). However, this will not quite do, since the structure of the model will depend to some extent at least (and, typically, to quite a large extent) on the expert's problem-solving structures uncovered in the KA process. After all, the expert's problem-solving behaviour is likely to represent the state of the art with respect to knowledge about the domain. Human problem-solving behaviour is generally respected in the conceptual modelling phase of KBS development.

In order to minimize the problems associated with the identification of the 'same' piece of knowledge over the testimony of several experts, not only does the knowledge engineer try to elicit knowledge, but also he will attempt to establish the role or roles that that knowledge can play at various stages in the problem-solving process. This leads to a separation in the components of a conceptual model into various epistemic types.

The basic epistemic division in the field is that between *domain knowledge* and *control knowledge*. Domain knowledge is simply knowledge about the objects (which may, of course, be abstract) in the domain, and their properties and relations. Control knowledge is knowledge about how to get things done. However, there are complications and simplifications that can be made to this type of epistemic hierarchy. One well-known example of such an epistemic hierarchy is the KADS four-layer conceptual model (Wielinga et al 1992). This is rapidly becoming a standard, so we will use this as a brief example of the sorts of distinction and division that are available. Karbach et al (1990) show that most types of conceptual model can be reconfigured into these four layers. Hence we will not lose generality if we take the KADS four-layer model as representative.

The KADS four-layer model respects the domain knowledge/control knowledge distinction, but effectively 'divides' the control knowledge into three. Hence the first layer, the *domain layer*, simply *is* the domain knowledge, and 'embodies the *conceptualization* of a domain for a particular application in the form of a *domain theory*' (Wielinga et al 1992, p.15). The second, third and fourth layers are all aspects of the control knowledge, the knowledge about what to do when. The second layer is the *inference layer*, and describes the inferences that can be made in the domain in terms of *classes* of sentences expressing domain knowledge. These sentence classes are developed according to the *roles* that various pieces of knowledge play in the problem-solving (for example, the same piece of domain layer information, say, *X has influenza*, might at different times and in different contexts function in the problem-solving as an observation, a hypothesis, a diagnosis, etc.). In theory, when one is possessed of a model of expertise consisting solely of a domain layer and an inference layer, one would then have enough knowledge to derive (extremely inefficiently) every possible piece of information in the domain. However, one would have no means of making the right inferences in the appropriate places. For this, the other two layers are required. The *task layer* is the third layer, and this contains information about how to combine elementary inferences to achieve goals; the fourth layer, the *strategy layer* takes this process further, containing knowledge about how goals are put together to solve global problems in the domain (it is doubtful whether there is a highly principled distinction between the task layer and the strategy layer). See the discussion of Figure 2.4 below for a brief example to motivate this epistemological distinction.

This epistemological layering of models has enabled knowledge engineering to adopt a key idea in software engineering, that of the reuse of software where possible. The central suggestion here is that (portions of) the control layers of models can be reused across domains and applications.[1] The suggestion is made partly as a response to an imperative, and partly as a means of cashing in upon a genuine discovery. The imperative is that the effort of creating a new model of every domain is great; the KA bottleneck would remain and possibly even be exacerbated (even if modelling methodologies would make knowledge representation or V&V easier). However, if *libraries* of models with wide application were available, then the knowledge engineering task would be made that much simpler; an off-the-shelf model could be selected which conceivably applied to the problem-solving in the domain (and indeed Chandrasekaran et al (1989) make the claim that such models can be used in explanation). Even if the model had to be altered slightly, the effort would be very much reduced. The discovery was that it was indeed possible to talk about problem-solving in these terms (Clancey 1985), and it was not at all implausible to suggest that certain problem-solving structures could be discerned across domains.

So, libraries of models began to appear. KADS produced an influential library of *interpretation models*, which were skeletal models of common types of problem-solving, containing an inference layer, together with a default task layer control structure (Breuker et al 1987). An example of the inference layer of such a model, systematic diagnosis, is given in Figure 2.4. The model can be seen in the following way. It represents the flow of data or knowledge. Each rectangle stands for a *metaclass* of domain knowledge, indexed according to the *role* the knowledge plays in the diagnosis (in the later KADS-II project, metaclasses are in fact called roles). So, for example, we have a *complaint*, a *model* of the system, a *norm*, a *variable value*, a *hypothesis*, etc.. The ovals stand for basic inferential steps (*knowledge sources* in KADS-I, *inference steps* in KADS-II). So in systematic diagnosis, we have such inferences as the *selection* of a system model on the basis of a complaint, the *selection* of a variable from a universum of observables, etc.. The arrows show the direction of dataflow.

[1]Obviously the domain layers could not be reused across domains.

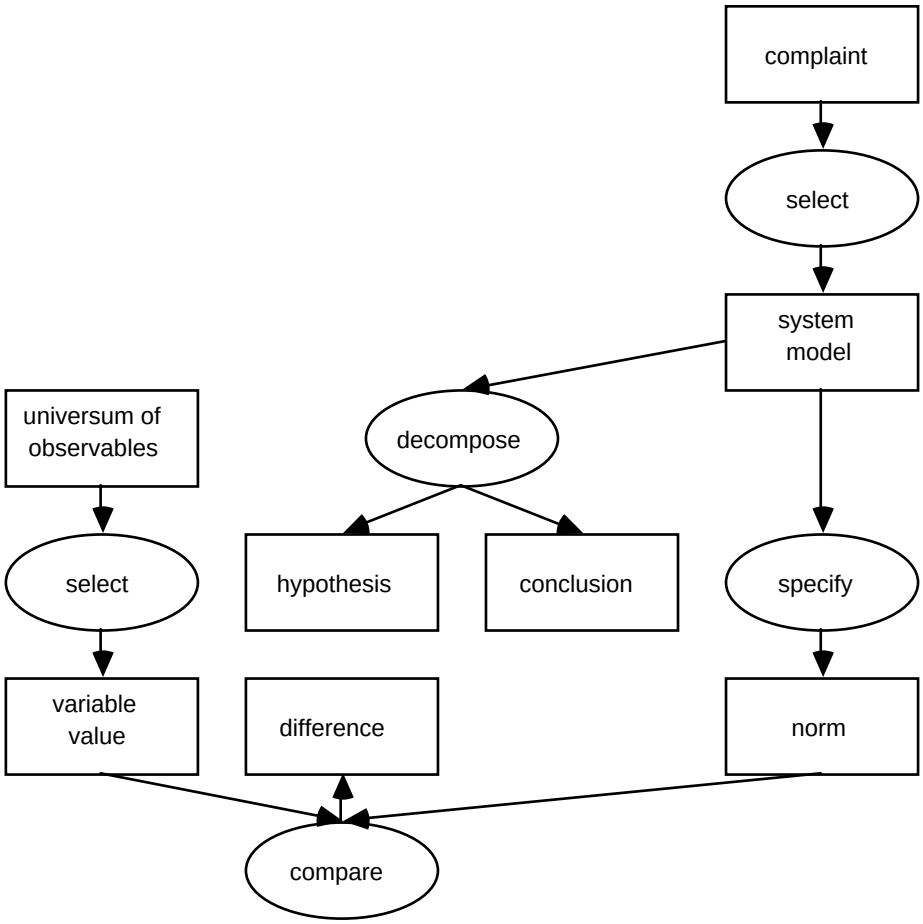


Figure 2.4: The KADS-I Model of Systematic Diagnosis

So the diagram of the inference layer of systematic diagnosis says that, first, the diagnostician receives a *complaint* — an indication of a dysfunctioning system. Such a system might be an economy, a human body, a piece of machinery, etc.. Let us use the example of a human body. So there is a complaint, or symptom. On the basis of this the diagnostician *selects* a *system model*. This is a model of how a subsystem of the whole would function normally. So, in our example, if the complaint were persistent wheezing on the part of the patient, the system model might be a model of the respiratory system. The system model is then *decomposed*; this produces a number of *hypotheses*. Each hypothesis will therefore be some subsystem of the system model. Hence if the system model is the respiratory system, one hypothesis might be that the lungs are malfunctioning, another might be that the system transferring oxygen from the inhaled air to the blood is faulty, and so on. Each hypothesis is then tested. The system model is used to *specify norms*, which are expected values of parameters. In parallel, an *observable* (test) is *selected* from the *possible observables*, such that that observable can be *compared* against the expected value of the

parameter. The comparison is made and the *difference* established. Then on the basis of this difference, the hypothesis is tested. If the difference is small or non-existent, then the hypothesis that the particular part of the system is responsible for the complaint is rejected. The system (the body in our example) is generally allowed to deviate to a small degree from the perfect functioning of the abstract system model. But if the difference is large, then the hypothesis becomes the *conclusion*.

Because this is the *inference layer* of the model of systematic diagnosis, it contains all the inferences that can be performed during systematic diagnosis, but it does not say when and why to perform them. The dataflow diagram can be navigated in various ways. For example, the direction of inference could be as described in the previous paragraph. This would correspond to the diagnostician diagnosing the problem with a faulty system. Or we could start with the conclusion and work backwards, to see what symptoms can be expected. For example, if it were known that there were a trade deficit in the economy, an economist might want to know what trouble that would cause in the economy as a whole, in order to decide whether or not it were worth the effort and political fallout of attempting to rectify the deficit. Or, the systematic diagnostician might test all hypotheses, or just test them until he finds one with a sufficiently large difference from the norm. All these ways of navigating through the dataflow diagram would be performed using the same knowledge, which would be 'contained in' (i.e. indexed by) the various boxes and ovals in the diagram (so, for example, the system model box would contain all the knowledge pertaining to the ideally functioning system, the compare oval would contain all the knowledge required to compare a norm with an observable, and so on).

So *task layer* knowledge is required actually to solve any problem (in real time) that the inference layer knowledge might be used for. Task layer knowledge will be used to structure the navigation of the dataflow diagram in order to achieve some particular goal. This task layer structure is called the *control structure* of the model, and is designed to tell you *which* inferences to perform *when*. A KADS-I interpretation model is made up of an inference structure such as that shown in Figure 2.4, and also a default control structure. The default control structure can be disregarded if it is inappropriate, and replaced with another structure. If no control structure is supplied, the model will not specify a way of solving the problem in real time.

The default control structure that comes with the KADS systematic diagnosis inference structure is the following.

```
find (diagnosis)
   select (system-model)
      WHILE (no conclusion)
            decompose (system-model)
            WHILE ((number hypothesis) > 1)
                  select (variable-value)
                  specify (norm)
                  compare (variable-value norm)
```

The control structure is written in pseudocode, a sort of cod programming language. The goal is to find a diagnosis. Hence a system model is selected, and while there is no conclusion, the following is done: decompose the system model, and then, while there is more than one hypothesis, select a variable value, specify a norm and compare them. Other control structures could be dreamed up to navigate round the inference structure to perform different goals.

The interpretation model, having being selected according to various criteria, would then be *instantiated* by having the inference layer classes filled with domain layer knowledge (this is really two knowledge engineering tasks in one: the addition of the domain layer and the connection of the domain layer with the inference layer), and the task layer control structures customized to fit the actual problem-solving. In complex models, a strategy layer could be added as well. This is the basic method of performing model-based knowledge acquisition. A skeletal model is selected from a library, and this model can be used to structure both the knowledge acquired, and the target KBS. Furthermore, the model can aid KA by telling the knowledge engineer what (types of) knowledge to look for. Note also that the high level descriptions of problem solving practice such as were required for the complex goal-based types of explanation discussed in §2.1.4 above, can be provided using the terms of such models as guidance.

Although KADS is in many ways an archetype, there are other methodologies in this field. A second example, which is becoming increasingly standard in the U.S.A., is the *generic task* methodology of B. Chandrasekaran, which identified a number of task/subtask structures on the basis of examination both of actual tasks (Brown and Chandrasekaran 1989), and of the problem-solving structures ideal for KBSs to embody (Chandrasekaran 1990). These task/subtask structures, together with their knowledge representation implications, are laid out in a task analysis. A third example is the approach of Mark Musen's Medical Computer Science Group at Stanford, which claims to have uncovered domain-independent

abstract models of problem-solving, called *problem-solving methods* (McDermott 1988). Based on these methods, knowledge engineers can build computer architectures that are oriented specially towards the method that they are built to carry out (Musen 1989ab; Marcus and McDermott 1989). Methods are AI-oriented ways of carrying out *tasks*, and themselves are *composed* of *mechanisms* that are close to computational primitives (a mechanism is described as 'a method that does not decompose a task into subtasks' (Puerta et al 1992, pp.2-3)). A fourth approach to reusable modelling structures takes this decompositionality further, and argues that models not unlike KADS interpretation models (minus the default control structures) can be developed using a library of the *components* of KADS models, linked with a *grammar* which allows the building of many (indeed, since some rules are recursive, infinitely many) non-standard models (O'Hara 1993). Therefore the *generalised directive models* (GDMs) built using this approach can avoid the charge that the finite libraries of KADS models often underestimate the *sui generis* nature of much problem-solving (O'Hara and Shadbolt forthcoming), and can make the indexing and organization of libraries of models more sound as well (O'Hara 1993).

The notion of a conceptual model will be very important in Chapter Five, when we begin to draw the various threads of this thesis together. So it will be worthwhile summarizing the important ideas behind conceptual modelling, to make matters clearer when the time comes to consider them as psychological explanations. A conceptual model is a model of the expertise required to solve a particular problem. It will not always be a model of a single expert. The knowledge acquired and put in the model can be typed epistemologically, resulting in a four-layer structure. The domain layer contains the static knowledge of the domain objects and their interrelations. The inference layer is a specification of the basic inferences made by the experts in problem-solving; these inferences can be grouped together in an inference structure. The task layer contains the knowledge about how these basic inferences are grouped together to achieve goals. This knowledge controls the inferences. Finally, the strategic layer contains the knowledge about how to configure a set of goals to solve a large scale problem. It is arguable that there is no principled distinction between the task and strategic layers, and many models will not have a strategic layer at all. Karbach et al have shown that the four-layer model can be seen as representative (1990). Groups of domain-independent skeletal models

(containing inference and task layer knowledge) can be put together as model libraries, to ease the modelling task for knowledge engineers.

2.2.3 Two Philosophies of Modelling

One point that was hinted at in the previous section is that there are two different philosophies of modelling that we can discern in model-based AI. On the one hand, there is the type of modelling that seeks for well-known structures in the problem-solving that goes on in the domain, and then attempts to match those structures with model components that are stored in a library of model components (such as interpretation models, generic tasks, problem-solving methods and GDMs, introduced in the previous section). And on the other hand, there is the view that emphasizes the uniqueness of any problem-solving practice, and maintains that modelling should respect this fact (Clancey 1991). This second view points out that any domain independent structure (such as a generic task or a KADS model or a GDM) will be a generalized structure that will not respect the actual problem solving practices in a domain — these structures will tend to cut corners and potentially miss out important domain dependent practices. On this second view, the process of KA should preferably be a process of model construction produced by interaction and negotiation between knowledge engineer and expert. Ready made structures such as the KADS model we saw in Figure 2.4 should be treated with scepticism.

Of course, for the purposes of a knowledge engineer, this philosophical question — top-down or bottom-up — is not particularly pressing, since he is not going to be interested in preserving exactly the expert's problem-solving practices. However, the question will be important for us. Most conceptual models are going to use generic languages (even if only first order logic). If the extent to which a conceptual model can stand as an explanation of the expert's problem-solving practices is going to be determined by the extent of the use of generic model components used in the model, then the position advocated by this thesis will be weakened considerably (although the main philosophical point will stand). We want to claim that the models developed during the KBS development process, almost as a by-product, give us psychological explanations of the expertise; hence we will need to argue that generic problem-solving structures do not compromise the ability of AI models to describe expertise.

To put the point more explicitly, it happens to be the case that such are the considerations of time (both time allowed for modelling and KA to be performed, and the time for which an expert is available), that ready made structures such as generic tasks or KADS models are essential for model-based KA. Models cannot be constructed from scratch unless either an expensive state-of-the-art KBS is required for which the commissioners of the project are prepared to pay, or the project is an academic project where time is not money. Hence virtually all model-based KBSs will be built using generic structures from model libraries. For our thesis to apply to more than a vanishingly small set of systems, therefore, we will need to argue that the use of such library-based generic structures will not compromise our main claim that models associated with KBSs can be psychologically explanatory of an expert's expertise. We shall produce this argument in §5.2.3.

2.2.4 Knowledge Acquisition Techniques

We now examine the techniques used by knowledge engineers to create the conceptual models that result from the knowledge acquisition (KA) process. This will give us a greater idea of the types of knowledge that go into a conceptual model, and also a better idea of the modelling process in general. The way that knowledge engineering works is to apply knowledge acquisition techniques, usually embodied in software tools, to the expert. If the techniques are embodied in tools, then the result will be a set of knowledge bases all of which are represented in standard formalisms (i.e. the KRLs of the various KA tools). If the tools are embedded in 'workbenches', suites of tools designed to be used in a complementary way, often within a methodology such as KADS or GDMs, then there will be the hope that redundancies of knowledge will be avoided, while covering all the knowledge required for problem-solving in the particular domain. Examples of such workbenches are the Shelley workbench, which accompanies KADS (Anjewierden et al 1992), the VITAL workbench, which is GDM-based (Domingue et al 1993), and the KEW workbench, which was developed in the ACKnowledge project (Shadbolt and Wielinga 1990).

Some techniques[1] are more or less obvious. For example, there is *interviewing*. Interviews can be *structured* or *unstructured*. An unstructured interview involves the knowledge engineer and expert discussing the problem-solving practices

[1]See (Shaw and Woodward 1990) for a lengthy comparative discussion of various KA techniques, including all the examples given in the following text.

under review with no set agenda, whereas in a structured interview, the knowledge engineer will have a series of questions that he will wish to ask, or a series of topics for discussion, planned in advance. Similarly, an expert can produce a *report* on an episode of problem-solving, either a report on her own solution to a problem (and this can be produced simultaneously with the problem-solving, or *post hoc* as she views a video of her own performance) or a commentary on the problem-solving of another expert. These reports (which are usually verbal) can then be analysed by their being transcribed to produce a *protocol*, which can then undergo *protocol analysis*. Protocol analysis is intended to bring out such things as domain ontologies; the expert will go through the transcript, highlighting pieces of text, connecting them, pointing out synonyms and antonyms, etc.. A hypertext system can usually keep track of all the manifold connections. Another technique is *card sorting*, which involves transferring the names of the domain objects onto cards (which may be 'cards' on a computer terminal screen), which are then sorted into 'piles' by the expert. This will produce a classification of the domain objects into groupings. *Laddering* is a technique whereby the expert creates and groups domain objects into a hierarchy, under the knowledge engineer's supervision. So, for example, the laddering tool may produce a *laddered grid* such as the one shown in Figure 2.5. This grid will also be a frame structure; i.e. each node in the tree will be a frame, with slots, values, default values and inheritance, as discussed in §2.1.2.

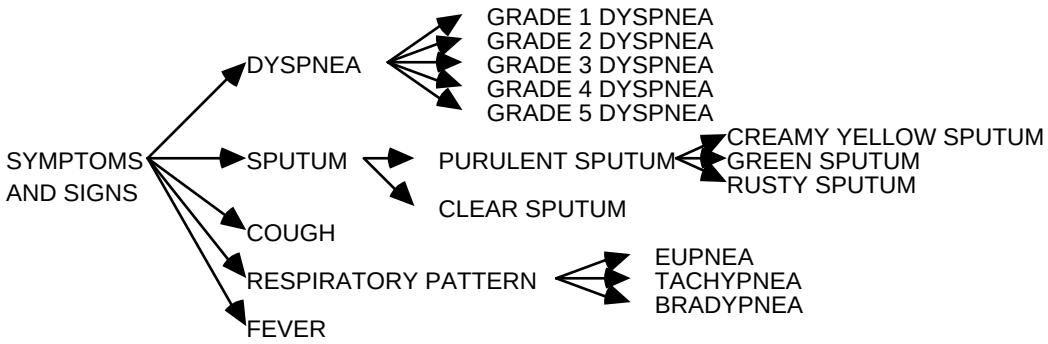


Figure 2.5: A Laddered Grid of Symptoms and Signs in the Respiratory Disease Domain

Another method is the *repertory grid* method, which is a method based upon Kelly's personal construct theory (Kelly 1955). Personal construct theory views the individual as a system that anticipates the future on the basis of hypotheses about the replication of the past, and that receives feedback about its actions

which causes the individual's constructs to change. A construct is a dichotomous distinction which the individual tries to fit over the world.

> Constructs are used for predictions of things to come, and the world keeps rolling on and revealing these predictions to be either correct or misleading. This fact provides a basis for the revision of constructs and, eventually, of whole construct systems.
> (Kelly 1955)

Personal construct theory can be used as a logic for describing agents' cognitive processes. A repertory grid (Shaw 1980) is a KA technique based on Kelly's theory. It is a

> two-way classification of data in which events are interlaced with abstractions in such a way as to express part of a person's system of cross-references between his personal observations or experience of the world (elements or entities), and his personal constructs or classifications of that experience (attributes). ... In the development of expert systems the entities might be key elements in the problem domain such as oil-well sites or business transactions, and the attributes express what the particular expert sees as the crucial distinctions between the entities in the context of the problem.
> (Shaw and Woodward 1990, p.201)

The main point about a repertory grid is that it reveals some ways in which the expert *unconsciously* parcels out the domain. The expert is not usually aware of the constructs that will be elicited. Nevertheless, it is difficult to explain the problem-solving without citing those constructs.

2.2.5 The Psychology of Knowledge Acquisition

This leads us to the question of the psychological basis for applying KA techniques. This question has a number of aspects. We want to know that the KA tools mentioned above are able to produce a knowledge base of sufficient power to perform the task. We want to know that the knowledge bases produced are accurate, and reflect the experts' problem-solving knowledge. We want to know that redundant or inconsistent knowledge doesn't get through. We want to know

that enough knowledge gets through, even knowledge that is very rarely used in day-to-day practice.

In a series of papers, Mike Burton et al (1987, 1988, 1990; Shadbolt and Burton 1990) experimentally investigated the efficacy of KA techniques. Techniques were divided into two sorts, 'natural' and 'contrived' — natural in the sense that it is more or less natural to sit and chat about one's work; contrived in the sense that it is contrived, for example, to sort one's domain into abstract hierarchies. Obviously, 'natural' and 'contrived' define opposite ends of a continuum, with perhaps unstructured interviewing on one end, and repertory grid analysis on the other. On the basis of some (rough) metrics, it was discovered that natural techniques tend to produce more information, yet take considerably more time than the contrived techniques (i.e. considerably more time per piece of knowledge elicited). Contrived techniques were more efficient than the natural ones. Interestingly, in some of the real-world experiments, there was little or no overlap between the knowledge elicited using the natural techniques and the knowledge elicited using the contrived techniques — the two groups of techniques appeared to be tapping different types of knowledge, at least in some cases. It is difficult to avoid the conclusion that the experts appeared to be 'using' knowledge that they did not know they had! In other words, the lesson we take from Burton et al is that a model built using contrived KA techniques may be more complete, but certainly not a model of the expert's *conscious* problem-solving procedures.

Using contrived techniques can also lead to other interesting, perhaps surprising, results. Barry Silverman has made a study of common biases in human judgment (1990); table 2.1 gives some examples

BIAS NAME	DEFINITION
Adjustment and Anchoring	Using of heuristics which may reduce the mental efforts required to arrive at a solution at the cost of using the full amount of information.
Base Rate	Ignoring abstract information at the expense of concrete information.
Data Preservation Context	Being more influenced by summarized data rather than the same data presented in detail.
Illusion of Control	Assuming a feeling of control over events that is not reasonable.
Law of Small Numbers	Expressing greater confidence in predictions based on small samples of data with nondisconfirming evidence than in much larger samples with minor disconfirming evidence.

Order Effects	Placing undue importance on the first and last pieces of information provided.

Table 2.1: Common Biases in Human Judgment

Most knowledge engineering practice initially ran counter to the discoveries of these biases. It was *generally* assumed that the expert was *generally* right. However, Silverman, in a series of exercises, investigated a number of serious (human) expert failures by using standard KA techniques, building a conceptual model of the problem-solving in the domain. These models were not used to build KBSs, but instead were scrutinized to reveal the causes of the failures. As a result, Silverman was able to produce his typology of biases. He advocates *criticism-based knowledge acquisition* (CBKA), where the knowledge engineer uses the KA phase, not simply to formalize the expert practice, but to critique it. In other words, the result of the KA phase will be some conceptual model of the desired expertise, but this model will have been developed critically by the knowledge engineer, who will have attempted to weed out biases from the expert's own account, biases both in the expert's original problem-solving behaviour, and also in any *post hoc* rationalizations of his problem-solving behaviour that the expert may have performed as part of the KA process. So an ideal conceptual model will not necessarily be a perfect model of the expert, because the expert may well have some undesirable qualities![1]

[1]Silverman has pointed out to me that his techniques are only really useful in spotting biases after they reveal themselves in corporate disasters (in Silverman's own work, the Shuttle disaster, the Bhopal chemical leak and the Five Mile Island nuclear disaster, for example). Weeding out a bias in advance is very difficult indeed. For example, what looks like a bias in one context might actually be a useful corner cutting hint in another, even if incorrect performance results occasionally. However, it is as well to realise that knowledge engineering involves attempting, not to model an expert warts and all, but to model some problem-solving practice in such a way as to enable a functioning system to be built on that basis. This may well involve the 'improvement' of expertise.

Chapter Three: Scientific Explanation

> DR HERDER: Lady Claire, don't come to me for the truth, only explanations.
> Peter Barnes *The Ruling Class*

To review our position so far: we have formed the intention of investigating AI philosophically by adopting the 'bottom up' manner of approach, i.e. developing a grasp of, and analysing the actual practice of, the first-order discipline. In Chapter Two, therefore, we set out a view of the current state of play in the field of KBS development methodology; our ultimate aim is to examine the philosophical significance of technological developments in this field. In particular, we will examine the claim that the models used in model-based KBS development can be seen as explanatory of the expertise which they model.

However, even bottom up philosophy cannot be performed in a theoretical vacuum, and so we have one further preparatory task. We need to set out the philosophical background to the discussions we will be having in the final chapters of this thesis. Since we will be looking at the explanatory power of a particular set of models, we need to find or develop a philosophy of explanation; we need to know what it means to say that a conceptual model is explanatory of the expertise it models.

The task is a slightly tricky one, in that we have to avoid two difficulties that inevitably suggest themselves. On the one hand, we have to make sure that we are sufficiently critical of any philosophical orthodoxy. Since we have adopted the bottom up approach to philosophy, we should be certain that the philosophy of explanation that we endorse is maximally congenial to current scientific practice. It is no good using a highly idealistic philosophy of explanation, which pays little heed to the actual practice of explanation either in psychology or AI, as a means of navigating through the pragmatic issues raised by a performance-based technology such as AI. On the other hand, we must not develop a philosophy of explanation which is simply designed to give the OK to KBS development models as explanations of the modelled expertise; the philosophy of explanation that we endorse must be *independently* convincing.

We shall begin to outline what we mean by 'explanation' in §3.1. We shall then attempt to deal with the general issues concerning current philosophical accounts of explanation in §3.2. §3.3 will put forward a positive account. The final section

of the chapter will introduce the distinction between causal and non-causal explanation; this distinction will be of relevance to some subsidiary results of Chapters Five and Six.

3.1 What We Are Talking About

We shall delimit explanations by a functional role that they play in scientific practice: explanations confer understanding. If someone understands a process/event/practice/regularity after being given an account of it, then, if he didn't understand it before, we can say that the account was an *explanation*, that the account *explained* the process/event/practice/regularity for him. Explanations are *intended* to confer understanding in this way. Note that the above condition is not necessary; explanations need not succeed in conferring understanding. I can explain Nottingham's one way system to a visitor, only to discover that his English was not sufficient for him to understand. Nevertheless, it does seem clear that my account could be an explanation. The account of explanation that we shall give in this chapter is not intended to be more than a beginning; hence we shall not attempt to define exactly the relationship between the intentions behind the presentation of an explanation, and the actual results of such a presentation.

Our account of explanation is so far in terms of the unanalysed concept of 'understanding'. We do not have sufficient space to explore this concept in detail. However, in this section, we shall explore a few distinctions, which should give us some hints as to how to understand understanding in the context of our explanation of explanation.

3.1.1 Scientific Explanation and Scientific Understanding

To begin with, one obvious distinction is that between explanation generally, and scientific explanation. In this thesis, we are interested in scientific explanation; does this mean that we can ignore ordinary explanation? Or is our account going to be a general account that includes within it scientific explanation? David-Hillel Ruben draws a distinction between two ways of drawing the distinction.

> There are at least two possible senses of 'scientific explanation'. In the first sense, it refers to explanations which are actually given in science. ... In the second sense, the meaning of 'scientific explanation'

> is commendatory, or honorific, in some way. In any event, in this second sense, it is an open question whether any of the explanations actually given in science are scientific explanations at all.
> (Ruben 1990, p.16)

This clearly chimes in with a major theme in this thesis, which is the distinction between top down and bottom up philosophy. Hence it is no surprise that we are going to be interested in the first sense of 'scientific explanation', the bottom up sense, keyed to actual scientific practice. As we noted in §1.1, the top down/bottom up distinction is not as clear cut as it might be. We certainly need to approach scientific explanation with a critical eye, and general philosophical accounts of explanation will provide important and interesting input to that exercise. But we also need to root the discussion in actual practice.

Since we are interested in the science of psychology, we shall restrict our comments to scientific explanation, in Ruben's first sense. This entails a simple distinction between ordinary explanation and scientific explanation, because some explanations are made in the ordinary course of events, and others are made by scientists in scientific situations. But we will not place much weight on this distinction. In particular, we shall not explore the question of whether scientific explanations can actually be *subsumed* by the class of ordinary explanations. For the remainder of this thesis, we shall use the term 'explanation' to refer to scientific explanation, and we will not attempt to evaluate how far our comments will apply to explanation generally.

But as a result of making this distinction, we can sketch a brief account of understanding that makes sense in this context. Since the scientific explanations we will be exploring are to be linked with the actual practice of scientists, then clearly so must our account of understanding. A successful explanation must be successful by scientific criteria. Hence we can say, roughly, that an explanation must confer scientific understanding on the receiver. Again, we won't try to define the notion of scientific understanding, but we can provide some examples of the sort of understanding that we might expect to find in a scientific context.

One example might be a scientist's coming to understand a phenomenon of an unknown type by discovering that the phenomenon falls under a well-known description, and hence being able to apply a set of laws to that phenomenon. So, for instance, the passage of an electrical signal down a neuron's axon might be

explained as an instance of osmosis, as the axon walls become semi-permeable, allowing charged ions to flow through them. Another example of understanding might be a scientist's coming to understand two or more apparently disparate sets of phenomena by discovering that they can be covered by a single set of laws (and hence that, relative to the new theory, the phenomena are of the same type). Here, one might cite the search for Great Unified Theories of force. A set of laws covering atomic forces and a set of laws covering electromagnetic forces might be unified into a single set of laws. In that case, atomic forces and electromagnetic forces would be explained in terms of the G.U.T.. A third example might be a scientist's coming to understand some process in terms of previously well-understood processes. A case of this might be a discovery of a mechanism within a bird that explains its navigation during migration. A fourth example might be a scientist's coming to understand phenomena sufficiently to exploit them technologically. A case of this might be an explanation of the stresses in various materials that enables structures such as bridges to be built more cheaply and more safely. It will be noted that these examples are all relatively instrumental; the explanations enable the scientist to achieve various goals. Wesley Salmon gives a similar set of considerations that might lead an applied scientist to ask for explanations in (Salmon 1988b, pp.120-2).

I don't wish to go into detail about understanding here. Suffice it to say that this mild instrumentalist account is meant to chime in with, say, Neil Cooper's recent discussion of the spatial and geographical metaphors governing understanding.

> The person who understands an event or phenomenon is somebody who can find his way about the mental environment in which his knowledge of it figures. The attempt to understand is like exploration. One can explore an area more or less adequately, it is a matter of degree. Once we have done this, knowing our way about is more than knowing where our destination is or knowing where particular places or objects are, it is knowing how to get from one part to another, how to *connect* them, and how to *distinguish* them.
> (Cooper 1994, p.4)

This is a more or less delphic characterization, but basically it allows us to make a few points without going deeply into the question of understanding. Firstly, we can make a distinction between understanding and knowledge. Understanding is a deeper notion than knowledge — knowledge can be transmitted easily,

whereas understanding is more like a capacity to generate knowledge, to see connections between propositions instead of merely being passively informed. Indeed, knowledge and understanding can even be mutually exclusive in some contexts (Cooper 1994, pp.5-8). We can also distinguish between the general instrumental view of understanding and the rigidly defined types of 'understanding-why' which some philosophers of science use (van Fraassen 1980; Achinstein 1983), and which we shall discuss *en passant* below. These narrow conceptions of understanding are connected with particular views of explanation, and are therefore not independently motivated; it is likely that as accounts of *understanding* they are not rich enough to do justice to the concept.

Finally, we should just note that, because we are talking of scientific explanation, when we go on to discuss psychological explanation in Chapter Four, our remarks will not be intended to apply to common, 'folk-psychological' explanations of human action, necessarily[1] (although it would be a nice bonus if they did). Our aim is simply to look at explanations in scientific psychology.

3.1.2 Explanation as a Process and Explanation as a Product

A second distinction we might discuss briefly is that between explanation as a process and explanation as a product (Bromberger 1965). An explanation can be seen as a process, or an act; you can sit in a lecture hall watching a physics lecturer take half an hour to tell his first year students why clocks on aeroplanes run slow in comparison to clocks on the ground. In that event, it makes sense to say that the explanation ran for thirty minutes, that it was interrupted by the bulb going on the overhead projector, that it took place in temperatures in the nineties, etc.. On the other hand, you can point to the text which the lecturer used, and discuss the man who discovered the explanation, or how complicated it is. In that case, you would be discussing the explanation as a product. Most philosophers of science take the product as prior to the process (i.e. the process must be explained in terms of the product); Peter Achinstein (1983) is the notable exception.

What we should note here is that, in the bottom up context, the priority of product over process is not uncontroversial. Explanations need to confer understanding (or be capable of conferring understanding), and our instrumental

[1]Although if the arguments of Churchland (1970), for example, hold, so that folk psychology is a kind of scientific theory, then our remarks should apply. I won't evaluate Churchland's thesis, although I think it is false.

notion of understanding does seem to entail that we need explanations that are capable of being used in scientific research. This in turn seems to imply that an unusable explanation in the product sense (e.g. an unknown or unsurveyable explanation) should not be seen as a good explanation. If we view the core of an explanation, in the product sense, as equivalent to its content (Ruben 1990, p.7), then it is not obvious that a useful philosophy of explanation can rest entirely on that notion, since actual scientific practice requires more in terms of communicability. Hence attention to the process is essential for a correct account of explanation in science.

Interestingly, this does seem to be a neglected possibility for the philosophy of explanation. There are only four possible relationships between explanation as process and explanation as product.

1. Process and product are independent.
2. Product is prior to process.
3. Process is prior to product.
4. Process and product are mutually dependent.

Since most philosophers of science cleave to the relative importance of product, they must choose either option 1 or option 2. Achinstein stands more or less alone by choosing option 3. However, our brief discussion suggests that the neglected option 4 is at least worth considering.

In fact, though it is important to clear up any troublemaking ambiguity, it is interesting to note that the conceptual models which are the focus of this thesis can actually be seen as either process or product, as the reader thinks fit. Conceptual models can be stated, and written down, in English, Dutch, in some KRL; in that sense they can be seen as products. Perhaps they are most naturally seen that way. However, each model comes associated with a *methodology*, which in turn suggests a *life cycle* for KBS development (recall §2.2.1, and especially Figure 2.3). Models are intended to be *used* in a very particular way, in very particular circumstances. They *inform* the KBS development process in specific ways; this is not surprising, of course, since they were developed precisely for that reason. Hence the context of conceptual model use certainly suggests a process; a KADS model, for example, is intended for use within an application of the KADS method. So any model will carry with it specifications of admissible processes in which it can be deployed. In the light of our bottom

up characterization of the philosophy of explanation, it is interesting to see this intertwining of process and product.

The upshot is that we need not be drawn on the question of whether we are dealing with process or product. If the reader prefers a product account, then what we say will apply to a conceptual model as a product. If the process account is preferred, then the following chapters can be taken as talking of conceptual models in their KBS development contexts. Either way, we should not fall foul of any arguments that purport to show that the product account is prior to the process account or vice versa.

3.1.3 Full and Partial Explanations

A third distinction we might make is between a full and a partial explanation. A full explanation need not be supplemented in any way; a partial explanation needs some additional information. Most philosophers of explanation are interested in full explanations; it is expected that partial explanations can be 'filled out' into full explanations, and therefore can be taken as shorthand for a particular full explanation. Some philosophers see the full explanation as something that is rarely, if ever, given, and therefore that all (most) actual explanations are partial (Railton 1981; Lewis 1986). Of course, what is a partial explanation will depend strongly on what a full explanation is, so this distinction cannot be characterized neutrally. However, in our bottom up context, we clearly will require an account that deals with explanations as they stand. This does not mean that the full/partial distinction cannot be drawn as Railton and Lewis would recommend, as long as actual explanations can be clearly and fairly measured against it. But it does mean that there has to be some pretty strong reason to postulate an unrealizable ideal explanation-form.

There are a few ways in which the distinction can be drawn relative to the account of explanation above. Firstly, given our linkage between explanation and understanding, one could say that a partial explanation is one which conferred partial scientific understanding. Since scientists are clearly neither omniscient nor omnipotent, one might then have grounds for saying, with Lewis and Railton, that all actual explanations are partial. On the other hand, one could argue as follows: understanding enables something to be done; an explanation either does or does not enable that thing to be done; hence, in that context, an explanation is either full or no good at all. So we have two perfectly defensible

ways of drawing the distinction consistently with our bottom up principles which place all explanations on the same side of the line. A third way in which the distinction can be drawn is to utilize the notion of a scientist's *goal*. A scientist wishes to achieve goal G; an explanation might give him sufficient understanding to achieve G, in which case it is a full explanation; or it might get him closer to the target without actually getting him there, in which case it is partial.

On this third construal of the distinction, we can see conceptual models as being either full or partial explanations, depending on the way in which we characterize the scientist's goal. Let us first assume that conceptual models are explanatory (we will attempt to prove this in Chapter Five). If the scientist's goal is to understand how an expert solves a particular problem P, and the conceptual model is the model of the expertise required to solve P, then the model is a full explanation, in that it gets the scientist to his goal. If the scientist's goal is to build a system which solves P itself, then, because the conceptual model still needs to be transformed into a design model, which in turn needs to be implemented (recall Figure 2.3), the conceptual model has to be seen as a partial explanation.

My own view is that this third way of drawing the distinction is the interesting way. One reason for this is that it is the only way of drawing the distinction which leaves actual explanations on one side, and actual explanations on the other. Secondly, it is tied in quite nicely with the instrumental notion of scientific understanding. In so far as the distinction matters, we shall be concerned in this thesis with full explanation. This means that, in particular, we shall be concerned with conceptual models as explanations of the expertise required to solve problems.

We should note that we should not take the full/partial distinction and the ordinary/scientific distinction to be the same. Ruben (1990) claims that if 'scientific' is meant in his second sense, that of an honorific commendation of an explanation, then all non-scientific explanations must be partial versions of this type. The scientific explanation is the only full explanation.

> In my view, the only distinction that can usefully be drawn is that between full and partial explanations, and the distinction between

> scientific (in the [honorific] sense) and ordinary explanations is either that distinction or no distinction at all.
> (Ruben 1990, p.17)

I think this must be false. In the first place, surely a scientific explanation can be partial. One could imagine the honorific sense of scientific being applied to explanations which were incomplete in some way. For instance, one could imagine the honorific sense of 'scientific' given as follows: an explanation is scientific if it gives the causal history of an event (cf. Lewis 1986). Then there is room for saying that the full explanation is the complete causal history, while a partial scientific explanation gives selected highlights of the history. This surely must be a case of a partial scientific (honorific sense) explanation. In the second place, there could be a full explanation which was not scientific. My recent large water bill could be explained astrologically by showing that the configuration of the stars meant that I would have trouble with money this month; the explanation could then be filled out by an account of how the stars affect our lives (there are many such accounts available in occult bookshops). The result would surely be a full explanation (not a good explanation, but an explanation nonetheless). And on any account of 'scientific', it shouldn't be a scientific explanation.

Earlier, we decided that the correct, useful, definition of 'scientific' should be given by observing what goes on in science (rejecting the honorific title view). It is therefore important to show that, by drawing the full/partial distinction as we did, we have not allowed an honorific notion to creep in by the back door. By rejecting Ruben's view, we should have prevented that possibility.

3.1.4 Good Explanation and Bad Explanation

Mention of a bad astrological argument brings us onto yet another distinction: that between good and bad explanations. When we talk about explanations, we will be concerned with explanations both good and bad; a conceptual model can be a bad explanation as well as a good one. Hence it should not be thought that conceptual models are always guaranteed to be good explanations. In general, in keeping with our instrumental notion of understanding, an explanation is good when it is not misleading. So, for example, an explanation of a phenomenon of an unknown type will claim that it falls under some well-known description; it will be a good explanation if the phenomenon actually does fall under that description, a bad one otherwise. An explanation of two disparate sets of

phenomena will claim that they both fall under some single set of laws; it will be a good explanation if the phenomena actually do fall under those laws, a bad one otherwise. And so on.

Further, the good/bad distinction is not the same as the true/false distinction. A counterexample would be a true explanation of some process which could not be used to get nearer a scientific goal (perhaps because it was too complex, or not detailed enough, or unsurveyable).

Note also here that a conceptual model can be seen as both good and bad in different contexts (just as it could be seen as both full and partial in different contexts). Assume again that conceptual models are explanatory. If we want to understand how an expert solves a particular problem P, and the conceptual model is the model of the expertise required to solve P, then the model could be a bad explanation, in that the expert actually solves P in a different way (the model could divert from the expertise in important ways, perhaps because some computational algorithm is known which makes the task easier). However, if we want to build a system which solves P itself, then the model, if it helps us to do that, is (part of) a good explanation (of how to solve P by computer).

We have now specified what concept we are looking at when we look at explanation. Now it is time to evaluate critically some recent and well-known accounts.

3.2 Top Down Accounts of Explanation

3.2.1 Review of Some Top Down Accounts

We now come to top down accounts of explanation, accounts which attempt to set out the concept, without necessarily regarding actual explanatory practice in science. Most philosophers of science have a top down view of explanation; Hempel may fairly be regarded as the major modern figure in the field (1965). His view is that a good explanation takes the form of a deductive argument, whose conclusion (the explanandum) is a sentence expressing that which is to be explained. Some of the premises are lawlike sentences which are actually used in the deduction of the conclusion, and all of the premises are true. Together, the premises form a minimal set with respect to the explanandum (in other words, all the premises are required for a derivation of the explanandum — this is to stop

premises being added to and then removed from the proof by &-introduction and &-elimination). This is the *deductive-nomological* model; the premises can also be statistical in form, which leads to an alternative model to the simple argumentative view (Hempel 1962a). In each case, Hempel draws up a relatively simple form as an epistemological standard to which all actual explanations must be compared. Most other major work in the philosophy of explanation has tended toward this line of providing an ideal form for an explanation.

For example, Wesley Salmon (1984) has developed the idea of a statistical relevance model of explanation. This model claims that explanations consist of sets of empirical probabilities of various possible outcomes. Whereas Hempel's statistical model tries to show how the explanandum was probable, Salmon merely wants to set out the probabilities of the various possibilities.

> An explanation does not show that the event was to be expected; it shows what sorts of expectations would have been reasonable and under what circumstances it was to be expected.
> (Salmon et al 1971, p.79)

Baruch Brody has attempted to synthesize Hempel's work with Aristotle's account of explanation (1972). His account supplements Hempel's deductivist account with further specification of necessary conditions for the premises. So, for example, Brody's causal model dictates that an explanation, as well as being a deductive argument with one or more laws in the premises, must contain a description of the event which caused the explanandum (1972, p.23). His essential property model insists that the premises contain a statement attributing an essential property to objects of a certain class (1972, p.26).

David Lewis's view (1986) is that the complete explanation of an event is its complete causal history (back to Big Bang); the business of the explanations that we are likely to encounter is to give some insight into selected parts of that causal history. Therefore, according to Lewis, in practical terms every explanation is going to be incomplete. Peter Railton (1981) goes further, claiming that the ideal explanation for anything is likely to be infinite.

> For example, an ideal text for the explanation of the outcome of a causal process would look something like this: an inter-connected series of law-based accounts of all the nodes and links in the causal

> network culminating in the explanandum, complete with a fully detailed description of the causal mechanisms involved and theoretical derivations of all the covering laws involved. This full-blown causal account would extend, via various relations of reduction and supervenience, to all levels of analysis, i.e. the ideal text would be closed under relations of causal dependence, reduction and supervenience. It would be the whole story concerning why the explanandum occurred, relative to a correct theory of the lawful dependencies of the world. Such an ideal causal D-N text would be infinite if time were without beginning or infinitely divisible, and plainly there is no question of ever setting such an ideal text down on paper. ... But it is clear that a whole range of less-than-ideal proffered explanations could more or less successfully convey information about

> such an ideal text and so be more or less successful explanations, even if not in D-N form.
> (Railton 1981, p.174)

It is essential that such an account, so clearly the account of a man who will not be called upon actually to give a scientific explanation, should not be seen as being at all descriptive of current practice. Railton is alive to this fact.

> Is it preposterous to suggest that any such ideal could exist for scientific explanation and understanding? Has anyone ever attempted or even wanted to construct an ideal causal or probabilistic text? It is not preposterous if we recognize that the actual ideal is not to *produce* such texts, but to have the ability (in principle) to produce arbitrary parts of them. It is thus irrelevant whether individual scientists ever set out to fill in ideal texts as wholes, since within the division of labour among scientists it is possible to find someone (or, more precisely, some group) interested in developing the ability to fill in virtually any particular aspect of ideal texts — macro or micro, fundamental or 'phenomenological', stretching over experimental or historical or geological or cosmological time.
> (Railton 1981, p.174)

This notion of an ideal explanatory text is an important part of the general project of providing top down accounts of explanation (Salmon 1989, pp.159-61).

David-Hillel Ruben (1990) proposes a dependency model of explanation, where facts are explained by the facts upon which they depend, in a metaphysical sense. Causation is the most obvious metaphysical dependency relation; were this the only admissible dependency relation, Ruben's account would more or less collapse into Lewis's. However, Ruben is alive to the possibility of non-causal explanation (1990, pp.211-18; cf. §3.4 below), and is prepared to countenance alternative metaphysical dependency relations, such as part/whole relations and structure/disposition relations.

The aim of this top down sort of account of explanation is explicitly *not* to describe the sorts of explanation generally found; neither is the aim to describe any restricted class of explanations (e.g. scientific explanations). This main

stream of the philosophy of explanation is more concerned with setting a standard. For example, Hempel is very clear that most explanations which we will actually encounter will not meet the standard. One such class of explanations are explanations in which it is not the case that each essential element of a classical Hempelian deductive-nomological explanation is present.

> When a mathematician proves a theorem, he will often omit ... certain propositions which he presupposes in his argument ... . If judged by ideal standards, the given formulation of the proof is elliptic or incomplete; but the departure from the ideal is harmless: the gaps can readily be filled in. Similarly, explanations put forward in everyday discourse and also in scientific contexts are often *elliptically formulated*. When we explain, for example, that a lump of butter melted because it was put into a hot frying pan [etc.] ... we may be said to offer elliptic formulations of deductive-nomological explanations ... .
> (Hempel 1962b, p.25)

The other case where explanations might fall short of the ideal is the case where an explanation only partially explains the explanandum. We discussed our own interpretation of partial explanation above. On Hempel's characterization, we would be using a partial explanation if, when trying to explain why x was in G (when we might expect that x was not in G), we argued that x had to be in F (of which G is a subclass). This is only a partial explanation, in the sense that it does not explain why x is in G (as opposed to the complement of G in F). Hempel seems pretty clear that most scientific explanations will fall into one or other of these two classes. 'We have found ... that the explanatory accounts actually formulated in science and in everyday contexts ... diverge more or less markedly from the idealized and schematized covering-law models' (Hempel 1965, p.424).

So, there is a tendency in the philosophy of explanation to regard actual existing examples of explanations as being good in so far as they approach the ideal form which it is the business of philosophy (or epistemology) to divine. This can clearly be recognised as an example of top down philosophy as discussed in Chapter One. Hempel (perhaps Popper) can be regarded as the modern founder of this field. Hempel's original paper (Hempel and Oppenheim 1948) has been described as epoch-making, and was the foundation for a consensus that lasted for a couple of decades (cf. Salmon 1989). However, this consensus has been

increasingly under pressure. Most alternatives to Hempel have been equally top down; nevertheless, sociological and epistemological studies of science have increasingly made their mark as bottom up accounts begin to gain footholds. In the remainder of this section, we shall flesh out three reasons to reject the top down approach to explanation; then in §3.3, we shall sketch our positive, bottom up, view.

3.2.2 The Lack of Top Down Consensus

The first reason to reject top down accounts of explanation is simply the plethora of incompatible accounts. The consensus around the Hempelian model that existed in the sixties has decayed. In a recent survey, Wesley Salmon detects no emerging consensus in modern philosophy of explanation, save for general agreement on a few basic propositions.

> Is there a new consensus in philosophy of science regarding the nature of scientific explanation? Not to any noticeable extent. There are, however, a few basic points on which there seems to be widespread agreement among those who are contributing actively to the philosophical discussion of the subject.
>
> (1) At the beginning of the four decades [of discussion of explanation], the view was rather widely held that it is no part of the business of science to provide explanations ... . It is now generally acknowledged that one of the most important fruits of modern science is understanding of the world. We do not have to go outside of science to find it.
>
> (2) There seems to be general agreement ... that the 'received [Hempelian] view' of the mid-1960s is not viable. ...
>
> (3) ... There is, I think, a general tacit recognition that the kinds of tools employed by Hempel and Oppenheim [i.e. the syntax and semantics of a formal language with precise logical definitions] are not especially fruitful for handling the problems [of scientific explanation]. ...

> (4) There appears to be fairly wide agreement on the importance of the pragmatics of explanation, and on the recognition that this aspect was not accorded sufficient emphasis in the 'received view'.
>
> Beyond these four points, I cannot think of any other areas in which consensus actually obtains. Nevertheless, another question should be raised, namely, is a new consensus *emerging* in philosophy of science? This question calls for a risky prediction, but I shall hazard a guess. It seems to me that there are at least three powerful schools of thought at present — the pragmatists, the deductivists, and the mechanists — and that they are not likely to reach substantial agreement in the near future.
> (Salmon 1989, pp.180-1)

Admittedly, the opinion of even one of the most distinguished philosophers of science probably doesn't count as major evidence, but the fact is that there are three apparently incompatible views of explanation being touted, and there is no sign of agreement. Pragmatic accounts (van Fraassen 1980; Achinstein 1983) focus on the relativity of explanation to context; this will be in partial accord with our notion of bottom up accounts of explanation, as can readily be imagined (although, just to confuse matters, this is arguable; we discuss this further in §3.3). Deductivist accounts follow Hempel in claiming that explanations need to be arguments (Watkins 1984). Mechanists, such as Salmon and Ruben, insist that explanations need to be couched in terms of some underlying metaphysical relation.[1]

There are a number of arguments and counterarguments flying around in this field. Many counterexamples have been developed attacking various positions. (Achinstein 1981) is a sustained attack on a number of top down positions, with counterexamples against all of them. There is no need to go into these examples in detail;[2] the main point is that we must suspect that top down accounts

---

[1]To confuse matters further, Salmon calls the deductivists 'top down' and the mechanists 'bottom up' (1989, p.183)! This is, of course, a very different distinction to the one we drew in Chapter One. Our distinction can be roughly mapped onto Salmon's three way distinction between the philosophies of explanation by saying that the pragmaticists could be bottom up, while the deductivists and the mechanists are both likely to be top down. This mapping is, however, a rough one indeed, as discussed in §3.3.

[2]For the record, Achinstein's arguments are aimed at the basic deductive-nomological model (Hempel and Oppenheim 1948), the dispositional deductive-nomological model (Hempel 1965, p.462), the motivational deductive-nomological model (Hempel and Oppenheim 1948, p.254), the functional interdependence model (Woodward 1979), the statistical relevance model (Salmon

(particularly deductivist and mechanist accounts) will be susceptible to the discovery of counterexamples. This is because they privilege one or more types of explanation over others that have at least seemed plausible to scientists in their everyday practice. Scientists have developed very different practices in various different fields, in response to the peculiar requirements of the individual contexts; hence it is plausible that (a) a type of 'explanation' that seems reasonable in some previously uninvestigated context may just be overlooked by the top down view, and (b) the explanation privileged by the top down view may well not transport easily to uninvestigated contexts. Point (a) can be answered in any typical case by saying that the top down account only described and legitimized a circumscribed area of explanatory practice (e.g. in theoretical physics) — in effect by weakening the position, which is unsatisfactory but not necessarily fatal to the top down theory. However, the effect of point (b) is to leave it open that non-explanatory texts (i.e. texts sanctioned by the out-of-context top down theory) get read as explanations; this is much more damaging. A type of explanation that was clearly of use in one context might well turn out to be totally useless in another (uninvestigated) context. For example, there are contexts in which teleological explanations make sense, and contexts in which they don't.

In general, we can anticipate what sort of counterargument could be developed against each of the three positions. The deductivists are vulnerable to explanatory *asymmetries*, cases where the explanandum can clearly be deduced, but not necessarily in an explanatory way. An example here is the explanation of the height of a flagpole, where a set of trigonometrical relations connect the angle of the sun in the sky, the height of a flagpole and the length of the flagpole's shadow. Each can be deduced from the other two, but whereas the length of the shadow is explained by the angle of the sun and the height of the flagpole, it is not clear that the height of the flagpole is explained by the angle of the sun and the length of the shadow (Bromberger 1966). Mechanists typically privilege a particular mechanism; they therefore run the risk that they will be faced with putative situations where their privileged mechanisms will not appear to be explanatorily relevant. For example, someone such as Lewis who thinks that all explanation elucidates the causal relations of the explanandum is likely to be faced with apparent counterexamples of non-causal explanation (we discuss non-causal explanation in §3.4.1). For example, if all explanations are causal,

---

et al 1971), the Aristotelian essential property model and causal model (Brody 1972), and the causal-motivational model (Davidson 1963).

what are we to say about an 'explanation' of why the square on the hypotenuse is equal to the sum of the squares on the other two sides? Finally, pragmaticists will be faced with attempts to reduce their pragmatic accounts to 'anything goes' relativism (Kitcher and Salmon 1987). If there are no restrictions on what is pragmatically acceptable, then it is unclear how to prevent people from saying that everything is pragmatically acceptable. On the other hand, if there are such restrictions, then what is pragmatically acceptable — which is determined by the pragmatic theory, which is supposedly applicable to all contexts — will not be a function solely of context. For example, van Fraassen (1980, pp.132-4) gives an example of a context where the height of a tower is supposedly explained by the length of its shadow.[1] The reasons why the length is explanatory in this case are highly dependent on the details of the story van Fraassen tells; it is not clear how a theory of explanation could possibly legislate for such an outlandish possibility in advance.

But whatever the details of any particular refutation of any particular top down theory, what I am urging is that the very diversity of views of explanation is circumstantial evidence that there is no common type of explanation that should be privileged above the others. The incompatibility of the various views is important. Deductivists are all agreed that explanations should be arguments; however, of course they disagree among themselves as to what properties these arguments should have. Mechanists agree that explanations should be based on metaphysical relations, but disagree among themselves as to which relations are important. Further, deductivism and mechanism turn out to be incompatible, because the mechanist view sees induction as an important method of scientific progress and explanation, whereas deductivists do not (Popper 1972; Salmon 1988a; Kyburg 1988), and because deductivists are generally more at home in a deterministic world, whereas mechanists — following results in twentieth century physics — are happy for the world to be nondeterministic (Salmon 1988b). The pragmaticists are obviously incompatible with both mechanists and deductivists, because they would want the context to determine both which mechanism, and which explanatory form, is appropriate. Hence choosing a top down view will involve choosing a position which is guaranteed to be under attack.

[1]Kitcher and Salmon (1987, pp.310-2) dispute this claim, but we needn't resolve the question here.

On the other hand, the bottom up view bases its account on actual explanatory practice. This is obviously incompatible with the top down views in one sense; it holds that a number of explanatory practices might be legitimate, which the top down views all agree is false. However, in another sense, the bottom up view has the advantage that it is compatible with the *accidental* truth of any top down view — it might actually turn out to be the case that only one (or a few) kind(s) of explanation work. Hence the bottom up view is in an important sense rather more general than the top down views; less prescriptive, more descriptive, but still critical.

To summarize this first reason: most, if not all, top down views, have experienced a measure of criticism based on counterexamples to their accounts. No consensus has emerged. This does not of course prove that all top down views are false (although necessarily at least all but one of them is false). But the absence of consensus is at least circumstantial evidence for competing intuitions, which perhaps may be treated in a less question-begging way with a bottom up view rather than a top down view.

3.2.3 Ideals of Explanation Do Not Explain Alternative Explanatory Practices

The second reason to reject top down accounts is that the ideal forms against which all explanation should be measured abstract away from pragmatics; the top down claim is that explanation can be analysed in the absence of a pragmatic account of explanation-giving (cf. Lewis 1986, pp.193-5; Ruben 1990, pp.21-3). For example, when Hempel talks about partial explanation, what he is really concerned about are the situations where what seem to be perfectly good explanations fail to meet his model. But this will always happen, since the Hempelian type of model takes no pragmatic or practical considerations to have any bearing on the explanatory project.

Nevertheless, sometimes what appear to be pragmatic principles can impinge on the analysis of explanation. For instance, Peter Lipton (1990) develops a view of explanation as *contrastive*; what a good explanation will do is distinguish between two or more possibilities. These possibilities will vary from circumstance to circumstance, and therefore *simple* models will find it problematic to take such variation into account. For example, the form of an explanation of why X's behaviour is paranoiac will depend on whether the

questioner wants to know why X's behaviour is paranoiac *as opposed to* schizophrenic, or why X's behaviour is paranoiac *as opposed to* Y's behaviour being paranoiac, etc..

The obvious claim to make is that top down accounts actually entail that the ideal explanatory text would contain every bit of information that the contrastive explanation would contain, and hence that the scientist could still be seen as 'filling in' slots in a skeletal ideal text. This is arguable; nevertheless, we have then only accounted for the reproduction of arbitrary parts of the ideal text. What is not explained is why *particular* scientists select *particular* contrasts, or *particular* parts of the ideal text, as being of special interest at any particular juncture of science. Explanation of *this* will be an explanation of the pragmatic element in the scientist's practice. But note that that explanation will be sufficient to account for the scientist's explanatory practice *in toto*. Hence work has to be done to show that an ideal account adds anything to the account of explanation.

We should expand on the claim that the explanatory ideal is redundant. If an account of explanation is based on actual explanatory practice, then that account can claim to be explanatory of that practice, even if the actual practice departs from the theoretical account. As an analogy, consider a physical account of the mechanics of ordinary medium sized objects such as billiard balls. That account is an attempt to model the actual interactions of such objects; yet the behaviour of those objects will differ from the model. The model will include such impossible phenomena as frictionless planes, and will ignore the effects of other phenomena such as air resistance. Now, when this account is used to explain the movements of actual objects, it will need to be supplemented by accounts of friction, etc., for the explanation to be accurate in sufficient detail. But the supplement is a genuine supplement — the original theoretical mechanical account is still an important part of the explanation.

We can see this supplementation of explanation in our own discussion of model-based KBS development methodologies in §2.2.1. We explained the activities of knowledge engineers by a model of KBS development that involved the drafting of user requirements, the development of a conceptual model on the basis of those requirements, the transformation of that conceptual model into a design model, and the implementation of that design. This was a bottom up explanation of KBS development practice. A knowledge engineer may well depart from that 'ideal form'; for example, a relatively simple conceptual model may well be

implemented immediately. (Motta et al 1994) gives an example of this. Hence, in such cases, the 'ideal' explanation needs to be supplemented by an account of the departure of the knowledge engineer's practice from that ideal. In the case of Motta et al, the account would discuss the reasons for interpolating a design model between the conceptual model and the implementation, and would then show that such reasons did not in fact obtain in that particular case. So the original 'ideal' explanation, plus the supplement, would together provide a fine-grained explanation of the knowledge engineer's behaviour, and each of the two components would be required for the explanation to stand up.

Contrast that case with the top down case, where the actual explanatory practice does not figure as a datum. Top down accounts of explanation are based on other, generally epistemological, considerations. They make claims such as: the understanding of facts E in the context of facts C could only be produced by procedure P. Now suppose that a scientist departs from the ideal, and follows procedure P'. Since P' differs from P, it follows, on the top down account, that the scientist does not understand facts E in the context of facts C. Then the top down account is incomplete, and has to be supplemented. The supplement will consist in a discussion of why the scientist followed procedure P', and a discussion of why P' was followed at all (e.g. to gain technological mastery, or partial understanding). But surely, here, the supplement — the account of the procedure the scientist used, plus the account of the gain the scientist made (or thought he would make) from the use of that procedure — is by itself sufficient for us to understand the scientist's actions. The supplement, by itself, *explains* the scientist's practice. It is not clear why the ideal procedure, P, and the scientist's target, understanding, need be mentioned at all.[1]

The difference, then, between top down and bottom up accounts of explanation is that the former relies on epistemological considerations, while the latter takes note of actual practice. So, in the bottom up case, what we are postulating is that the ability to achieve goal G in the context of facts C is typically produced by following procedure P. Here, when the scientist departs from P, and follows P' (because this is a bottom up account, we can assume that P' does not differ radically from P[2]), the supplement will be concerned with showing how to

[1]The one obvious exception to this argument is the case where the scientist is deliberately trying to follow the ideal, top down, model. In that case, the ideal model would have an important part in the explanation of the scientist's practice.

[2]In the event that P' and P are completely dissimilar, the position is like that in the top down case, and these considerations will not apply.

achieve G' in context C' by following P', where G' and C' are specific instances of G and C. The supplement will consist in showing how G' and C' are not typical, and therefore why P' is preferable to P this time; the supplement will have no *general* significance, however. In the top down case, what is being claimed is that a particular epistemological state cannot be achieved because of a departure from the ideal; the departure is explained by showing which epistemological state *has* been achieved. But this can be discussed without reference to any epistemological states which *haven't* been achieved (and which haven't been aimed at at all), and therefore without reference to any procedures that might achieve such unwanted states.

Hence, in cases where departures from the ideal are the rule rather than the exception, the ideal does little or no explanatory work. The case of scientific explanation is one such case; there are no models of scientific explanation which are such that most scientists' behaviour conforms to them.

As an example, consider a top down explanation of knowledge engineering practice, parallel to our bottom up explanation of that practice in §2.2.1. The brief for such an explanation is this: explain how to produce a computer system that reproduces expertise. So, for the sake of argument, let us assume that our top down account suggests that the knowledge engineer produce a complete domain theory, from which the system could derive all the answers (e.g. if the knowledge engineer was to develop a KBS that diagnosed respiratory system disorders, then build a theory of the respiratory system into the KBS such that the KBS could deduce the disorder from the input of patient symptoms). As we know, actual practice is different from this; we have the conceptual model and design model being built. Therefore the top down explanation needs to be supplemented by an account of the differences from theory of the actual practice. This supplementary account will be an account of why the knowledge engineer built a conceptual model, and why he built a design model,[1] and what the system will achieve, assuming that the resulting KBS would fall short of some epistemological standard that the ideal KBS would have met. But this supplementary account by itself explains the knowledge engineering practice — the ideal account doesn't seem to add anything at all. Another example: if we take Railton's (1981) second order explanation of the practice of giving

[1]And presumably also some sort of account of why the knowledge engineer did not try to conform to the ideal. This may well be interesting, but would seem to be further explanation over and above what is required to explain knowledge engineering practice (unless the contrastive class for the explanation was the ideal practice).

explanations, what the actual practice of scientific explanation will consist in is the 'filling in' of 'arbitrary parts' of the ideal explanatory text. So, in a particular case, to explain the practice of scientific explanation, we will need to explain why the scientist filled in *these* parts of the ideal text rather than *those*, on top of the ideal account. But that is just to explain why the scientist filled in the parts of the text that she actually filled in, which is an entirely adequate explanation of her explanatory practice.[1]

So, to summarize, top down explanation is only of value in a particular case when it coincides with the actual practice. When the particular case departs from the top down account, the supplement will be sufficiently explanatory on its own. Of course, the issue is not as clear cut as we are pretending here; there will be a continuum of usefulness of ideal accounts as they vary from actual practice — which is just to say that the closer to actual practice the account is, the better, which is just to recommend the bottom up approach. The top down theorists can breathe more easily as they take actual practice into account more often, as, for example, Ruben (1990) tries to. However, most top down theorists explicitly ignore actual practice, and some accounts are even designed so that no actual explanation can meet the ideal at all. Hempel is very cavalier about actual practice, although a number of scientific endeavours will count as partial deductive-nomological explanation. On the other hand, it seems pretty clear that the ideal accounts, of, say, Lewis or Railton will never describe ordinary practice.

3.2.4 Actual Explanatory Practice Needs a Philosophical Account

The third reason for rejecting the top down type of account depends on an insistence on doing justice to a particular practice of science which is commonly called 'explanation'. Scientists produce texts which they call explanations, and undoubtedly serve some communicative purpose in their work. This practice actually exists — in different forms in different areas of science — and deserves some philosophical discussion. This insistence can be recognised as an aspect of the bottom up style of philosophy we argued for in Chapter One, and as such the evaluation of this reason does depend to an extent on the arguments in that chapter in a way that the first two reasons do not. The first reason was that the

[1]Though Railton's account will certainly be useful in providing a contrastive class for the scientist's explanatory practice.

lack of consensus meant that the top down approach did not actually look very attractive in the abstract; the second reason was that, in particular cases at least, the top down approach was not doing any explanatory work (except where the actual practice closely coincided with the approach). But the third reason for rejecting the top down approach is simply that there is a practice in science which would effectively be ignored if the top down approach were accepted, and if that happened, another opportunity to cross-fertilize disciplines would be missed.

The point is that explanations are developed for purposes within contexts, and the quality of an explanation is absolutely tied to the context within which it was developed. All top down models have to recognise this contextual relevance, as we noted above (recall that this was one of Salmon's 'basic points of agreement' quoted in §3.2.2 above). The usual response, we saw, is to say that the context twists the explanation out of ideal shape, but its explanatoriness is still dependent on the ideal. But what is noticeable when scanning scientific journals (or even philosophy journals) is how little explanatory work the ideal itself is doing. Explanations just do not tend to follow ideal forms. When scientists argue between themselves about whether some account or other is explanatory, they simply do not refer to the philosophical literature. Their arguments are context-based and more or less *ad hoc*.[1] It might be replied that even if they don't refer to the philosophical literature, they certainly ought to; this reply is inadequate, of course, thanks to the lack of consensus demonstrated in §3.2.2. Because of this lack, scientists would have to resolve the philosophical arguments before they could resolve the scientific ones.

We can give an example from AI. If an AI programmer wants an explanation of some cognitive capacity in order to be able to reproduce that capacity (or at least the overt IO behaviour associated with that capacity) in a machine, then the explanation that he will require will be of use only in so far as he is able to program the machine accordingly. A Churchlandy explanation in terms of neurons and brain structures, for example, would be of no use at all, since we do not know how to understand a computer in those terms (even if we so understood the brain). Hence the eliminativist 'explanation' is no explanation at all — not even an ideal one — in this context, even if it is explanatory in other contexts. I

[1]Note that this is not the same reason as the second reason for rejecting the top down model. The second reason was that if scientific explanation departed from the ideal, the ideal would not itself be explanatory of scientific explanation. This third reason states that scientific explanation does in fact usually depart from ideals, and further, is (or may be) philosophically interesting.

simply cannot progress from brain-descriptions to computer-system-specifications. In the context of AI, if I can go from the explanation to code, then that is all I require. Anything else is a waste of time — and, in the case of industrial applications, money. Now, what the AI programmer actually does is philosophically interesting (i.e. the provision of accounts of cognitive capacities for AI programmers is a legitimate field of philosophical study). So, were it the case that these Churchlandy explanations were proven to be ideal on some top down account, there would be an 'explanatory' practice (AI programmers asking for 'explanations' of cognitive capacities) which was ignored by the philosophy of explanation.

To summarize the third reason, the idea is that there is a scientific practice which merits attention, and we have agreed in Chapter One that the philosophy of science can plausibly investigate science via a study of science (as opposed to investigating it via epistemology, or logic, of the philosophy of language). This bottom up style of philosophy is not the only legitimate style, of course, but it *is* a legitimate style. To crystallize the point, imagine a situation where, somehow, a top down theorist actually came up with the *correct* account of explanation. We can ignore the question of how we would recognize such an account. If there were a sustainable philosophical consensus on the matter, and if scientists themselves adopted the practice, then all three of our reasons for rejecting the top down route would be undercut — and this would be all well and good; the philosophical task would be much easier. However, if there were no philosophico-scientific consensus, the last two reasons would stand; the first would not, of course (since, although there would still be no consensus, we are assuming that there is a correct approach, however we would recognize it). The gist of the third reason would be that there is a practice — call it imperfect-explanation — which clearly differs from explanation proper. That practice still needs a philosophical account; the third reason for rejecting top down approaches holds that actual practice should be analysed philosophically, even if it is imperfect in important ways. The second reason shows that the top down account, though it explains explanation, does not explain imperfect-explanation, and hence that we must still examine the actual imperfect practice if we are to get a cogent philosophical account of it.

3.2.5 Two Top Down Responses

A top down theorist might wish to respond to these three reasons in two ways. Firstly, he may wish to make a countercharge against the relevant scientific methodology (in our example, AI methodology) in some specific way conditional on the actual state of affairs with respect to scientific (psychological) explanation. If it turned out that eliminativism was correct and any correct explanation of behaviour should simply bypass the propositional attitudes, then all the AI complaint shows is that the architectures upon which AI is being performed are inappropriate to the task. The proper task of AI, it then turns out, is to develop architectures that are compatible with the correct explanations in the new vocabulary (e.g. neuronally-based architectures). Then the correct explanations would be easily exploitable in the new architecture. It is obviously the case that a von Neumann architecture is likely to be inadequate for implementing models that don't exploit the propositional attitudes; hence the correct response of AI is to develop architectures that aren't inadequate in this way. In other words, the correct type of explanation will suggest methodological directions.

This top down theorist, then, would give the following rebuttals to our three reasons. Reason 1, the lack of consensus, is simply uninteresting; one account is right, and it is your responsibility to find it. Reason 2, the lack of ability to explain actual practice if it departs from the ideal, can be dismissed on the ground that the ideal is a normative concept. The criticism is, in effect, correct, but can be rejected because the purpose of top down explanations of explanation is to provide an account of what scientific activity *should* be performed. Scientists may well do all kinds of strange things, but that is their affair. Reason 3 is rejected on similar grounds.

The flaw in this response is clear given our discussion in Chapter One. In the first place — to continue with the AI example — it is ludicrous to suggest that AI projects should wait (a) until the disputes between the various luminaries in the philosophy of mind, Dennett, Fodor, Churchland and Churchland, Block, Putnam, Dretske, Searle etc. etc. are settled, and (b) given the settlement of that dispute, until psychologists have started producing explanations of the requisite type. The dispute in the philosophy of mind shows every sign of going on for decades, and, in the unlikely event that, say, the eliminativists are proved right, it

is not clear how long it will take even for a suitable psychological vocabulary to be agreed upon, never mind for an explanation to be produced.

In the second place, if a new architecture did have to be developed for AI, this would delay any current project even further. The discipline, which has had some success, is in place. It is all very well to try to persuade AI projects to take a more sound line psychologically, but if that entails a total, fundamental rethink, then it is unlikely that one would sound very persuasive. The most efficient way for philosophers to interact with AI is for philosophers to take note of what AI actually *does* and *needs*, as we showed in Chapter One.

In the third place, the top down reply assumes that the purpose of AI is to build people. This, as we argued in Chapter One, is false. AI attempts to program machines to perform tasks for which intelligence is generally required. Relevant parameters for any particular project will include the cheapness and the simplicity of the architecture. What any AI project will actually need is a well-known architecture which can be programmed easily. Hence, if I wish to build, say, a KBS for a corporation, then I should be as prepared to run it on a Macintosh as I would be to run it on a connectionist architecture. Even if the result of the discussion in the philosophy of mind were a clear indication that the architecture of the human brain was of a particular type, then all that indicates is that the processes and algorithms used in the human brain would fail to run with the same efficiency on any different artificial architecture, and hence, that there would be no point in using them on such architectures. The context of AI provides overriding reasons for requiring accounts of a particular nature, whether or not they are ideal in some philosophical sense or not.

Note also that the discipline of AI[1] doesn't really have much of an interest in the provision of psychological explanations (if psychological explanation is defined in a top down way as some particular practice such as giving neuronally-based accounts). AI requires that programs be built that run in real time — there is not much point using a computer as an investigative tool if you cannot get such programs. If the only way for AI to be of psychological interest entails that functioning programs can't be developed, then AI will not be of psychological interest. This top down view will tend to make the relationship between AI and psychology problematic. On the other hand, the bottom up view is much less uptight about what is psychologically explanatory. For instance, the conceptual

[1]This is arguable, but this consideration certainly applies to the subfield of KBS development.

models we will be discussing in Chapter Five are psychologically explanatory (we shall argue) even though their primary purpose is to enable the development of a type of industrial technology.

The second top down response to the arguments in this subsection would be to bite the bullet and say simply that AI, if its interests preclude the use of the actual, true, correct, ideal explanation of the desired behaviour, is therefore not interested in psychological explanations. What the AI community wants is not explanation, but description, or specification: it is not an explanatory discipline but an engineering discipline. This response is a more realistic one than the first, though still not decisive in the top down direction.

In the first place, note that, as a defence of top down accounts of explanation in science as a whole, this response must be generalizable. AI is not the only field which has been taken to be explanatory, but which is likely to be dismissed as non-explanatory on some top down views; others might include medicine and meteorology. All these disciplines have imperatives other than simple description. If this response is to succeed as a general defence of top down accounts of explanation, the considerations set out in the previous paragraph about AI must carry over to all other disciplines which are deemed non-explanatory counterintuitively. Let us suppose that they do, for the sake of argument: does the response succeed?

What the AI community really wants is answers to questions, and, moreover, answers with particular properties. Whether these are called explanations or descriptions seems to be unimportant, as far as progress in the discipline itself is concerned. However, philosophically, if it turned out that the answers provided in AI *were* acceptable as explanations on the bottom up view, then this top down response would fail to establish its claim, though equally this would be no particular proof of the *falsity* of the top down claim. All that would have been demonstrated is that the two approaches are incompatible in the absence of a top down consensus. We would simply have two differing accounts of explanation, under one of which AI dealt in explanations, which it wouldn't under the other. But it would also have to be noted that psychology won't necessarily issue the ideal explanations on the top down view, whereas the bottom up view at least is prepared to take the actual output of psychology (and the input of AI) seriously.

What the AI community (or at least that portion of the community with which we are concerned) is interested in is a model of the expertise to be used in a system, as described in Chapter Two. We will attempt to show that these models, on a bottom up view, do function as explanations of the expertise to be modelled by a system. In the next section, we shall sketch a bottom up view as an indication of the sorts of factor that are of interest in the purpose-relative evaluation of an explanation. I do not intend to argue strongly for any detailed position; what I want to suggest is that it is at least *prima facie* possible that an account of explanation can be given which respects local practices and is sensitive to the requirements of particular disciplines and projects.

3.3 A Bottom Up Account of Explanation

The bottom up view of explanation is likely to have a lot in common with pragmatic views. However, what constitutes a pragmatic account of explanation has been an area of dispute (van Fraassen 1980; Achinstein 1984; Kitcher and Salmon 1987; Lipton 1990). For our purposes, we can take Hempel's discussion of the pragmatic character of explanation as a pretty straightforward and defensible account.

> In a pragmatic context, we might say, for example, that a given account A explains fact X to person P1. We will then have to bear in mind that the same account may well not constitute an explanation of X for another person P2, who might not even regard X as requiring an explanation, or who might find the account A unintelligible, or unilluminating, or irrelevant to what puzzles him about X.
>
> Explanation in this pragmatic sense is thus a relative notion: something can be significantly said to constitute an explanation in this sense only for this or that individual.
> (Hempel 1965, p.425)

So a pragmatic account of explanation will need to specify, for a particular individual (or group of individuals), what will count as an explanation. This specification constitutes the pragmatic element of the account.

However, note that this does not in itself ensure the equivalence of pragmatic and bottom up accounts of explanation. The reason for this is that one could still

use top down methods to determine the pragmatic element of the account. For example, in van Fraassen's account, his relevance relation R does most of the pragmatic work (1980, p.143). The relation determines what answers will be suitably relevant to the problem to be explained. It is clear that one could, in a particular case, decide on a relevance relation by *a priori* thinking (i.e. top down), or by bottom up study of actual explanatory practice. Similarly, Achinstein's account relativizes explanations to audiences via the notion of appropriate instructions (1983, p.113), and again these could be drafted in a top down or a bottom up way.

Conversely, we note that it is possible — though not likely — that a bottom up account might result in, say, the discovery that the deductive-nomological account is the correct account of explanation. This could happen if study of all areas of explanatory endeavour revealed that they all actually used (successfully and exclusively) D-N explanation. Of course, this would still contradict the Hempelian account in that it would still be possible for the Hempelian account to be false in some unanalysed (or imaginary) areas. A more plausible 'coming together' of a bottom up account with a top down account might be some discipline-relative discovery such as, for example, explanation in *physics* is deductive-nomological.

The converse is not really important here; the main lesson is that we cannot simply choose an off-the-shelf pragmatic account and apply it, and be sure of having respected our bottom up intentions. What we should do is allow the account to apply to *any* putatively explanatory practice. This will have the effect of being very inclusive — however, if someone wants to claim that a practice is explanatory, the onus is on him to show how the practice is explanatory, and in what contexts. We need to know about these contexts, and what the basis for the explanation of the explanandum is. Once we know about this sort of item, we are able to judge the quality of the explanation.

Therefore the account of the explanatory practice should include details that enable us to make that judgement. The purpose of this thesis is not to deliver an account of explanation; hence we will not be supplying an exhaustive list of such details. However, details of the following would seem to be important for the judgement of the quality of an explanation. When we come to evaluate the explanatoriness of conceptual models in Chapter Five, we shall measure them against these yardsticks.

1. The value of the explanation should be relativized to the audience. The audience will require the explanation as a basis for scientific understanding, as discussed in §3.1.1. Hence the explanation should enable, as far as possible, the audience to achieve the type of scientific understanding that is characteristic of their particular science. In other words, a characterization of the audience's requirements is essential if the value of an explanation is to be assessed. Of course, these requirements should be characterized bottom up.

The articulation of the explanation should, all things being equal, involve a change of state for the audience. This requirement might be phrased as *the audience should come to understand the explanandum*, although there may be circumstances in which an explanation fails to achieve this goal. Basically, though, whatever account of understanding is given, what should generally follow is that the audience, after hearing/reading/apprehending the explanation, is nearer to being able to perform the action for which the explanation is required than it was before. The audience should be interested in an explanation of the explanandum in the proffered form, and furthermore the proffered form is a form which is appropriate for the audience's interests.

2. Generally speaking, the explanation should be true. However, this is not always the case; what is more important is that a good explanation should increase our understanding of the domain. Good approximations could be OK (Newtonian mechanics should be OK as a basis for explanations of the medium-sized physical world, for example, despite their being strictly false). Secondly, sometimes even good approximations don't do the trick. It may be the case that in some context, only a very crude and obviously false model will enable scientists to explain outcomes, either because the modelled system is too complex or chaotic to be accurately modelled, or because an accurate model would be computationally intractable (Morton 1993). Note how the notion of computational intractability automatically relativizes what is explanatory to a time — what is computationally tractable will depend on the current state of computational technology (e.g. some arithmetic problems would have been intractable using Roman numerals or systems without a zero). Thirdly, it might be the case that what is genuinely explanatory about an explanation turns out to be some fruitful metaphor (Hoffman 1980, 1985). An example here might be the explanation of how enzymes work by saying that they are catalysts. This is literally false, and not even approximately true. The operation of enzymes is explained, for many purposes, by the use of the *image* of a catalyst, a chemical

which speeds up a chemical reaction while itself remaining unaffected. Finally, it might be that arguments about truth and falsity get bypassed. As an example of this we might include psychoanalytic explanations. The basis for these explanations has long been disputed, quite rightly in my opinion, but sometimes at least the explanations (= diagnoses) do turn out to be helpful for the patient, in which case their value ought not to be viewed as somehow dependent on the resolution of a century-long philosophical dispute. Setting out precise limits for the allowability of literally false explanations is beyond the scope of this thesis, but there should be some leeway.

Basically, what is required in any account of an explanatory practice is that the claims made in the practice be evaluated for truth/approximate truth/fruitfulness, etc.. In the case where the explanation is not true, what the explanation does (i.e. how it provides understanding) needs to be set out very clearly.

3. It should be clear how the explanans relates to the explanandum. Certain minimal claims should be made about how the explanans results in the explanandum, and about the responsibility that the explanans has for the explanandum. We obviously want to rule out examples like the flagpole and its shadow (in the general case, while leaving it open that in some strange context, the usual explanatory order may be reversed). What I mean by saying that the claims need to be minimal is that it may turn out that a claim about the connection is stronger than required for the explanation to succeed for an audience; in that case the claim need not be made. For example, suppose United lose a football match, and I wish to explain the defeat for the purpose of picking next week's team. Then I may claim that Bannister is to blame, because his own goal was the *single cause* of the defeat. However, all I need to claim is that his presence was a contributory factor to the defeat. Or, for another example, I may explain someone's actions by their beliefs. All I need to show there is that the beliefs made a difference to the performance of the actions, not the stronger claim that a neural analogue of the belief has some causal connection to some appropriate set of macro-events in the real world. The explanation of the action by belief should not be dependent on the proof of the more controversial philosophical claim.

4. It should be clear exactly what the import of the explanandum is, i.e. the contrastive element (cf. Lipton 1990) should be clear, and furthermore the contrast should be maximally useful for the audience.

5. It should be shown how the account might fail to be explanatory (i.e. an explanation should be defeasible). We can phrase this condition as the condition that the explanation be *contentful*. For example, it should not be the case that *everything* is an explanation (Kitcher and Salmon 1987). Neither should it be the case that the audience can simply *decide* (in unprincipled fashion) what counts as an explanation (Lewis 1986).

6. Notice should be taken of what is explained in a particular context and why. For example, van Fraassen (1980) maintains that every explanation is an answer to a why-question. However, this would seem to be clearly false. Answers to other questions (e.g. 1, *How could this possibly have happened?* or 2, *How did this actually happen?*) would seem to be explanatory, but it would be difficult to convert these questions into why-questions. Van Fraassen himself has an abstract theory of why-questions (1980, pp.141-6), and such questions as (1) and (2) could be expressed in his theory. But it is, I think, stretching the point to say that his theory is an *analysis* of why-questions (and not an analysis of how-questions, etc.).[1] Furthermore, sometimes explanations are required which are not answers to questions at all. Physicists may want an explanation of the atom; linguists may want an explanation of medieval Spanish; Americans may want an explanation of the lbw law. In each case, though the ignorance of those who require the explanations may *prompt* questions, it seems to be clear that something more than answers to those questions is required. Recall that explanations should supply understanding; recall also that understanding is a relatively rich concept which should be distinguished from the narrow concepts of 'understanding-why' which some philosophers of science favour (cf Cooper 1994, n.2).

Point 1 takes on board Achinstein's (1983) point about the importance of an audience's understanding of an explanation — though note that we have still remained neutral over the process/product distinction here (recall from §3.1.2

[1]A why-question is defined by van Fraassen as a triple containing the topic (which, if the question is "why E?" is E), the contrast class (a set of other possibilities containing E, such that only E is true), and the relevance relation R. An answer A is relevant to the question when A bears relation R to the topic and the contrast class. So if I want an answer to the question *How could E possibly have happened?* then I am looking for an explanation of the occurrence of a freak event, whereas if I ask *How did E, as a matter of fact, occur?* then I want the actual cause of E. I may well want different answers for the two questions, even if E remains constant over the two cases. In the first case, what I want is an indication that the probability of E was non-zero (even if not very high); in the second case, what I want is the cause of E. Hence in the first case, the van Fraassen why-question is as follows. The topic is 'Pr(E) > 0'. The other member of the contrast class is 'Pr(E) = 0'. The relevance relation is 'derivable using probability theory'. In the second case, the topic is E, the contrast class is {E, not-E}, and the relevance relation is that of cause to effect. In other words, the account appears to be an analysis of questions in general, not why-questions in particular.

that Achinstein, unusually, favoured the *act* of explanation over the *product*). Point 2 directly contradicts the common requirement for explanations to be true (e.g. Hempel 1965; Lewis 1986) — although it leaves it open that, accidentally, all explanations turn out to be true. Point 3 in effect is asking for a specification of something like van Fraassen's relevance relation, although how that relation is specified should be a bottom up process. Point 4 is a common requirement (van Fraassen 1980; Lipton 1990). Point 5 respects Kitcher and Salmon's point (1987, p.314) that, if one is not careful, the relevance relation might turn out to be vacuous. We should show how the relevance relation is determined, and how a relation determined in that way will tend to bring instrumental scientific understanding to the appropriate audience. Point 6 is a requirement for a specification of something like van Fraassen's notion of a topic, or Hempel's explanandum. However, we remain neutral on the question of whether such topics/explananda are necessarily linguistic.

3.4 Causal and Non-Causal Explanation

The main task of this chapter is now completed, and a positive view of explanation has been sketched. The final issue with which we are concerned with respect to scientific explanation is that of causal and non-causal explanation. We shall begin by attempting to show that this is a genuine distinction in scientific explanation; then we shall look at a particular type of non-causal explanation, normative explanation. Thirdly we revisit causal explanation, and close with a discussion about program explanation; the idea there being to show various different ways in which an explanation can be causal. The ideas in this section will become important when we discuss the type of explanation provided by a conceptual model in §5.3.

3.4.1 Non-Causal Explanation

It is not uncontroversial to say that it is possible that explanations can be non-causal. Bas van Fraassen, for one, (1980, p.124) denies it. However, that claim is a rather general one, since he spends little time discussing when and how a relation can be causal, and seems, in effect, to define causal relations circularly as relations invoked in scientific explanation. However, other thinkers do have non-trivial accounts of causation on the basis of which they claim that there are no non-causal explanations. For example, Lewis's main thesis is that to give an explanation is to give some information about the explanandum-event's causal

history; he therefore is prepared to spend time reinterpreting apparent cases of non-causal explanation as causal (1986, pp.188-92). Salmon too has a non-circular account of causation, on which he claims that scientific understanding is the process of coming to see how an event was produced by the various causal mechanisms by which the world works (1984, pp.132, 242-59).

Naturally, these top down approaches to causal explanation do not sit very well with our bottom up premiss. The only ground we will accept for the assertion that all explanations are causal will be the empirical discovery that all explanations are in fact causal, or that the practice has built into it the assumption that the explanations it provides are causal. We will not lightly accept an account that insists on *a priori* grounds alone that scientific understanding is identical with appreciation of the causal antecedents of an event; recall our instrumental view of understanding from §3.1.1. All the types of understanding that we set out in that section need some philosophical explanation (cf §3.2.4) and even if we insist that the term 'scientific explanation' applies only to the sorts of explanation countenanced by Lewis or Salmon, there clearly is some other practice going on that is similar to explanation in the restricted sense, and which isn't necessarily concerned with causal relations to the exclusion of other types of relation.

In fact, there is a simple argument that shows that non-causal explanation does exist, an argument used by Peter Achinstein (1983, pp.235-7) and David-Hillel Ruben (1990, pp.218-22). The basic premiss for the argument is that nothing can cause itself (or at least that there are many things such that they do not cause themselves, and which are explanatory in the way required for the argument). Since the argument is sound, we may as well borrow Ruben's summary of it.

> Causation ... must be a relation between two distinct existences. Since there are cases of empirical explanation in which there are not two distinct (or even different) existences that figure in the explanans and the explanandum, it follows that there are some cases of non-causal explanation. ...
>
> In its simplest form, we can sometimes explain why some particular, a, has property P by identifying P with a property, Q, which a also has. ... Achinstein argues that identity explanations cannot be a species of causal explanation , since the having or acquiring of

> property P can't cause the having or acquiring of property Q, if P = Q. It makes no difference to my argument whether these identities are metaphysically necessary or contingent.
>
> Temperature = mean kinetic energy (for some temperature t and some mke m, having temperature t = having constituent molecules with mke m). I can explain a gas's having a certain temperature t by its constituent molecules having mean kinetic energy m, and I can explain a change in a gas's temperature by a change in the mean kinetic energy of its constituent molecules. We explain in these cases, not just by laws of the coexistence of two types of phenomena, but by property or type-type identities. This kind of explanation, relying as it does on identities, cannot be assimilated to causal explanation.
> (Ruben 1990, pp.218-9)

The existence of identity explanation — and it seems clear that the gas's temperature *is* scientifically explained as Ruben maintains — means that scientific explanation need not be causal. The standard sort of reply to this point is that there *is* a species of causal explanation going on here: the fact that the temperature has a certain value will result in various effects and it is these effects that are explained causally by the mean kinetic energy of the relevant molecules. This can be accepted, without compromising the main thrust of the argument which is that mke is explaining *temperature*, not the effects of temperature. The main point stands, which is that inter-theoretic reduction is often explanatory, and is itself not a sub-species of causal explanation. Hence non-causal explanations are possible.

3.4.2 Normative Explanation

Having established that there are non-causal explanations, the next question is what types of non-causal explanation there are. This is not a question that we need answer in full in this thesis; all we need to do is to set out the type of non-causal explanation in which we will be interested. That type is *normative explanation*.

An explanation is normative when it sets out the *norms* for performance of a particular task. This is clearly a non-causal type of explanation, in that norms do not necessarily cause events to happen — since they are norms, they can be

followed or not. Because norms are basically guidelines for action, clearly most normative explanation will be psychological (although one could imagine normative explanations of animal behaviour). Hence a practice will be defined by norms, and actual behaviour explained by showing how it conforms to norms.

For example, in football, the offside trap is governed by norms of successful behaviour. The idea of an offside trap is that, fractionally before a long through ball is played from midfield to the strikers, the opposing defenders run away from the goal they are defending, thereby leaving the strikers in an offside position, which wins the defenders a free kick. At a given signal, then, it is the defenders' job to run forward. Hence, the question "why did the defenders run from the goal they were defending?" can be answered with reference to the norms governing the offside trap. Their apparently strange behaviour (leaving the strikers alone in possession of the ball near their goal) can be given a normative explanation by giving their tactics. Note that the behaviour is explained whether or not it succeeded in conforming to the norms; offside traps are often unsuccessful, yet even so, the defenders' moving out is still explained by the norms of the offside trap. The normative explanation explains the behaviour by telling us what the *point* of the behaviour was. It provides a *justification* for the behaviour.

That example was not an example of a scientific explanation, but normative explanations can be found in science too. A large part of linguistics is devoted to the study of linguistic norms. In anthropology, norms govern a number of practices. Economics is largely concerned with the norms governing rational behaviour and attempts to increase marginal utility; ditto game theory.

The basic structure of a normative explanation is that the behaviour to be explained, B, whose purpose is the achievement of goal G, is shown to be an instance of a practice P. P is described by norms, which are shown to be the (or a) way to achieve G. Then B is explained by pointing out that it was performed to achieve G, and that it conformed to P, and P is the way to achieve G. So, in our example, we say that the defenders wanted to catch the forwards offside, so they ran away from their own goal — that's how you catch forwards offside.

Hence, normative explanation is generally appropriate in a context where it makes sense to assume some sort of design or purpose is at work. For example, in economics, the theory of perfect competition, despite being inherently less

realistic than the theory of imperfect competition, is argued to be of greater use (Friedman 1953). The theory relies on a small number of unlikely assumptions about rational individuals and their dealings in a society of like-minded others; it sets out norms of rational behaviour (e.g. all things being equal, a consumer will consume a good until the marginal utility of consumption of a unit of the good falls to zero). It seems pretty clear that such a view is untenable as a causal description of individual consumers in a society (Sen 1976-7; Simon 1983). Yet these norms do explain certain types of behaviour, both at the macro- and the micro-level. At the macro-level, they allow explanations on a national scale to be given, since, in theory at least, economies behave as if they are aggregates of individual consumers all of whom behave perfectly rationally. At the micro-level, certain types of behaviour, which conform to the norms of rational behaviour, are thereby explained — why did A do X? Because it was the rational thing to do. The norms of rational behaviour need not have been followed reflectively by A, only that A's behaviour conformed to them.

Another example of normative explanation might be the explanation of animal behaviour as governed by norms. For instance, the flocking behaviour of birds might be explained by the role it plays in defence (i.e. birds in a flock are much more likely to detect predators — since only one bird out of the whole flock need perform the detection — than a single bird), food gathering and possibly migration. This is a normative explanation — that is the point of having the whole flock move and behave in the way that it does. But these norms need be of no relevance to the direct causal explanation of the flocking behaviour. Realistic flocking behaviour of birds can be simulated by giving a group of artificial agents a small number of local norms and rules. Hence each agent reacts only to the behaviour of its near neighbours. Yet highly complex patterns of actual avian flock behaviour that seem, to the untutored eye, to involve an extraordinary degree of coordination between the individuals concerned, can be produced by these small sets of local rules. Clearly, this is not (necessarily) a causal explanation of actual avian flocking, since there is no evidence as yet that mechanisms of the appropriate type are instantiated in flocking birds. But what it does show is that the complex group behaviour can in fact be produced by aggregates of agents which only respond to local conditions. And this means that, until the causal explanation of avian flocking behaviour is discovered, we have no reason to assimilate the normative explanation to the causal.

A third example comes from linguistics; the linguistic norms appropriate for a natural language like English will explain utterances in English. But these clearly can't be the (whole) causal story, since relatively few utterances, particularly in spoken English, conform wholly to those norms. The norms of English provide a 'goal' for speakers of English; most utterances will fall short of that goal by not being grammatically correct, etc., but the norms are still explanatory in a number of circumstances.

A fourth example might be the use of a normative explanation as *justificatory*. What a normative explanation can do is to set of the (or a) 'right' way of performing a task. So, what the explanation might say would look like "given input I, you perform subtasks S, T, ..., and then you get output O." This would justify someone's actions who had performed the task, having got output O from input I, even if that particular series of subtasks had not been followed exactly.

Note that these examples show that normative explanations need not be causal explanations too; in this respect, they are different from Philip Pettit's notion of a *normalizing explanation* (1986). A normalizing explanation, for Pettit, though normative — making ineliminable reference to norms — is also causal (1986, p.21). The norms are to be taken as providing a hint as to the causal antecedents of the behaviour. This is certainly possible (see §5.3 below), but not necessary for a normative explanation. For example, if Pettit's normalizing explanation were the only type of normative explanation, the 'explanation' of bird flocking behaviour would not (as yet) be an explanation, since it has not been proven that such norms are causally active in birds. But it seems clear that flocking has been explained (i.e. how it is possible in the absence of occult processes has been explained) by the computer simulations.

3.4.3 Causal Explanation

Finally, we move on to causal explanation. A causal explanation is quite straightforward. We have a causal explanation of the explanandum when the explanans is the (a) cause of the explanandum. The explanation must make clear which property of the explanans is causally relevant. For example, take two billiard balls colliding. The first was motionless before the second hit it. The second billiard ball is the cause of the first billiard ball's ending up in the pocket. The explanation is not yet complete without specifying which property of the

ball is the causally relevant one.[1] After all, the second billiard ball is 93,000,000 miles from the Sun, yet that is not a causally relevant factor. What is causally relevant is the momentum of the second ball.

Note that this account is independent of which particular account of causation is preferred. It will work with van Fraassen's wide notion of cause, or Salmon's tightly defined notion, or even an interpretivist view of causation (Child 1994). However, there is a problem with causal explanations, which has been set out by Jackson and Pettit (1988, 1990, 1992).

The problem is a simple one; if one is giving a causal explanation of an effect, then one way to give it is to give the causally efficacious property of the cause. For example, if an explanation is required of why the glass broke, the causally efficacious property is that of the glass's having a certain molecular structure. Now the problem arises if the further assumption is made that giving a causal explanation is to give the causally efficacious property. Because it seems that there will be apparent cases of causal explanations which do not give the causally efficacious property. For example, one might explain why the glass broke by saying that it was fragile. The purported explanation is true; on the other hand, the fragility is not the causally efficacious property (Jackson and Pettit 1990, pp.108-11). Recall that we have defined a causal explanation as giving a causally relevant property of the explanans; the further assumption in our terms, then, would involve the assertion that the only way for a property to be causally relevant is for it to be causally efficacious.

It turns out to be too restrictive a notion of causal explanation if causal relevance equals causal efficacy. Is there an independently motivated account of causal relevance that will get round Jackson and Pettit's problem? Can we have a ground for saying that the fragility of the glass is causally relevant to its breaking?

3.4.4 Program Explanation

We can have such a ground, with Jackson and Pettit's notion of a *program explanation*. In such explanations, the causally relevant but non-efficacious

[1]It might, of course, be complete if what is required by the explanation is an indication of which body caused the ball to move, as opposed to an indication of how the second ball caused the first to move.

property is relevant precisely because it ensures that there will have been one of a group of causally efficacious properties realized in the explanans.

> The realization of the property ensures — it would have been enough to have made it suitably probable — that a crucial productive property is realized and, in the circumstances, that the event, under a certain description, occurs. The property-instance does not figure in the productive process leading to the event but it more or less ensures that a property-instance which is required for the process does figure. A useful metaphor for describing the role of the property is to say that its realization programs for the appearance of the productive property and, under a certain description, for the event produced. The analogy is with a computer program which ensures that certain things will happen — things satisfying certain descriptions — though all the work of producing those things goes on at a lower, mechanical level. (Jackson and Pettit 1990, p.114)

Hence the fragility of the glass programs for a molecular structure such that the glass will break when struck. Knowing that the glass is fragile does not tell us which molecular structure we have — we know that there is a large, probably infinite, series of molecular structures which it could have given its fragility, but we do not know which one. The fragility is causally relevant, though non-efficacious, since the glass's being fragile will ensure (make probable) that it has a molecular structure which *will* be causally efficacious in destroying the glass when struck. Hence the fragility will causally explain the glass's breakage, despite not being causally efficacious.

Note that we need not know which property is causally efficacious in such cases; we need not know even roughly. All we need to know is that the program property will make probable the claim that there will be some causally efficacious property of the required type. Further, it need not be the case that there actually is some causally efficacious property. If there was a downward sequence of levels of explanation (e.g. psychological, biochemical, chemical, physical, atomic, subatomic, ...), then a program explanation would tell us how the explanatory property of the explanans programs for some property at a lower level, which would itself program for the explained effect, and so on. Hence, there is nothing in the program account to worry anti-realists about causation.

Indeed, since we do have causal explanations, and since we reach causally efficacious properties so very rarely (if ever), unless there are other ways to be causally relevant to an effect, we must have program explanations extremely frequently in science. For example, there will be a number of explanations for the various effects of the comet Shoemaker-Levy 9 on Jupiter; it is inconceivable that any will be in terms of particular atomic particles (say) of either body. This is strong circumstantial evidence for the usefulness of program explanations.

Chapter Four: Psychological Explanation

In this chapter, we will examine the idea of explanation in psychology. Because we have taken the view that the value of an explanation is relative to its context, it may well turn out that there will be aspects of psychological explanation which depend, not on properties of explanation as such, but on properties of the particular types of explanation used in psychology. The explanations in which we are interested, conceptual models, are information processing models, and so we should examine the general explanatory pedigree of information processing models in psychology. In particular, we shall attempt to distinguish a range of ways in which such models can be explanatory. This will establish the possibility that conceptual models can be explanatory of the expertise they model; we will, of course, still have to examine such models and their context of use very carefully to establish whether or not they actually are explanatory (this will be the business of Chapter Five).

The first thing we should note about the psychological explanations in which we are interested in this thesis is that they are specifically scientific explanations. It may be that our general discussion of scientific explanation will generalize to the non-scientific case. Or, it may be that scientific psychological explanation and non-scientific psychological explanation are continuous with each other. In either event, our ruminations may well apply to ordinary, common-sense folk-psychological explanations (such as *he wants South Korea to beat Germany because he always goes for the underdog*). On the other hand, if there is a discontinuity, or if our discussion will not generalize, then our claims will not apply to folk psychology. But we will leave that issue unaddressed — when we talk of explanations, we shall only be referring explicitly to explanations in scientific psychology.

4.1 Computational Models in Psychology

What is noticeable in scientific psychology is how often computational models get used for explanatory purposes. Our aim in this section is to justify the practice of explanation using such models. This will not, of course, entail that *conceptual* models in particular are explanatory in the context in which they are typically used. The remainder of this chapter can then be devoted to examining some properties of computational explanations.

The first question we have to ask is why computational/information processing models are used at all. The answer to that is, I think, that computational models allow a position that seemed impossible prior to their introduction. Psychology, being a science, involves a number a characteristically scientific imperatives: the maintenance of objectivity, the repeatability of experiments, the discovery of mechanisms. That is not to say that the adoption of these imperatives is either necessary or sufficient for a discipline to be given the status of 'scientific', only that these are fairly natural imperatives for a putatively scientific discipline to adopt. Naturally, such imperatives are selected over time for the successes they bring (epistemologically by producing fruitful research programmes; pragmatically by bringing in research funding).

Because of these imperatives, the use of folk psychological techniques is effectively ruled out. Asking people why they do things, for example, would be ruled out by all three imperatives: objectivity would be replaced by subjectivity; such questioning would not be strictly repeatable unless performed under laboratory conditions; folk psychological questioning does not lead to the discovery of mechanism, as we know from millenia of the practice. The obvious candidate for a mechanism would be neurophysiological, since we know that there are strong and interesting connections between brain and mind. However, it has long been clear that, since both brain and mind are fearsomely complex, it will be a very long time before a great deal of important psychology can be deduced from the study of the brain — after important inroads in the 1960s, with the work of Hubel and Wiesel, and Marr, for example, the results have not come thick and fast. The only strictly objective psychological alternative seemed to be behaviourism, whose vocabulary was too attenuated to provide much of an advance.

Computational models provide a middle way between these alternatives by focussing on the structural properties of the mind, abstracting away from the details of what the mechanisms actually were in order to concentrate on their properties (cf. Deutsch 1962). The idea is that a mechanism can be described more or less abstractly. So, for example, some mechanism, horribly complicated at the neural level, might be described at a more abstract level as *recognizing an object as a person by comparing the shape of the object to an archetype*, say. This, although not absolutely specific, would at least be empirically testable, and may suggest a class of mechanisms.

What do we get with such an information processing mechanism? Well, if the mechanism, described in a model, is consistent with the behaviour, we get predictive power. If the terms of the model make sense in the context of behaviour, we get ways of talking about the intermediate stages of cognition in between stimulus and response. For example, if we had a model that decomposed behaviour B into a series of subprocesses B_1, ..., B_n, it is legitimate to ask about the significance of the B_1, ..., B_n. Psychological evidence can be adduced to support the postulation of these subprocesses, for example, by checking the decomposition of the model against psychological indicators such as reaction times, characteristic error patterns, etc.. If the evidence suggests that these subprocesses also have predictive power (i.e. that the decomposition of the model is significant, not merely its input/output structure), then we can say that we can understand the global behaviour in terms of the simpler behavioural patterns postulated by the decomposition (these simpler patterns may include memory recall patterns, face recognition, word identification, etc.).

Note that, in the context of AI, these simpler mechanisms are likely to be easier to implement in a machine than a piece of behaviour globally described (e.g. identifying computer faults, reading aloud, etc.). If the mechanisms underlying the decomposed parts of the model are understood, then the model as a whole will suggest to the psychologist how the mechanism underlying the global behaviour might be instantiated. If psychologists can agree about a certain set of components (i.e. simple behaviour patterns, construed either as unanalysed IO patterns or as actual mechanisms) of such models, then those components might constitute a set of psychological primitives, categories of primitive behaviours for human psychology. In all these ways scientific purposes will be served, and therefore, on our instrumental definition of understanding (§3.1.1), scientific understanding achieved. In that sense, then, by providing routes to such understanding, computational/information processing models can be seen as explanatory.

Such models are instances of functional explanations in Robert Cummins' (1975) sense, and are associated with the explanatory strategy of analysis.

> ... the analytical strategy proceeds by analyzing a disposition *d* of *a* into a number of other dispositions d_1 ... d_n had by *a* or components of *a* such that programmed manifestation of the d_i results in or amounts to a manifestation of *d*.

(Cummins 1975, p.186)

Here 'programmed' just means organized in a way that can be specified by a program. Cummins is clear that this is a common and fruitful strategy in psychology.

> Perhaps the most extensive use of the analytical strategy in science occurs in psychology, for a large part of the psychologist's job is to explain how the complex behavioural capacities of organisms are acquired and how they are exercised. Both goals are greatly facilitated by analysis of the capacities in question, for then acquisition of the analyzed capacity resolves itself into acquisition of the analyzing capacities and the requisite organization, and the problem of performance resolves itself into the problem of how the analyzing capacities are exercised.
> (Cummins 1975, p.187; cf also Cummins 1983, pp.28-51)

Cummins provides a robust defence of the analytical strategy as explanatory in general terms (1975; 1983), and says much that is persuasive. However, our bottom up view means that we should look at the operation of the particular types of model in which we are interested in context, and judge their explanatory value finally on that basis (Chapter Five).

Having established that computational models can be explanatory in psychology, in the remainder of this chapter we shall discuss various ways in which such models explain.

4.2 Levels of Explanation

When a model is being used for explanatory purposes, what will be explanatory are various properties of the model. The question now arises: which of the properties of the model are explanatorily relevant? A computational psychological theory will have a number of elements to it. There will be a set of primitive commands, perhaps, together with a conceptualization of the faculty to be explained, a conceptualization of the expected IO behaviour, and, if the explanation runs, a program, a computer on which it runs and a visualization of the program's performance on the computer. The question is: which of these

elements of the theory are to be counted as having explanatory force and which not?

For example, suppose I have a machine learning (ML) program which examines a collection of data and induces rules to connect various data items, associating a value with each rule on the basis of how well-confirmed that rule is in terms of the number of antecedent/consequent pairs of data which confirm the rule and the number of pairs which are anomalous. Suppose too that this program is intended to be psychologically explanatory. Then there will be a number of aspects which can be taken as explanatory or not, as the case may be. For example, the calculation of the confirmation value might involve some relatively complex calculation producing a real number between -1 (analytically false) and 1 (analytically true). Is this calculation part of the explanation? If it is, is it intended to correlate closely with some calculation that people actually use in induction, or just to correlate very broadly (perhaps the numerical values map onto a small set of certainty values, such as {yes, no, maybe}), or is the formula just an *ad hoc* measure that is basically intended to cut out erroneous output. The program may use data structures such as lists; is this fact explanatory? A list might be used simply because the program is written in Lisp and lists are more naturally dealt with in Lisp than in, say, PASCAL. If the use of list structures *is* explanatory, then is the fact that these lists are searched by the program from left to right instead of right to left part of the explanation, or just an artefact of the program? The lists are ordered before being searched: is *this* fact part of the explanation, or is that just to speed up the search? The program will be being run on a certain type of computer; is this fact explanatory? Usually not, but I may be claiming that the brain has a certain architecture which can be modelled in some way by the computer (e.g. if I were running a connectionist system).

4.2.1 Marr's Three Levels

All these possibilities suggest that some scheme could be developed for classifying theories by what is explanatory in them and what is not. Marr (1982) first addressed this problem by suggesting three *levels* at which a computational psychological explanation could be couched. The three levels are:

Level 1: An abstract formulation of what function is being computed by the explanatory model/system, and why;

Level 2: An algorithm for computing such a function; and

Level 3: An architecture for realising such an algorithm.

The level 1 explanation will specify which function is being computed in terms of IO relationships, but will contain no detail about how the function is computed. So in a functioning program which is intended to be explanatory, what is explanatory is the output of the program given the input (i.e. the set of input/output pairs). The fact that the program performed the particular set of operations that it did in the order in which it did is not intended to be significant. Neither is the fact that the program had the particular representational format that it had. At level 2, that sort of information (both the operations performed and the representation used) is important, but the details of the way that the operations are embodied in a functioning machine are not. At level 3, all this information is significant.

So, for example, one might use our ML program as an explanation by saying something like: the program models inductive practice by examining a collection of data, inducing rules to connect various items of the data, and associating a value with each rule based on how well confirmed it is (in terms of the number of data items of which the antecedent and the consequent of each rule are both true, and the number of anomalies, where the antecedent of the rule is true and the consequent false). That would be a level 1 explanation — the explanation tells us *what* the program does without telling us *how* it does it. There could be two different rule induction programs, which, for example, might demand the data be given in radically different forms, but which would get the same level 1 characterization. One might take a symbolic form, statistically inducing 'rules' or dependencies between attributes of data objects; the other might be a connectionist system which systematically alters the weightings of the connections between the nodes to express the likelihood of a dependency. These programs would clearly work in different ways; even so, the level 1 explanations would remain constant.

On the other hand, the explanation of the program might go into a little more depth, and discuss how the data for our machine learning program have to be formalized in a particular way, with, say, a complex datum representing each real-world object under study. Each of these data will contain a number of *slots*, each of which contains various bits of information. If the program were to

attempt to induce rules about the edibility of mushrooms, it might be given a series of objects like

```
mushroom-256
colour: red
edibility-raw: n
edibility-cooked: y
pungent: y
bole-shape: round
...
```

to deal with, and will perhaps be able to induce rules such as

```
all x. IF colour(x,red) THEN edibility-raw(x,n)
```

on the basis of statistical regularities in the data. Then the explanation will go into the statistical tests that the program will perform on the data, and into the method of calculating the value it associates with each rule. This sort of explanation gives the algorithm which the program uses, and is an explanation at level 2. With an explanation at this level, it might be the case that I had two versions of the program, one written in Lisp, which processes list structures, and one written in PASCAL, which does not possess any integral list structures. In those circumstances, the calculations — which remain the same — and the manipulation of the data might well both have to be done in different ways, entailing that the two algorithms used in the Lisp and the PASCAL programs would be different. The result here would be that two different level 2 explanations would have to be given for the same program. Finally, the explanation can go into the particular details of the implementation of the system, and will discuss how the information is carried around the chips of *this* very machine, so, for example, if I ran the program on my Macintosh IIsi, and again on my Sun workstation, there would be different explanations for the same algorithm.

Now, what will have been noticed is that there is a way of describing the model — which I have termed the *program* — which does not seem to be equivalent to any of the levels. Recall that we were able to talk of there being different programs equivalent at level 1, and of there being two versions of the same program at level 2 (and *a fortiori* at level 3). Furthermore, as we shall discuss in §4.2.2, the aspects of the model described at the three levels do not seem to be as clearly explanatory, in the context in which we are interested, as we would like. Recall from §3.3 that we required of an explanation that it be appropriate for the purposes of the audience, among other things. The audience we have in mind in this thesis is the developers of KBSs. Can we say that one of the three levels will

be appropriate for such an audience? There would seem to be good reasons why a KBS developer would not be satisfied with an account of expertise at either level 1, 2 or 3.

Level 1 explanation in the context of KBS development is not likely to be very explanatory; a specification of the IO structure would certainly be useful, but more information would be required to enable the system actually to be *built* — remember it has to work in real time, and provide explanations for the user, as well as compute the right output. The simple IO relation would not give any help to the knowledge engineer as to how the output could be *computed* given the input (I am assuming that the domain was sufficiently complex that the inputs and outputs could not, for reasons of memory limitations, be represented as a look-up table). Further, the user of the system would not be content with the system's own explanations of its output if it gave no indication of how the output was arrived at. Level 2 might well be adequate — an algorithm would make programming comparatively trivial. Here, though, there is the problem of knowledge acquisition (KA). There is no known way for extracting the algorithms used by experts in their deliberations. Hence, although such an explanation could be useful if we had one, there would be extreme difficulty in obtaining one in most circumstances. Added to which, it might be the case that an algorithm that ran efficiently on a neural architecture was not so useful on an artificial computational architecture. The same KA considerations apply to level 3, of course. For all these reasons, we need to go beyond Marr's analysis and discover a new level at which an explanation in this context should be pitched.

4.2.2 Explanation at Level 1.5

There certainly seems to be an exploitable gap between levels 1 and 2. Consider the example of the ML systems developed above. Let the four systems discussed be called CONNECT (the connectionist system), SYMBOLICP (the symbolic system programmed in PASCAL), SYMBOLICLM (the symbolic system programmed in Lisp and run on the Macintosh) and SYMBOLICLS (the symbolic system programmed in Lisp and run on the Sun). There is clearly an interesting level of what is going on which I have termed the *program*, where we want to say that CONNECT is one system, while SYMBOLICLM, SYMBOLICLS and SYMBOLICP are three aspects of another system. At this level, we want to group together systems which use a similar *method* to get results. So, in this example, the three systems all have in common that they

describe objects in a slot/filler notation, and then use a particular set of statistical formulae, or methods, based on the relative frequencies of covarying data, to induce universally quantified rules about the domain. Yet at level 3, all three systems are different; at level 2 we have three systems, CONNECT, SYMBOLICP and a system of which SYMBOLICLM and SYMBOLICLS are aspects; while at level 1, we have that there is only one system, of which they are *all* aspects. How can we define an extra level where we get the desired granularity of explanation?

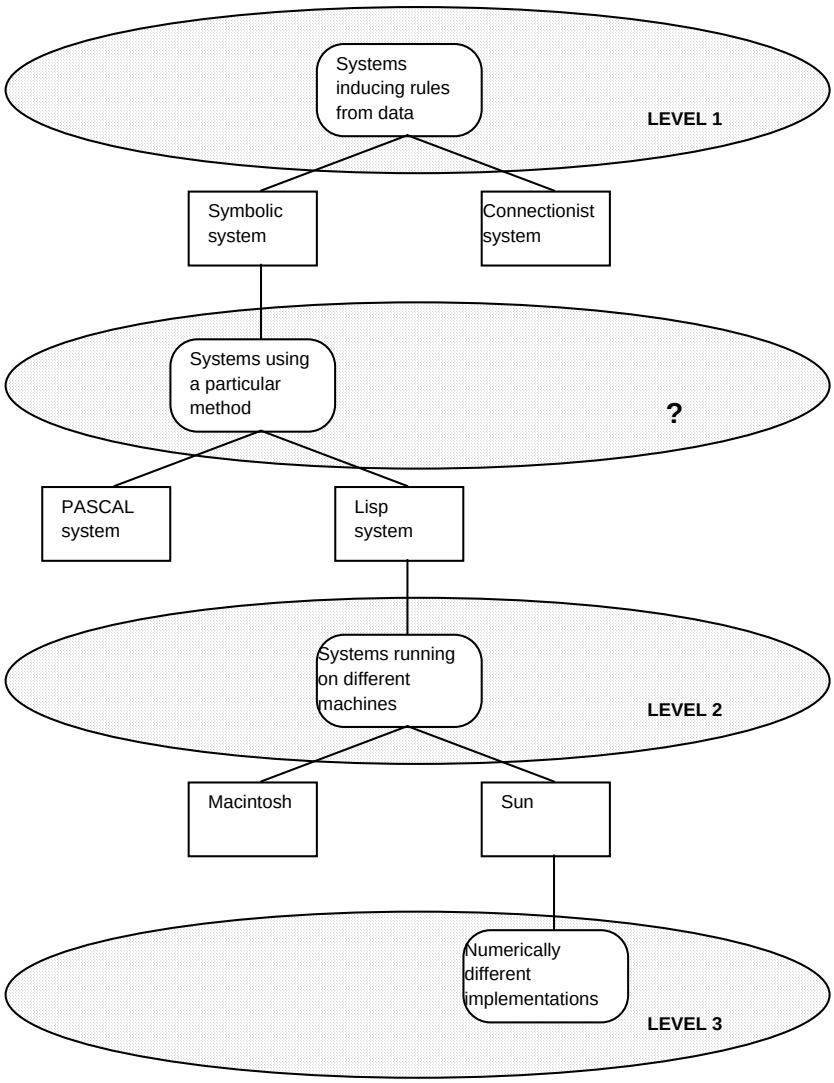


Figure 4.1: An Intermediate Level of Explanation

Let us examine one such candidate for a further explanatory level, one suggested by Christopher Peacocke (1986). Peacocke formulates the intermediate level as follows.

> Consider a subject's ability to pronounce words he sees written on a sheet in front of him. We can fix on a given function described at Marr's first level, a function from inscription-types — words, say — to sequences of phonemes. One can conceive of three quite different bodies of information which might be drawn upon by the algorithm which computes this function. There might (a) be a listing which specifies for each complete inscription-type the corresponding sequence of phonemes. ... There might (b) be an algorithm which draws upon information about the pronunciation of sublexical syllables, and computes the sequence of phonemes for the whole word from information about phonemes corresponding to its parts. A third possibility (c) is that there is an algorithm which draws on information about the pronunciation of a proper subset of the given class of whole words, and extrapolates from their pronunciation ... .
>
> These characterizations of the information drawn upon are not descriptions at Marr's level 2: they do not specify the particular algorithm employed. ... [They] are not at Marr's level 1, either. The same function (the 'function-in-extension') ... may be computed in cases (a)-(c). ... The pattern of one-many relations between Marr's levels is preserved when we insert level 1.5. One and the same function may be computed drawing on different bodies of information. One and the same body of information may be drawn upon by different algorithms.
> (Peacocke 1989, pp.111-2)

So equivalence at Peacocke's new level 1.5 will depend on the two systems *computing the same function* by *drawing on the same bodies of information.* This notion of 'drawing on the same information' is one that will need to be made more precise; nevertheless, there is enough there to enable us to see that level 1.5 is going to be genuinely distinct from Marr's three levels. The level is not equivalent to level 1, because more than the mere specification of the function is required; neither is it equivalent to level 2, because the explanation will involve a *class* of algorithms. Hence, in terms of our ML example, we may have found a

level at which CONNECT and the three symbolic systems may be distinct, assuming that the connectionist system did draw on different information.[1]

The question is, then, is this level the level we need? What sort of information is required by KBS developers?[2] For example, one might insist in model-based KBS development that a theory's ontology correspond to the ontology of the object system (i.e. the expert's expertise), for the reason that the target KBS must provide comprehensible explanations to the user. Peacocke's own pronunciation example implicitly accepts this possibility. The characterizations of the three classes of algorithm he mentioned make various assumptions about the speaker's phonemic ontology. Clearly it will not be the case, typically, that the speaker will have an overt or explicit ontology — most pronouncers of English words do not know what the word 'phoneme' means, of course. However, the linguistic researcher can home in on the correct ontology via some simple tests — seeing which syllables get the same pronunciation and when, for example (Peacocke 1989, pp.112-3). Another example: in medical diagnosis, the same body of information may be used in a number of possible ways of diagnosing some sort of disease. This body of information might well be, for example, the information available from a series of medical testing instruments. However, a model of a doctor's diagnostic expertise would hope to distinguish between the various ways in which she uses that body of information. For example, does she abstract from the raw data, and, if so, what are the categories available, and what is the mapping between raw data and the categories? Hence, a model of medical diagnosis could be such that it not only specifies that the doctor uses, say, information about the patient's temperature in her attempt to isolate his malady (as opposed to, say, information about the species of the microbes in the bloodstream); it will also specify "how many" temperatures there are (three: {low, normal, high}? five: {very low, low, normal, high, very high}? ten: {below 34, 34, 35, 36, 37, 38, 39, 40, 41, above 41}?).

---

[1]That assumption, of course, need not be made. Peacocke himself is quite clear that level 1.5 is likely to group together symbolic and connectionist systems (1989, pp.122-5). However, this is not a serious problem — the example was for purposes of illustration. The really serious problem would occur if level 1.5 could not be used as a level of explanation for the purposes *we* are interested in.

[2]One point that we should note here is that in the context of KBSs, the information drawn upon by a program is the knowledge stored in the KB — i.e. a (variable) store of information separate from the inference engine. It is not, however, a necessary requirement that the information used by any system in the scope of Peacocke's discussion be represented independently by the system. Information can be hard-wired into a system.

The question is whether Peacocke's level 1.5 covers this sort of information; there is, I think, an important ambiguity in his account. Consider his level 1.5 account of ways of computing the pronunciation of words quoted above. One of his level 1.5 characterizations is: 'an algorithm which draws upon information about the pronunciation of sublexical syllables, and computes the sequence of phonemes for the whole word from information about phonemes corresponding to its parts' (Peacocke 1989, p.112). Now, this characterization does not distinguish between the following three algorithms.

[1] An algorithm that goes directly from a syllable, considered as a unit, to a sequence of phonemes. This algorithm might contain the following mappings:
cat -> *kat*
catt -> *kat*
kat -> *kat*
katt -> *kat*

Here, each mapping realises a piece of information about the syllable in its left hand side (e.g. that the syllable 'cat' is pronounced *kat*).

[2] An algorithm that recognized possible intermediate equivalence classes of syllables. This algorithm might contain the following mappings (where [katt] is the equivalence class of syllables pronounced like 'katt':
cat -> [katt]
catt -> [katt]
kat -> [katt]
katt -> [katt]
[katt] -> *kat*

This algorithm draws upon pieces of information such as that the syllables in [katt] are pronounced *kat*. Note that if that one piece of information went missing, the system would fail to generate a pronunciation for any of the syllables in the equivalence class; this gives us a handle on a possible empirical test to decide between [1] and [2].

[3] An algorithm that, like [1], goes directly from syllable to phonemes, but is selective, excluding rare syllables or syllables which do not actually crop up in English. This algorithm might contain the following mappings (explicitly excluding the mapping for 'katt'):
cat -> *kat*
catt -> *kat*
kat -> *kat*

Here, given that 'katt' does not appear in English, [1] and [3] would have the same IO relations (and therefore would be equivalent at level 1). Indeed, for a given IO transition performed by both algorithms, the same information would be used. The difference between [1] and [3] is that [3] lacks a piece of information that [1] has, since [3]'s input domain is smaller than [1]'s.

So whereas [1] and [3] simply store the mapping between syllables and phonemes, [2] recognises that generalizations can be made about classes of syllables. [2] might therefore be more flexible; it might, for example, facilitate the assimilation of new syllables into the scheme. But in each case, we note that the algorithms might fairly be described as drawing upon information about the pronunciation of sublexical syllables, and (ultimately, although we haven't suggested how they do this) computing the sequence of phonemes for the whole word from information about phonemes corresponding to its parts.

Or, as another example, when Peacocke mentions a class of algorithms which 'draw on information about the pronunciation of a proper subset of the given class of whole words, and extrapolate from their pronunciation', he does not appear to distinguish between different algorithms which have different subsets of canonical words, or different algorithms that use distinct similarity measures to make the extrapolation. But one can see that such distinctions (still above level 2) might also be explanatorily interesting.

For Peacocke, the idea of an algorithm drawing upon information

> ... requires that a state which carries the information drawn upon is causally influential in the operation of the algorithm ...; indeed it requires that the algorithm ... produce states with the content they do in part because of the content of the information-carrying state.

(Peacocke 1986, p.102)

This is ambiguous as to how to specify the information carried. Peacocke appears to be interested in *bodies* of information rather than *states* of information: information *about*, in his example, the connection between phonemes and syllables, etc.. In the medical example we gave above, what we have at level 1.5 is information *about* temperature, but is discussion allowed about how temperature has to be be represented? A level 1.5 explanation of the diagnostic process would involve saying that the doctor uses information about the patient's temperature, and so on, to make the diagnosis — the relevant contrastive class is the class of algorithms where temperature is *not* necessarily used in the diagnosis. But it is clearly relevant for KBS purposes to specify how temperature is represented. If the 'correct' representation must be from the class {low, medium, high}, then it may be possible to devise a program which goes from the numerical representation to the diagnosis; however, if the 'correct' representation is numeric, it may be impossible to devise a program which goes from the abstracted representation to the diagnosis (i.e. the thermometer gradations are too coarse-grained for effective use in problem-solving). Is there a level 1.5 distinction between the two ways of representing temperature?

There is clearly an ambiguity here at the least. The question is whether *temperature(john,41)* and *temperature(john,fever)* are level 1.5 equivalent or not. Clearly, *different* information is carried by the two expressions (the first is more specific than the second). On the other hand, each expression might plausibly be seen as *conveying information about John's temperature*. In that case, two algorithms, one of which used expressions of the first form, and the second of which used expressions of the second form, could be seen as *drawing on the same body of information*, viz. information about John's temperature. At different points in his papers, Peacocke seems to be adopting different points of view on this issue. This is understandable, given that his main purpose in the papers was to establish a contrast with level 1 and level 2 accounts. However, here we have accepted his thesis that there are intermediate levels between 1 and 2, and therefore we need to resolve the ambiguity.

This leads us to suggest a new level, level 1.6.

4.2.3 Explanation at Level 1.6

We will suggest that Peacocke's work has uncovered two distinct levels of explanation, which we shall term level 1.5 and level 1.6. Level 1.5 is as defined in our long quote from Peacocke above; our interpretation of him is that he does not mean to distinguish between accounts with different representational formats. Level 1.6 is as level 1.5, with the additional constraint that different representations of the same body of information must be distinguished. We know that representational format is important at level 2, but since representational format is not on its own definitive of level 2, we are free to speculate that there are interesting levels above level 2 where we can take representational format into account. Hence the two expressions *temperature(john,41)* and *temperature(john,fever)* would be distinguished at level 1.6 and not at level 1.5. Our three pronunciation algorithms [1], [2] and [3] would similarly be distinguished at level 1.6 and not at level 1.5. Note that this definition of level 1.6 preserves the pattern of many-one relations between the levels: a level 1 explanation subsumes a class of level 1.5 explanations, a level 1.5 explanation subsumes a class of level 1.6 explanations, a level 1.6 explanation subsumes a class of level 2 explanations, and so on. Note also that we do not commit ourselves to the claim that Peacocke in his discussions of (his) level 1.5 is necessarily discussing (our) level 1.5 and ignoring level 1.6 — indeed, sometimes Peacocke writes as if it is (our) level 1.6 in which he is interested. All we claim is that Peacocke can be taken at various times to be writing about each of our levels, that the two levels are indeed distinct, and that each of the two levels can be explanatory.

How can we make this definition precise? One way would be to note that what makes the difference between the various algorithms (strictly, classes of algorithms) is the ontology they use. So, whereas level 1.5 insists that we should draw on information about body temperature, level 1.6 adds an account of the temperature ontology (i.e. are temperatures real numbers, natural numbers, the set {low, medium, high}, etc.?). Level 1.5 insists that we draw on information about phonemes associated with particular syllables. Level 1.6 adds a specification of which syllables there are, and how various *classes* of syllables which can be treated as equivalent in this context can be isolated. Classes of syllables are clearly ontologically distinct from syllables themselves, and therefore level 1.6 makes an ontological distinction. However, information about such classes is still information about which phonemes are associated with which

syllables (since this is the basis of the equivalence relation), and therefore level 1.5 as we have defined it makes no such distinction. Even more clearly, level 1.5 has no ground for making distinctions between our algorithms [1] and [3], which differ only in the syllables that they take as constitutive of the language.

It will be noted that level 1.5 is rather less precise than level 1.6 — which is not a criticism of it, of course, since less precise accounts are often of use. Note that this lack of precision is inherent in level 1.5. This is not because the levels get less precise as we ascend through them. Level 1 (functional equivalence) is precise, since input/output pairs are very precisely specified. On the other hand, level 2 is imprecise, because of the complete lack of any agreed identity conditions for algorithms. Hence the precision of level 1.6 and the comparative lack of precision of level 1.5 follow from their definitions, and not from their relationships with the other levels.

This imprecision suggests another way of resolving the ambiguity, which we can call the *extensional move*. The idea is to make level 1.5 precise. We can represent the information that the algorithm draws upon as a set. So the level 1 account is a relation R, which is a set of input-output pairs. The level in between levels 1 and 2 is a pair, whose first element is R, and whose second element is a class of expressions which may be drawn upon. Hence algorithm [1] would look something like this:

> <{ ... <cat, *kat*>, <catt, *kat*>, <kat, *kat*>, <katt, *kat*>, ... }, { ... "E contains syllable 'cat'", "E contains syllable 'catt'" ... }>[1]

while algorithm [2] would be something like this:

> <{ ... <cat, *kat*>, <catt, *kat*>, <kat, *kat*>, <katt, *kat*>, ... }, { ... "E contains syllable 'cat'", "E contains syllable 'catt'" ... "Syllable 'cat' is in [katt]", "Syllable 'catt' is in [katt]" ... }>

The extensional move basically resolves the level 1.5/level 1.6 ambiguity by ruling out the non-extensional level 1.5; the account automatically finds itself equivalent to what we have called level 1.6. Hence on this view Peacocke, if he means anything at all, can only mean level 1.6, and there is no ambiguity; the imprecise notion of a body of information which we were using earlier is simply

[1]Of course, in practice, the specification of one or both of these sets could be recursive.

ruled out of court. We shall see how tempting this move really is when we produce a more formal discussion in §5.1.

The extensional move could be made here; certainly, we will be concerned with level 1.6, and so it will not affect our point to rule out level 1.5 (as we have defined it here). However, the imprecision of level 1.5 might well be explanatorily useful, and I will not advocate the extensional move in this thesis. In the next subsection, we discuss a practical example of the explanatory usefulness of a level 1.6 model.

4.2.4 Models of Expertise

In KBS development, the intention is to build programs which simulate expert reasoning by formalizing the knowledge that an expert reasons with in the KB, and developing an inference engine to manipulate the contents of the KB. In the model-based methodologies upon which we are focussing, the aim is often to draw up a *library* of typical problem-solving structures, which a knowledge engineer could use to describe (perhaps at a coarse level of detail) the problem-solving (we saw a model from the KADS library in Chapter Two). One well-known, and often-used, problem-solving method is *heuristic classification* (Clancey 1985), which we shall use as an illustrative example in this subsection.

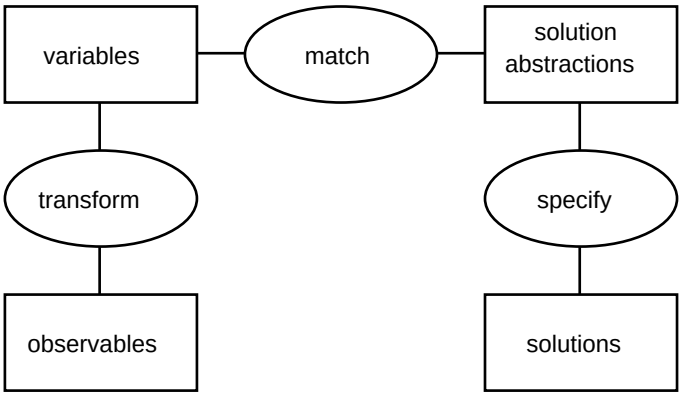


Figure 4.2: Heuristic Classification

The diagram has the same semantics as the KADS diagram of systematic diagnosis we saw in Chapter Two. The rectangles stand for metaclasses of domain information, characterized according to the *role* the members of that class play in the problem-solving process. Take, for instance, the information that 'x has pneumonia'. Depending on context, this piece of information can

function in a number of different ways: it can be a diagnosis (solution), or an hypothesis, or a datum, and so on. There could be a rectangle for each of these different ways. The ovals stand for ways of transforming or gaining information, specified at a relatively high level (for instance, they are not necessarily to be understood as algorithms). The connecting lines stand for data flow (i.e. how the information gets used in the problem-solving). Control (i.e. what gets done when) is not specified. So, heuristic classification involves three types of inference: firstly, observables are *transformed* into variables or findings — they are not used 'raw'; secondly, these findings are matched with abstract diagnoses on a heuristic basis (i.e. on an uncertain basis — heuristic associations are based on notions of typicality, and might only be poorly understood correlations); thirdly, the abstract diagnosis is refined into a more detailed diagnosis.

One can do these inferences in any order, because there is no specification of control at the global level. In other words, the inferences (transform, match, specify) are reversible: an abstract solution might be refined by specifying the non-abstract solution, while an abstract solution might be derived from the actual solution. Replacing the lines by arrows, we can see a number of possible ways of navigating through the diagram. Figure 4.3 shows three of these. Number 1 will solve a diagnosis task; given some observables, the diagnostician gets variables, abstract solutions and finally solutions in that order. Number 2 is a prognosis task; given a solution, the prognostician will compute what observables (properly, what classes of observable) the system would be expected to show. Number 3 is a hypothesis testing task; given the observables of the system, and the hypothetical solution, the expert works her way from observables to variables, and from solutions to variables, and looks for a match (i.e. are the observables of the system consistent with the proposed solution?).

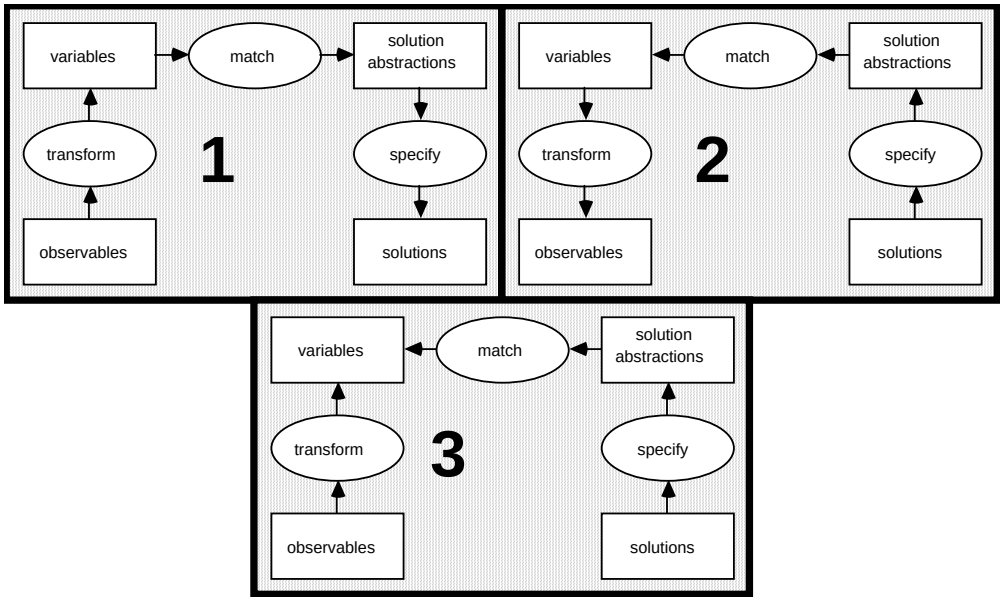


Figure 4.3: Three Ways of Navigating Through the Heuristic Classification Inference Structure

An example of an episode of heuristic classification is given in (Clancey 1985, p.295), taken from the inference structure of the KBS MYCIN. An *observable* might be that an adult patient's white blood cell count is under 2,500. This then undergoes a series of abstractions: the white blood cell count is low; this suggests leukopenia, which in turn suggests immunosuppression, leading to the *finding* that the host is compromised. This finding (with others) is matched heuristically onto disease *types*, for example, gram-negative infection. This general diagnosis is then refined to produce the subtype of the disease type which is the *diagnosis*, for instance E.coli infection. This episode is an instance of control structure 1 in Figure 4.3.

Clearly, this is a specification of the expert's problem-solving at a level more detailed than level 1.5 as we defined it in §4.2.2. The bodies of information drawn upon are the observables for the input, and the theory *within which* the solution refinement takes place — typically a hierarchical theory of solution classes — as the output. However, the model of heuristic classification also makes reference to some manipulative steps to be performed to those bodies of information. The observables are transformed into usable form; a general solution heuristically arrived at is refined into a more detailed solution. On the other hand, the specification is not detailed enough to be counted as a specification at level 2 — there is no *algorithmic* advice on how to perform the

transformation, or the matching, or the refinement. At best, the model specifies a *class* of algorithms.

What would happen in the event that the problem-solving *were* specified at level 1.5? We would have an input metaclass, the *observables*. We would have an output metaclass, the *solutions*. Note first that there could only be one step — because we are at level 1.5, we cannot specify that the input should be transformed into usable form, nor that the output should first be specified at a high degree of abstraction, and then refined. We make no assumptions about ontology, the forms that the bodies of information can take. So, what would the single inferential step in the middle be? We cannot assume that it would be a step of heuristic matching. If it were, that would in turn involve the assumption that the generally observed heuristic correlations between input and output were being used, of course, but those correlations may only be visible from the point of view of abstracted findings and solutions abstractions. From actual observables to solution subtypes there may be no obvious rough correlations to make. All one could specify about the inferential step in the middle, at level 1.5, is that a classification step is made, without any further information being available about the form that that step will take. We get the model shown in Figure 4.4. Note that this reflection on the distinction between levels 1.5 and 1.6 is not specific to this AI sort of problem, and can indeed crop up in Peacocke's own examples, as the discussion of the three syllable pronunciation algorithms [1], [2] and [3] above shows.[1]

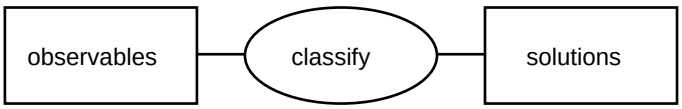


Figure 4.4: Simple Classification

The simple classification model may, of course, be of some use; however, it is clearly a model at a very low level of detail, and is unlikely to be of much help in KBS development (except in the case where it conveys the *simplicity* of the

[1]To give another example, consider Peacocke's example of kind perception by a 3-D model description (1986, pp.105-7). That model description 'uses various volumetric primitives and ... a coordinate system' (p.106). The level 1.5 account would not distinguish between different sets of volumetric primitives and coordinate systems, while the level 1.6 account would. Again it is not obvious on Peacocke's account whether he is giving an account at (our) level 1.5 or 1.6.

classification step, in contrast to other, more complex, varieties of classification).[1]

To give a small example of the difficulty of conveying the right information at level 1.5, consider Clancey's example from MYCIN of the diagnosis of E.coli infection from the white blood cell count (and other information). In order to get from wbc < 2500 to E.coli, a number of abstractions are performed, and we are forced at level 1.6 to give these abstractions. At level 1.5, these abstractions are suppressed. Yet the problem-solving process may not be understandable in the absence of the abstractions. Recall that KBSs need to give understandable explanations. To do that they must adequately represent not only actual problem-solving but also the vocabulary which the expert uses to justify her reasoning (otherwise the explanation will be unrecognizable as such). Further, the problem-solving itself may become intractable if not represented at the right level. For example, if the knowledge engineer reasoned directly from wbc to disease, the function could be too complex to compute in real time; on the other hand, if too much abstraction was performed on the wbc input, then the resulting classes of wbc could be too coarse to be of use in the classification (i.e. too inclusive to map onto any useful classes of disease). Similarly on the output side, both gram-negative infection, the disease *type* and E.coli infection, the disease itself, are equivalent from the level 1.5 perspective (both give information about infection). At level 1.6, we can distinguish between the two, and not only that, we can use the distinction; if a definitive diagnosis is too hard, we know that we can plump for gram-negative infection as a diagnosis, which, although it is not an exact diagnosis, will have some use (for instance, all diseases in that class may be treatable with antibiotics). Hence the program would be able to use this hierarchical disease structure to grade its diagnoses.

Having established that an important class of AI model is pitched at a level between 1.5 and 2, we need to establish now that a model of this class is of genuine utility. In other words, we need to show why we need a model at level 1.6 in KBS development, and not a model at level 1.5, or at level 2. This discussion will show that there is a constituency for level 1.6 models, and that there is a clear area for the application of such models. It will not show that such models are explanatory of expertise in the KBS development context — that is

[1]And, in fact, there is a further purpose that we can see for such a model, where a simple model such as that given in fig.4.4 is contrasted with other high-level models for the purposes of model *selection*; i.e. the model in fig.4.4 could be used to specify a *class* of classification models as distinct, say, from the class of *design* models. However, in such a case, the model would not be used actually to build the system. See (O'Hara 1993) for a discussion of these issues.

the business of Chapter Five. It will show that such models are useful. However, since to be explanatory is to be (among other things) useful (§§3.1.1, 3.3), it does at least show that such models meet a necessary condition for explanatoriness. Conversely, since the discussion will also show that models at levels 1.5 and 2 are not as useful (all things being equal), it will also entail that, if any models are explanatory in the KBS development context, level 1.6 models will be explanatory in the KBS development context.

There are (at least) two good *pragmatic* reasons why a level 1.6 model such as the one given in Figure 4.2 would be most likely to be useful. Firstly, recall that an important constraint on sensitively situated KBSs is that they provide understandable *explanations* of their output for the KBS user (§2.1.4). Imagine a system for medical diagnosis acting as a replacement for an expensive expert in some remote location, such as in an Antarctic polar base, or in a space shuttle. A patient's symptoms are fed into the computer, which then suggests that the patient's leg be amputated. A justification for the suggestion is then essential — no-one would sanction such drastic action on the basis of computer output only; suppose there were a bug in the code? But a justification in terms of the code may well be unsurveyable, especially if time were pressing. However, if the system's code and knowledge base were structured according to a level 1.6 model, then the explanations could also be given at that level of specificity. But this level of specificity also accords with the original expert's justifications; the reason for this is that the model is a model of the expertise at level 1.6, and hence will share the ontology of the expert. The expert will use the same terms as the model. Whereas a level 1.5 model would only specify the body of information on which the inference should draw, and this might be conceptualized very differently in the machine and the expert. Hence it will not necessarily be the case that an explanation derived from a model at level 1.5 will be sufficiently like the expert's own justifications (which, of course, will be based on a practice specifically designed to convince laymen of the expert's veracity). On the other hand, a level 2 explanation will probably be too specific in most circumstances. What one wants to know in the envisioned situation is the basis for the decision, not the detailed steps of the decision making itself (although one might well want to know this in some circumstances, the algorithms themselves should have been verified during the development of the KBS). Chandrasekaran et al (1989) discuss these issues in much more satisfying detail.

The second reason for wanting a level 1.6 model is that not all KBSs are there to *replace* experts. If the KBS is *simply* to perform the task that the original expert performs, then plausibly a performative equivalence (i.e. level 1) will suffice (qualms about explanations notwithstanding). But many KBSs are used for other purposes, e.g. training. If an expert's time is expensive, and in demand, then it may pay to employ a KBS in at least the early stages of training others in the expertise. Then a system that reasons in similar ways to experts in the field will have greater value for that purpose (such a system would also have to produce understandable explanations too, of course). Again, level 2 equivalence (algorithmic equivalence) is at too high a level of specificity; this would make distinctions between very minor variations in computation (e.g. between scanning a list from left to right and from right to left). Level 1.5 equivalence, on the other hand, would tell the students what information to look for, but would provide no guidance as to the most useful form that that information can take (i.e. how to conceptualize the information, whether to abstract the information, what basic ontology should be used). Level 1.6 equivalence, on the other hand, would not make the very fine-grained distinctions of level 2, while being less coarse-grained than level 1.5. Such a model would tell the students what ontology the experts use, and the most useful form for problem-solving that the information can be presented in.

So, to conclude, a level 3 model will be too detailed for the purposes of building KBSs to replicate expert performance. A level 2 model will also be too fine-grained — algorithmic equivalence will cut across systems that merely use different primitive steps for the same ends. On the other hand neither level 1 nor level 1.5 models are likely in the general case to specify enough to allow the system to be built. This means that level 1.6 looks like a likely resting place for the preferred models in KBS development. What this shows is that this particular type of account is going to be more useful in that context than other types. In Chapter Five, we shall attempt to draw all the threads concerning our discussion of explanation generally, explanation in psychology, and models for KBS development, together, and try to show that such level 1.6 models are explanatory in the context of KBS development.

Chapter Five: Conceptual Models, Competence Models and Psychological Explanation

So, our preliminary work has been done. To recap, we set out a philosophical project we called bottom up philosophy. In this project, the aim is to bring philosophical considerations to bear on the actual output of various areas of discourse, as opposed to theorising on the ideal output of those areas. In particular, in the philosophy of AI, the aim of bottom up philosophy would be to try to discern how the actual practice of AI is of philosophical interest — and try to discern this with an eye on the essential conceptual and practical limitations of AI — as opposed to developing and examining the consequences of various ideal thought experiments. In Chapter Two, we reviewed an actual area of AI discourse, the field of knowledge-based system development; our aim is ultimately to show that the models of experts developed in this field can be seen as explanatory of their expertise. Chapter Three looked at the philosophy of explanation, in which we brought our bottom up principles to bear. Finally, in Chapter Four, we examined a particular type of explanation, computational explanation in psychology, and concluded that, to be useful in KBS development, the computational models would ideally be explanatory at a level which we termed level 1.6, and which makes explicit certain distinctions which are rather glossed over in recent work by Christopher Peacocke.

Having established our basic position on these matters, we can now move on to our final task, the establishment of our claim that models of expertise in KBS development can be seen as explanatory of the expertise that they model. We will argue in three stages. Firstly, in §5.1, we will set out a type of psychological model, which we shall call a *psychological competence model*, and try to show that such a model is explanatory. Secondly, in §5.2, we will try to show that the conceptual models of expertise from KBS development are instances of such models; their explanatoriness will therefore follow. Then, in §5.3, we will go further, and try to show not only that such models are explanatory, but also what type of explanation they embody.

5.1 Competence Models and Psychological Explanation

Our aim in this section is to examine models of problem-solving. We shall look at ways of solving problems, and ways of representing the problem-solving steps. Our claim will be that models of such methods of problem-solving are interesting explanations of problem-solving in their own right. Of course, not all psychological phenomena or abilities are necessarily related to problem-solving — although that will depend on how wide is the interpretation of the term 'problem-solving'. Nevertheless, expertise, however that is to be defined, will surely be based upon the solutions of characteristic problems. Hence explanations of expertise will involve explanations of problem-solving behaviour, and therefore the concentration on models of problem-solving, which might be a handicap in other areas of psychology, is certainly to be welcomed here.

5.1.1 Models of Problem-Solving

How can we model problem-solving? We can begin by noticing the obvious point that what someone does when she solves a problem is to find a solution. Her problem-solving activity takes a problem and maps it onto a solution. Hence, we can express — at a very high level — what is going on by a two-place predicate which contains an expression of the problem in the first argument place and a solution in the second place. This assumes that we can express what the problem is, and what the solutions are, but this should not be too controversial an assumption. This leads us to define a *problem domain* Ω as follows:

Ω = <P,S,solution> where:

- P is the set of problems
- S is the set of solutions
- solution is a relation between P and S

A solution for a problem p is any s in S such that solution(p,s) is true. The problem p can be said to be *solvable* if there is an s in S such that solution(p,s) is true.

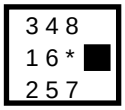
This, as we have said, is a very high level characterization of problem-solving competence. In the terms given in §4.2 above, this is a level 1 characterization, where only the input and output are mentioned. Although the full

characterization of the solution relation will enable any potential solution to be *recognized*, it does not allow a solution to be *generated*. An account of this form may be explanatory (at level 1) if the solution relation is trivial enough to be characterized easily in full. The result of such a characterization would be a look-up table. However, because of the size and complexity of most non-trivial problem domains, the solution relation would generally have to be expressed by some sort of formula, rather than as a list of acceptable pairs. It is the development of such formulae that will interest us, for they can be seen as expressive of the expertise. If such formulae are well-chosen, they may act as explanations at a other levels than level 1 (for instance, if the problem-solver's own algorithm is provided, then we would have a level 2 explanation).

So, to give an example, consider the so-called *eight puzzle*. This is a problem domain where eight tiles are initially arranged randomly on a 3 ∞ 3 matrix, and are moved round until the tiles form a pattern. Many children's toys have a similar form, with (mn-1) tiles on an m ∞ n matrix; an example of a start position of the Apple Macintosh 4 ∞ 4 puzzle (standard issue on Macintosh computers) is given in Figure 5.1. When the tiles are moved into the space, they should be manipulable into the shape of the Apple Macintosh logo, a rainbow-coloured apple with a bite out of it.

Figure 5.1: A Puzzle

The eight puzzle (which is the smallest non-trivial puzzle of this kind) has often been used as a sample problem in AI (Nilsson 1980; Van de Velde 1988). We can represent the state of the board at any time as a 9-tuple, with an asterisk representing the space, and the tiles numbered and represented in order. Figure 5.2 shows an example representation.



$\Rightarrow$ <3,4,8,1,6,*,2,5,7>

Figure 5.2: Representing States in the Eight Puzzle

Now we can represent the eight puzzle; the set of problems will consist of the set of pairs of initial states I and goal states G:

P = {<I,G> | I,G □ States} where States is the set of states of the eight puzzle

So, if we wanted to transform the state represented in Figure 5.2 into a state where all the tiles were in order with the space in the centre, we should state the problem as follows:

(<3,4,8,1,6,*,2,5,7>,<1,2,3,4,*,5,6,7,8>)

A solution consists of a series of moves. There are only four moves possible: a tile can be moved leftwards, rightwards, upwards or downwards into the space. If we represent these moves as L, R, U and D, then a potential solution would be an n-tuple of these moves (e.g. <L,D,D,R,U,R,D,L,U>). The solution predicate then would have the form:

solution(<I,G>,<M^n>), where M^n represents n moves in order, and I and G are initial and goal states

This would be true when and only when the solution in the second argument place transformed the stated initial state I into the goal state G.

This formalism will express problem-solving knowledge at a very high level for any problem which can be stated. For example, in diagnosis, problems would be tuples of symptoms, and the set of solutions would be the set of recognised malfunctions in the diagnostic domain (e.g. the set of diseases in a medical domain). The solution predicate would be true when the solutions in the second argument place were, or could be, the causes of the symptoms in the first argument place. As another example, in design, the problem would be a set of specifications, and the solution a configuration of components into an artefact.

Then the solution predicate would be true when the artefact in the second argument place satisfied the specifications in the first argument place.

However, whether or not such models could ever be explanatory — and in a number of circumstances I think they could be — it is clear that, except in very simple domains, they could not be explanatory for the purposes of AI. The context of AI demands a greater depth of explanation than the simple expression of the solution predicate. For example, whoever is building the system will require a method of getting from the problem to the solution. It is not enough to be able to recognise a solution when it comes along (except in very small domains where it is feasible to search through the solution space[1]). The programmer must have a way of generating the solution from the problem, and this, of course, is not provided by a simple specification of the predicate. Furthermore, the method of getting a solution from the problem must also be as efficient as possible, and on top of that it must run in real time — no point having the system if it takes days or weeks to produce the answer. Hence the AI programmer will require some constructive account of the solution predicate, and therefore our account here will need some more detailed way of modelling methods of constructing solutions.

Such a characterization has been suggested by Walter van de Velde (1988), and in this subsection, we will present a theory that has been adapted from his. Van de Velde's theory is intended to classify AI systems, but we will adapt and extend it. Note also that van de Velde has restricted his attention to problem-solving; since we are only concerned with problem-solving, this is also acceptable.

We begin with some definitions.

> If R is a binary relation, then R^{-1} is its inverse, and T(R) is its transitive closure. We say that (a,b) is in R if and only if R(a,b). R(a,b) if and only if R^{-1}(b,a). (a,b) is in T(R) if and only if *either* R(a,b) *or* there is a c such that (a,c) is in R and (c,b) is in T(R).

[1]And few domains actually come under this rubric. For example, the eight puzzle, which is very formal and relatively trivial, has 1,398,101 solutions of ten or fewer moves alone. It would not be feasible in terms of time to search even this attenuated space of solutions; neither would it be feasible in terms of memory to store all these solutions.

So, the inverse of a relation R is the relation S such that S(x,y) obtains if and only if R(y,x) obtains (so the relation *child-of* would be the inverse of the relation *parent-of*). T(R), the transitive closure of a relation, holds whenever a chain of instances of the relation can be set up (e.g. if we had R(1,2) and R(2,3) and R(3,4) and ... and R(n-1,n), then we would have T(R)(1,n)). So the relation *ancestor-of* would be the transitive closure of the relation *parent-of*. The relation *greater-than* over the natural numbers is the transitive closure of the relation *immediate-successor-of*.

We now need some intermediate definitions of relations which themselves depend on a series of relations.

> If R_j (j = 1,...,m) is a set of binary relations, for each j let R_j^+ be the union of R_j and R_j^{-1}. Hence (a,b) is in R_k^+ if and only if *either* $R_k(a,b)$ *or* $R_k^{-1}(a,b)$.
>
> If R_j (j = 1,...,m) is a set of binary relations, let R^+ be the union of all the R_j^+. Hence (a,b) is in R^+ if and only if there is a k ($1 \leq k \leq m$) such that (a,b) is in R_k^+.
>
> If R_j (j = 1,...,m) is a set of binary relations, let R' be $T(R^+)$. Hence (a,b) is in R' if and only if *either* $R^+(a,b)$ *or* there is a c such that $R^+(a,c)$ and R'(c,b).

We now have sufficient logical apparatus to define the key notion of a decomposition of a relation.

> If R is a binary relation on sets A and B, then a *decomposition* of R is a collection of sets E_i (i = 1,...,n), such that $E_0 = A$ and $E_n = B$, and binary relations R_j (j = 1,...,m), such that R is a subset of R'. The R_j are referred to as *inferential relations*. The E_i are referred to as *inferential types*. For each j ($1 \leq j \leq m$), there exist g ($1 \leq g \leq n$) and h ($1 \leq h \leq n$) such that R_j is a relation on $E_g \infty E_h$.

The idea of a decomposition of a relation is roughly that, if R(p,s), then it should set out a number of intermediate steps between p and s. So, for example, take the relation *cousin-of*: x is the cousin of y if one of x's parents is the sibling of one of y's parents. Then we can *decompose* the *cousin-of* relation into two *inferential*

relations, *parent-of* (we will need both it and its inverse), and *sibling-of*. If *cousin-of(p,s)*, then there must be q and r such that *parent-of*$^{-1}$*(p,q)*, *sibling-of(q,r)* and *parent-of(r,s)*. The inferential relations have decomposed the more complex relation *cousin-of*.

To encode this example in terms of the more formal definition, R is the relation *cousin-of*. There is only one inferential type, E (the set of people). R_1 is the relation *parent-of*; R_2 is the relation *sibling-of*. There are no other inferential relations. All the relations (i.e. the inferential relations and R) are relations on $E \infty E$. We have $R_1^{-1}(p,q)$, $R_2(q,r)$, and $R_1(r,s)$. Therefore, (p,q) and (r,s) are both in R_1^+ while (q,r) is in R_2^+. Therefore, by definition of R^+, (p,q), (q,r) and (r,s) are all in R^+. Since (r,s) is in R^+, it is also in R', by definition. This entails that (q,s) is in R', because (q,r) is in R^+ while (r,s) is in R'. This in turn entails that (p,s) is in R', because (p,q) is in R^+ while (q,s) is in R'. But p and s were completely general, so it follows that whenever R(p,s), R'(p,s). Hence we have a collection of inferential types, a collection of inferential relations, and R is a subset of R'. Hence we have a decomposition of the *cousin-of* relation.

We have already defined the notion of a problem domain, Ω, which is a triple of problems, solutions and the solution relation. The insight that we want now to exploit is that, since the solution relation is a binary relation, we can decompose it. We can call the decomposition of the solution relation a *pattern* or *structure* of inferences over Ω. In other words, the structure of inferences should mark off a number of intermediate steps between the problem p and the solution s.

The inferential types (the E_i) in the decomposition are sets of primitive objects; we can refer to a member of an inferential type as an *inferential primitive*. So, as another example, consider the relation between a cricketer and his batting average. The solution relation *average-of* is a subset of *cricketers* ∞ *rationals* (indeed, since it is functional, it maps a cricketer onto a rational number, although this account applies to all relations, not just functions). The inferential types referred to are therefore *cricketers* and *rational numbers*. Inferential primitives might include *Graham-Gooch*, *Curtley-Ambrose* and *Allan-Border* on the one hand, and any rational number on the other. We can decompose the *average-of* relation into a pair of relations designed to improve the comprehensibility of the top-level relation. Firstly, the cricketer is paired with his batting figures, which we can store as a lookup table. His figures are a triple of integers, containing the number of his innings, the number of times he was not

out, and the total number of runs he has scored (so, for Gooch, the respective numbers for the 1993 season are *35*, *3* and *2,023*).[1] This gives us a new inferential relation, which we can call *figures-of*, and a new inferential type, which is a triple of integers. Finally, we can go from a triple of integers to a rational, by noting that a cricketer's average is the number of runs he has scored divided by the number of times he is out. Hence if *(a,b,c)* is a suitable set of figures, this must be mapped to (c/b-a); in Gooch's case, *63.21875*. This is our final inferential relation, which we can call *(c/b-a)*. The *average-of* relation has been decomposed.

We should note here that the inferential types might be characterized either intensionally or extensionally. So we could insist that the inferential type which provides the domain of the *figures-of* and the *average-of* relations should be defined as either *the set of cricketers*, an intensional entity which could be used time and again, or as a particular extensional set of individuals which changes from year to year (in 1993 according to *Wisden {Yates, Boon, M. Waugh, Martyn, S. Waugh, ...}*). If the characterization was intensional, then it would be possible for two models to agree on their inferential types, but not on the inferential primitives. A triple containing Ω, the set of inferential types and the set of inferential relations we can call an *inference structure*. If the characterization of the inferential types was extensional, then the inference structure would also determine the inferential primitives.

Roughly speaking, then, what we interested in is a decomposition of the solution relation in terms of a set of inferential relations and a set of inferential types. The particular form of a competence model that we are seeking is one where all the inferential relations are more or less simple and tractable, and together they can be put together to construct a solution. The account of how the inferential relations relate to the solution relation is the *competence theory*.

The inferential relations are 'put together' as follows. We can define an *inference path* for an inference structure, which is a sequence of inferential primitives $(e_0,...,e_r)$, such that for each pair (e_k,e_{k+1}), there is an inferential relation R_j such that either $R_j(e_k,e_{k+1})$ or $R_j^{-1}(e_k,e_{k+1})$. We saw in the *cousin-of* example how the inverse could be employed — a lot of expert problem-solving practice does

[1]Of course, this is not a great example, since the average could just be added to the lookup table. However, the point is still made, because what we really want to do is to show how the decomposition can (help to) explain the solution relation. The mere pairing of Gooch with his average fails to explain anything. When this decomposition is displayed, the relation of Gooch with an apparently arbitrary rational is made clear.

involve working backwards through an inference. A *solution path* is an inference path such that e_0 is the initial state and e_r is the goal state — i.e. a path from a solvable problem to one of its solutions. Hence, the competence *theory* can be seen as decomposing the path from the problem to a solution into a number of steps, based on the inferential relations and inferential types of the inference structure. If the theory is any good, the steps will all be tractable.

We can finally define a competence model as *an inference structure plus a competence theory*. Of course, many different competence theories could be associated with the same inference structure.

To illustrate with our example from the eight puzzle, we can imagine one possible decomposition being as follows. Take our original starting point, <3,4,8,1,6,*,2,5,7>. Our first task is to get the space into the centre of the board, which will involve one move (an L), to give <3,4,8,1,*,6,2,5,7>. Having done this, one could imagine having a series of routines that would swap round pairs of tiles, only temporarily disturbing the others (Korf 1985). These routines could be stored in a lookup table, because there will not be that many of them — 56 as a maximum, and a large proportion of those would be simple transformations of some primitive routines. Hence, the puzzle could be solved simply by swapping round appropriate pairs of tiles, to give: <3,4,8,1,*,6,2,5,7>, <1,4,8,3,*,6,2,5,7>, and so on down to <1,2,3,4,*,5,8,6,7>, <1,2,3,4,*,5,6,8,7>, <1,2,3,4,*,5,6,7,8>. The solution would then consist of the stringing together of all the appropriate moves for these transformations, which would be read off the lookup table. This would not necessarily result in the most efficient solution, but would, by decomposition, render the solution tractable.

And if such a method described someone's problem-solving practice for the eight puzzle (i.e. if she followed the method, by virtue of going through just those steps), it could be regarded as a model of her problem-solving. We would need to specify which features of the model are taken to correspond to the practice and which not. Our next task will be to define the class of competence models which have this correspondence.

To summarize the results of this subsection: we have defined a particular notion of competence as a decomposition of the solution relation for the problem at hand. A competence *model* consists of an inference structure (problem domain, inferential relations and inferential types), plus a competence *theory* which

consists of an account of how to assemble the inferential relations into a solution path (i.e. an inference path which has a problem as its first member and a solution to that problem as its last). So a competence model specifies a way or ways of solving the problems in the problem domain whose solution relation which is decompose by the model.

5.1.2 Psychological Competence Models

In this subsection, we will look at the question of whether competence models as defined above could be psychologically interesting. What we need to know is what such competence models can tell us. We will try to provide a general account of the limitations and specialities of models of competence. This will prepare us for our examination of the question of whether such models can be explanatory in §5.1.3.

The notion of competence (as a general notion in the psychological literature) can be seen as related to the notion of performance. The performance of a system is its input/output behaviour, the way it behaves in concrete situations. It is often the case with a complex system that that behaviour can be rationalized, or given some short functional description. So, for example, we can see the performance of an electronic calculator when we press buttons; we see what output follows what input. And, given sufficient observations, we can formulate a theory to enable us to predict what output will accompany what input. This theory might be seen as a competence model of the calculator — very low level, of course, but the main point is that it gives us an account of the output that is not entirely dependent on observed performance. In such a model, a problem would be admissible sets of keys to be pressed before pressing the '=' button; the solutions would be the correct answers displayed in the calculator's visual display unit. One possible decomposition could be in terms of three inferential relations: one which related the buttons pressed to a particular numeric problem; a second which related numeric problems to their answers (real numbers); and a third which related real numbers to the visual display. Other possible decompositions might take into account the structure of the electronic circuits of the calculator. We would hope that our miniature competence model would work counterfactually, as well as predicting and retrodicting actual performance. The competence can be seen as in some sense *underlying* the performance. In the case of the calculator, the competence model describes what the calculator is *designed* to do.

But, because the calculator is a finite system — though too large to be adequately represented in a simple lookup model — the competence model is likely to go wrong, by idealising the calculator's output. The calculator would only be able to represent numbers of a certain size; hence if standard arithmetic were to be used as the model, in extreme situations, the calculator's output would diverge from the output of the model. The model, indeed, might be able to flag these divergences, and so the model could also act as a warning device, telling us when the calculator was reliable and when not.

In general, a competence model gives a way of inferring a solution to a problem, but does not necessarily tell you how any given system actually solves the problem. A piece of machinery need only conform to a competence model in one respect: the problem-solution relation is the same (modulo idealization) as the machinery's IO relation. Traditionally, in psychology, a competence model is generally seen as a model of an information processing structure underlying some actual regular performance. We can see that our notion of a competence model defined above could come under that rubric. The performance (at least some of the performance) is described by the solution relation in the problem description. In the competence theory, the decomposition of the solution relation will exactly preserve the IO relation, by definition of a decomposition and of a solution path; however, of course, the solution relation itself might be an idealization of the actual problem-solving that goes on.

We can make the relationship between given systems and their competence models closer by placing *further* requirements on the candidate decompositions for competence modelling. Many decompositions of problem-solving are possible, and each one of these is a competence model. For many purposes, there will be few restrictions on which competence models are interesting. However, not all such models will be of value in *psychological* investigation. Our tactic therefore is to consider those competence models as defined in §5.1.1 which are also psychological information processing models such as were discussed in §4.1. Therefore we need further constraints that can be placed on competence models to ensure that they will be interesting as psychological information processing models. We can define a *psychological competence model* as a competence model that meets these conditions.

Firstly, of course, there will be infinitely (indeed, uncountably) many decompositions that will do the trick. But the decomposition we choose ought

also to have some predictive value; we need to be able to form reasonable expectations of the performance of the modelled system. If there is a requirement for prediction, then this will rule out most other candidate decompositions.

Secondly, as we have already discussed, the competence as defined here need not measure up exactly against performance. Of course, the performance should be respected most of the time, but it is important for a psychological competence model to remain tractable. If the model is to explain the performance, the explanation itself should be tractable, by our discussion of explanation in Chapter Three. An analogy would be between performance and competence, and the scatter of points on a graph and the smooth curve drawn through them.

Thirdly, a psychological competence model will have to satisfy various further constraints on our making *sense* of the faculty or practice. These will be variable with context, but basically, as far as possible, the model should be some sort of *rationalization* of the performance. In some circumstances, there should, perhaps, be a sort of (possibly rough) mapping from concepts of which the performer is conscious of manipulating during any reasoning process preceding the performance, and concepts (i.e the inferential types, primitives and relations) in the model. Other circumstances might call for a mapping from interpretations of brain states to concepts of the model. Basically, the (deliberately loosely phrased) requirement is that the psychological competence model should bring some understanding to the outside observer. The components of the model, the types, primitives and relations, should carry some wider significance where possible.

Fourthly, the fact that a psychological competence model is an information processing model means that various constraints are applied to the competence model that apply to information processing models generally: its exercise must involve the use of a finite amount of memory; there must be a finite set of transformation or rewrite rules to manipulate the data; ambiguities or conflicts must be restricted. Further, each inferential relation in the model should be computationally tractable where possible; this last is not an absolute requirement, but there should be good reasons for neglecting it.

Fifthly, psychological competence models should describe the *production* of performance; *mere* prediction is not their game. Hence, there must be some sort of claim that "this is how it is actually done". As we saw in our discussion of

levels of explanation in §4.2, the phrase in quotation marks is very ambiguous. But the main point is that the manipulable entities postulated by the psychological competence model, and the inferences suggested by the psychological competence theory, must, where possible, have some sort of justification independent of their computational utility in the model itself. This justification will depend on the level at which the psychological competence model is intended to function: 1, 1.5, 1.6, 2 or 3 (or others).

Finally, we note that to describe competence, what needs to be given is an expression of admissible outputs, *not* a specification of actual output. At any point in the exercise of a more than moderately complex competence, it is likely that there will be a number of options available that still conform to the competence. For example, faced with a recalcitrant case, a doctor might prescribe any one of a number of possible tests/remedies. Some remedies might be 'worth trying', since they are unlikely to undermine the patient's health further, and may resolve the problem (e.g. a small dose of antibiotics, or a diet of fluids). Others might be riskier long shots, which might be attractive in serious cases (e.g. exploratory surgery). Given a patient with a particular set of symptoms and a particular history, it is implausible that in all cases there is only one possible thing to be done. In fact, this condition chimes in nicely with the definition of a competence model, since the solution relation and the inferential relations are relations, not (necessarily) functions. We can express what we require by saying that our psychological competence model should decompose the whole of the solution relation, and not just a functional subset of it.[1]

So, we have defined a psychological competence model as a competence model that also meets these six constraints. From now on, we shall only consider psychological competence models, not competence models in general.

5.1.3 The Explanatoriness of Psychological Competence Models

So we come to the real business of §5.1. How can a psychological competence model be explanatory when the subject of the model does not possess (at least in

[1]We can define a functional subset of a relation as follows: any subset of a relation, whose domain is the same as that of the relation, and which is a function. So {(a,1), (b,2), (c,4)} and {(a,2), (b,3), (c,4)} are functional subsets of {(a,1), (a,2), (b,2), (b,3), (c,4)}. The only functional subset of a function is the function itself; hence functional subsets are minimal. A decomposition of a functional subset of the solution relation will be guaranteed to find a solution for any problem for which a solution exists.

any overt, explicit way) the model? To take a simple example, we might wish to know why it is that a sheep runs away from a wolf. But it may well be no help to be told that

{Sheep(s), Wolf(w), (∀x,y)(Sheep(x) & Wolf(y) ∅ Eats(y,x)),
(∀x,y)(Eats(x,y) ∅ RunsAwayFrom(y,x))}
entails
'RunsAwayFrom(s,w)'.

The latter is a theory of sheep and wolves, that provides a derivation of the fact that s runs away from w from a couple of highly plausible universally quantified first principles, lawlike principles which are actually used in a deductive derivation of the desired conclusion; hence the argument is a deductive-nomological explanation according to Hempel's model, for example. But whether this is a satisfactory explanation really depends on what we want to know. There are many circumstances in which the theory is perfectly explanatory (e.g. in explaining what has happened to the sheep to someone who was not very conversant with sheep and their behaviour). But if we want to know why *this sheep* runs away from *this wolf* — i.e. if we want some sort of discussion of the *internal sheep processes* that lead it to run away from this specific object, the wolf, then the theory is next to useless. The Hempelian explanation certainly *would* act as an explanation in the desired sense were we able to credit the sheep with being a fellow possessor of that theory. The sheep would have to satisfy a number of conditions for this model to work as an explanation in this sense: 1) it would have to believe the four propositions (with suitable substitutions for the unquantified variables); 2) it would have to possess sufficient logical apparatus to derive the conclusion of the argument; 3) it would have to accept the theory as having impelling force (in other words, it should take the conclusion to mean that it *ought* to run away from the wolf); and 4) in the particular circumstances in question it should be disposed to consult the theory in order to determine its future actions, and not to consult any other theories. Clearly such support will not be available.

Psychological competence models operate in a similar sort of way. Possession of the model would be sufficient to explain someone's competence; however it is never (or rarely, at the very least) the case that someone possesses a model of his or her own competence. Hence in this section we will have to show how such a model really could be explanatory and in what circumstances.

We have discussed the notion of an explanation in Chapter Three above. In §3.3 of that chapter, we set out a number of conditions which an explanation should fulfill. We should make clear how psychological competence models as we have defined them meet these conditions.

The first condition was that the value of an explanation should be relativized to an audience. What the psychological competence model does is to decompose the solution relation into a number of inferential relations. In other words, what seems a complex inference or process is broken down in the competence model into a number of component parts. We used the example of the eight puzzle and the cricketers' averages in §5.1.1. In order for such a model to be explanatory in the sense outlined in Chapter Three, therefore, it is necessary that there is an audience for which such a decomposition is valuable; the audience should desire to perform some action, and the explanation should aid in that performance. The obvious audience here is an audience which wishes to build some sort of computational system to mimic the problem-solving performance. The decomposition of the model would have the effect of breaking down the competence into a number of tractable steps; this decompositional idea is precisely what underlies many influential programming methodologies. Hence there is at least one audience for this sort of model, and the condition is satisfied. In fact, there are many other audiences. As one example, someone interested in making a process more efficient might benefit from knowing which sub-processes were involved in the process. As a second, an ergonomist might be able to restructure a process in a way more suited to those who have to carry out the process (i.e. use a decomposition to isolate inappropriate subprocesses and replace them with other, more appropriate ones). As a third example, someone who wishes to learn the complex process might well benefit from having the process decomposed into a number of less complex processes which he is already able to carry out.

The second condition is that the model should be true or approximately true. The essence of the psychological competence model is the relationship borne by the inferential relations to the solution relation, conditioned by the six properties mentioned in §5.1.2 that

psychological competence models should have.[1] We need to check that a model which claims to be psychological explanatory satisfies these properties. That the model is predictive and idealizes the performance can be checked by simple tests. That it rationalizes the performance can also be checked, possibly by running through it with the subject. The computational context can easily be verified. That the model produces the admissible output will also have to be tested. The final point is that the model should describe the production of the performance at some level. This can also be checked, but we will leave the description of this verification for later in this subsection, where we discuss the level of explanation we are aiming for. Hence a decompositional model can easily be shown to be true (if it is true) in all these ways. All the aspects would have to be satisfied in order for the model to be true.

The third condition is that the relation of explanans to explanandum needs to be clear. Formally, this relation is clear, since we have defined what a decomposition is and how it relates to the solution relation describing the problem-solving. What may also be required to establish this condition in a particular case is that the inferential relations, inferential types and inferential primitives had some independent justification apart from their value in the decomposition. In other words, we would need to understand why the inferential relations, types and primitives were chosen independently from their role in the model. One way in which this might be done is to show that the competence model correctly described the production of the problem-solving performance. This option boils down to the requirement discussed above that the account of the production of the performance be true at some level, and is therefore postponed until we discuss levels of explanation below.

The fourth condition is that the contrastive class should be clear. This is straightforward enough; the decomposition proposed in the model can be seen in opposition to other decompositions that may be available. The import of the model is that the problem is solved *this* way rather than *that*.

[1]The competence model be predictive, idealize but not neglect the performance, rationalize the performance, take place within a computational context, describe the production of the performance, and specify the admissible output.

The fifth condition is that the explanation should be shown to be suitably contentful; i.e. it must be defeasible. How this is so will depend on the methodology underlying the model-building process, and will, of course, vary from context to context. Hence we cannot establish this point generally. When we discuss conceptual models as psychological competence models (§5.2), we will need to show that the methodology for conceptual model development *is* suitably restrictive.

Sixthly, we need to know what exactly is explained. Basically, psychological competence models explain problem-solving; given a solution relation, the model shows how the competent performer can go from the domain of the relation to the range. The two most obvious questions that a psychological competence model might answer, or be used to answer, are: *why did X do that?* and *how did X accomplish that?* The first question is a request for the reason for an action X took. This action should — assuming the model *is* explanatory — either establish that a suitable inferential relation holds, or contribute to such an establishment. Then the model explains the action by showing how it contributes to the establishment that the inferential relation holds between two objects, and further by showing how the inferential relation contributes to the decomposition of the solution relation. The second question is a request for an account of how the expert's actions took her from the domain of the solution relation to the range, and can be answered by showing that the expert's actions were the establishment that the various inferential relations held between the various inferential primitives used in the problem-solving. Note that if an action is incorrect (the expert makes a mistake) the model need not answer such questions (though it might). The model describes competence, not necessarily incompetence. If the expert's solution of the problem is correct, then it should be the case that her particular solution is modelled by a solution path in the model. Of course, by the sixth constraint on psychological competence models, there may be many admissible solution paths; if the expert's solution was correct, and if the model is a good model, the expert's solution path will appear in

> the model, and can therefore be shown to have decomposed the solution relation as required.

This is enough to suggest that in some circumstances at least psychological competence models as defined here are explanatory of the problem-solving processes they model.

So, to illustrate this with an example, we can take the eight puzzle discussed above. An action would be a move in the game, taken from the four possibilities L, R, U and D. The input would be the current situation, and the competence model would suggest goals for the move to achieve. This setting out of goals would be part of the model's description of the relationship between the inferential relations and the solution relation; recall that the competence *theory* is that part of the competence model which tells how the solution relation is decomposed into the inferential relations. The subpart of the competence model that suggests the goal would be the competence theory; the subpart that suggests the action would be one of the inferential relations. This would answer the question *why did X make that particular move?* The rationalization of the series of moves made by X in a solution to the problem would answer the question *how did X accomplish that?*

So there is no problem finding questions which are answered by (the manipulation of) a psychological competence model. Further, as we noted, there should be no shortage of audiences interested in such a decomposition. Note also that these answers will not necessarily correspond to verbal answers that a competent performer produces. If we ask X "Why did you do A?" he may well flounder, or suggest some alternative reason. As an example, a cognitive psychologist might well be interested in competence models that only used a severely restricted set of inferential relations and types: relations that are 'understood' as being primitive operations (over the inferential types) available to the brain. Such operations might include edge detection on a retinal image, or filtering a grey level array (Johnson-Laird 1988 sets out the basics of cognitive psychology). Clearly there is no implication that the competent performer need be *conscious* of carrying out such operations.

We can characterize interested audiences, partly by considering the form of the psychological competence model, and partly by considering the methodology of construction. In the first place, we note that a psychological competence model is

an information processing model, and may well be computable in real time. Then it is clear that people who are interested in such models are ones who are interested in information processing accounts of the competence. There are many such people, of course. What we have learned about the brain indicates that information processing accounts of human faculties may not be absurdly out of place — though such accounts may well need to be supplemented with extra stories to describe many interesting phenomena. An account of how some task may be performed computationally may well drive neuroscientific investigations by suggesting computational structures that may be isolated — for example, we might cite the discovery and investigation of cortical maps (Changeux 1985, pp.115-24).

But in particular we are interested in the context of AI, and it is clear how such accounts are of interest there. For the purpose of an AI investigation is the eventual production of software which will enable a machine to imitate some human performance; some examples of domains of KBS research are given in Chapter Two. Hence, to take the example of MYCIN, MYCIN was a system that assisted doctors in the selection of appropriate courses of treatment for patients with bacteremia, meningitis and cystitis. What would have been of use for the developers of MYCIN would have been a model of the competences of various specialists in the treatment of those diseases. Had such a model — i.e. an information processing model — been available, then the task of coding up a system that could perform the task would have been made that much easier. For any KBS developer, a model of the domain and the likely inferences over the domain that are able to suggest a class of information processing structures will be of great interest. Recall from Chapter Two that the knowledge engineer will require something more than a simple explanation which enables him to program the IO relation, because the KBS will need to have explanation facilities built into it.

We now need to discuss the level of explanation provided by a psychological competence model. This will tell us what the model claims — and does not claim — about the production of the output of the modelled system. Clearly, a psychological competence model is explanatory at least at level 1; the solution relation itself is a sufficient IO characterization of the problem-solving behaviour, and *a fortiori* the trivial decomposition consisting solely of the solution relation is a competence model, and will meet the constraints on psychological competence models given in §5.1.2, and is therefore explanatory.

Equally clearly, a competence model is not necessarily explanatory at level 2. Whether or not it is will depend very much on the nature of the inferential relations into which the solution relation is decomposed. If the inferential relations are sufficiently low level as to be considered as corresponding to primitives in a programming or algorithmic language, then it might be averred that the decomposition itself provided an algorithm, and that therefore the model was explanatory at level 2. But this cannot be assumed to be the case in general.

What we do know is that, as well as the inferential relations, the model includes inferential types and may include inferential primitives.[1] The types are the sets related by the inferential relations, and hence they (together with the relations used in the decomposition) express the information drawn upon by the problem-solver; since a condition on a psychological competence model is that the decomposition rationalizes or makes sense of the problem-solving behaviour, we can assume that the inferential types are not there solely for the convenience of the model (we discuss the exact meaning of this phrase below). Therefore, we can say that a psychological competence model, by virtue of the specification of inferential types and the inferential relations which govern them, is explanatory at level 1.5. Recall that level 1.5 is the level of explanation where what is given is the input/output relation (in competence-model-speak, the solution relation), and the bodies of knowledge drawn on by the algorithm which actually will compute that IO relation (which we can take to be analogous to the inferential types and relations).

Furthermore, the inferential primitives are the elements of the inferential types; therefore they constitute an ontology of the inferential types. Again because of the rationalizing purpose of psychological competence models, we can assume that if the model contains inferential primitives, then the inferential primitives are not there solely for the convenience of the model. Therefore, we can say that a psychological competence model, by virtue of the specification of inferential primitives, is explanatory at level 1.6. At this level, recall, a model gives the IO relation (the solution relation), the bodies of knowledge or information (the inferential types and relations) together with a specification of an ontology of these bodies of knowledge (the inferential primitives). Note that, if the inferential types are characterized extensionally, then level 1.5 collapses into

[1]We say "may include" because one could imagine competence models which left the exact primitives unspecified. For example, recall our example of the computation of cricketers' batting averages. One might give the model of that computation, but abstain from actually specifying which cricketers it applied to. The inferential type (*cricketers*) would, in contrast, have to be specified.

level 1.6. However, in the definition of inferential types above, we did allow that they could be defined intensionally, which would make a gap possible. If the reader is extensionally-minded, and is not happy with any intensional characterizations at all, then there is no essential flaw in the position. It is level 1.5 as we defined it in §4.2 that depends on intensional characterizations; if they are ruled out, level 1.5 disappears. However, what we are interested in is level 1.6, and, since that is extensionally characterized, the definition remains sound even if intensional characterizations are ruled out.

We made use in the previous paragraphs of the assumptions that inferential types and inferential primitives are not there solely for the convenience of the model. By 'solely for the convenience of the model', I mean that the types and primitives are there to enable the model to work, and for no other reason.[1] We need just to flesh out the implications, for which we can look at the modelling methodology. The main point to note here is that psychological investigation can throw up facts that are not accounted for in the competent problem-solver's self-justification. For instance, it might be the case that such a problem-solver consciously makes two inferences which he claims are simple in some way, but that the time he takes to make the inferences varies wildly. This might be a ground for assuming that one inference is more complex than another, unbeknownst to the subject of the model. Another example: the time in which a subject searches a space may rise linearly with the increase in size of the space, which would suggest that the subject performs a simple search, but if the subject's search time rises more slowly than the increase in the size of the space, this would suggest that beyond a certain threshold the subject employed strategies to prune the size of the search space, possibly by not investigating implausible areas (such a strategy would not result in success every time). If a psychological competence model takes such phenomena into account — as it should — then this implies that the questioner is interested in accounting for

[1]As an example of the sort of thing I mean, suppose a model had two (meaningful) variables a and b, and their values needed to be swapped around, so that a takes on the value of b and vice versa. One way of writing this would be as follows.

```
a := b;    (a takes the value of b)
b := a     (b takes the value of a)
```

However, this causes problems. If a has the value 1 and b 20, say, then this program does the following: a becomes 20, and then b takes on the value of a, which is 20. Hence a and b have the same value, and have not swapped values. This suggests the following program to do the trick.

```
c := a;
a := b;
b := c
```

A new variable c is declared which acts as a scratchpad for the value of a. But c is only there to allow the program to be written; it has no meaning outside the program, unlike a and b.

such phenomena, as well as the 'surface' phenomena that might be dealt with satisfactorily by the subject's self-justification.

In the world of KBS development, one such example is the ontology of the explanation. It is often the case that, on the ontology suggested by the expert, the reasoning is difficult to perform, whereas supplementing the ontology by the output of a repertory grid analysis of the expert makes things easier. The repertory grid technique (§2.2.4) is used to discover 'constructs' about the domain which the expert uses, possibly unconsciously. The technique is basically to ask the expert to distinguish between various elements of the domain, and then to question her about the constructs she used to make the distinction. So, the expert might be asked to distinguish between X on the one hand, and Y and Z on the other. She might say that X had property P, while Y and Z do not. Then the expert will be asked about property P: is it 'all or nothing' or does it come in degrees? does it take values? can the property be deduced from other properties? etc.. Then property P can be added to the model; although the expert may not have referred to it in her original account of problem-solving, the fact remains that she used it to distinguish between X, Y and Z, and in that respect if in no other it will feature in an account of her competence.

So the information processing methodology of psychological investigation suggests experiments to perform — for instance, analysing how an inference was performed, rather than taking it 'on trust' — and can account for unexpected results of such experiments. If the investigator wishes to take such results into account, then the restrictions associated with the production of a psychological competence model as an explanation should respect that. Now, a straightforwardly psychological investigation is pretty well duty bound to take such phenomena into account. If one inference takes five times as long as another supposedly similar one, then any psychologist worth her salt should take note of that. And it is likely that any KBS or AI related enterprise will want to know what 'deep' structures are around, for it will probably be difficult to understand how inferences are made if such underlying structures are ignored.

This completes our account of the level of explanation of psychological competence models. Our final task is to relate this account to the promises made above with respect to the truthfulness of such models; recall that we said that the psychological competence model should describe the production of problem-solving output. This, of course, fundamentally, is an empirical matter, and will

depend on the way that the model is to be understood. If a model functions at, say, level 1.6, then it will not answer questions at level 2. So if such a model includes an algorithm, then it cannot be claimed that that algorithm is used by the competent performer. On the other hand, it should be true to say that the problem-solver drew on the information specified by the model's inferential types and relations, and furthermore drew on the information conceptualized in the same way as the inferential primitives. If such a claim is disputed by the problem-solver, the dispute should be settled by an examination of the model development methodology. The position should be that the model is the *best* explanation of why the problem-solver gave the output she did when analysed by the model development techniques (or KA tools, or whatever), where the meaning of 'best' will vary from context to context, relative to the audience's requirements. On the other hand, computational 'fixes' will not aid the explanation at all — although see (Motta et al 1994) for a discussion on various issues associated with this point; it is not always the case that it is easy to tell which is a computational fix and which is a principled piece of computing.

So, now that we know what questions a psychological competence model can answer, and what audiences might be interested in such answers, and how far we can tell that such answers are correct, then we are entitled to conclude that we have given an account of their explanatoriness. Psychological competence models are explanatory at level 1.6 (if they specify inferential primitives, and at level 1.5 if not).

5.2 Conceptual Models and Competence Models

§5.1 established a lemma that we need for our conclusion. We want to show that conceptual models are explanatory of expertise. Our lemma showed that psychological competence models are explanatory of problem-solving. Given our plausible assumption that expertise is a type of problem-solving, this leaves one more result to be shown, that conceptual models are psychological competence models. Recall from §2.2.2 that a conceptual model is a knowledge-based model of the expertise required for problem-solving in the object domain. A general sort of model-based KBS development methodology will produce first of all a conceptual model of the expertise, transform that model into a design model of the target KBS, and then transform that design model into an implementation. Hence a conceptual model is the first major product of the KBS development process, and will form the basis of the final implementation. We

will remind ourselves in more detail of what conceptual models look like when we come to discuss them in relation to psychological competence models in §5.2.2.

We will begin by discussing alternatives to the model-based KBS development methodologies — our deliberations will not, of course, apply to KBSs developed in such ways, and it important that we are clear about which KBSs are the subjects of the argument. Secondly, we will measure conceptual models against psychological competence models. Thirdly, we will briefly discuss the special types of conceptual models which are library-based. Finally, we will complete the inference about the explanatoriness of conceptual models and KBSs generally.

5.2.1 Modelling Methodologies and the Alternatives

It is important to realise that we are discussing conceptual models in the context of one class only of KBS development methodologies. For example, many systems are developed by *rapid prototyping* (Waterman 1986). Here, a small running system is put together as soon as possible on the basis of the first few consultations with the expert. This system can then be shown to the expert, and the knowledge engineer can re-design the system during further iterations of the development process on the basis of the expert's feedback. There are a number of advantages of this methodology. Firstly, the expert can see the consequences of her input running on the early prototypes — thus avoiding a major problem with many KA tools, which is the presentation of the expert's input back to her for validation, and the consequent need for sympathetic interfaces between the human and the computer. It is not sufficient for the expert simply to provide input for the knowledge engineer; the input will typically be given in unfamiliar forms (i.e. the expert will usually give the input in a way unfamiliar to the knowledge engineer, and the knowledge engineer will then encode that input in a way unfamiliar to the expert), and should be validated as soon as possible. The obvious way is for the expert to see the results of her small KB in action, and the easiest way to manage this is to develop small-scale prototypes of the target system as soon as possible into the project. Then the expert can see what output follows from her input.

Secondly, the method can be useful when there is a situation, as often occurs, of a hierarchical organization of experts. It is usually the case that the time of the

expert whose input might be considered, all things being equal, to be the most valuable — the 'top man/woman' — is too expensive and precious for him or her to give full attention to a KBS development project. A rapid prototype allows a system to be built quickly with the minimum of fuss. After the main expert has given his or her input at the early stage, the lesser experts can then attend to the fine tuning of the system in later iterations of the methodology. Or, alternatively, those lower down the hierarchy can supply the basic information and structure, while the main expert adds his or her special contribution in fine detail. Further, the process of rapid prototyping is simple to execute and understand.

However, it is the *disadvantages* of the approach that will make a difference to this thesis.[1] These all revolve around the lack of models as products of the various stages of development.[2] Models of the expertise, or of the target system, are very useful as evidence during verification and validation. Bugs in a KBS which was developed using multiple models can be traced and localized relatively easily, since each implementation decision will be traceable to some modelling input. In that event, the error can be placed as being in the conceptual model (i.e. an error in the conceptualization of the knowledge required to perform the task), in the design model (i.e. the design of the system itself was flawed), or in the implementation (i.e. the design was correct but not adhered to in the code).

Further, there will be an interesting trade-off when considering the important issue of efficiency. A rapid prototyping methodology gets a system running, and, if that system is a good one, can be the quickest way to an efficient system. However, with a model-based development methodology, greater efficiency can

---

[1]In fact, model-based methodologies generally help themselves (at least partially) to the advantages of rapid prototyping by including prototyping stages in their life cycles. The development of prototypes is an important part of virtually all large-scale computing projects, although there is a school of thought that says that programmers are generally *too* reliant on an intrinsically flawed method of testing output (Gries 1981; Hoare 1984). The difference between rapid prototyping and model-based methodologies is that rapid prototyping sees the development of prototypes as the means of 'homing in' on the target system, while model-based methodologies develop prototypes to test the various models as they are being developed.

[2]Although some (e.g. Clancey 1992) would claim that the system itself is a model of *something*. However, this is only to say that the model is the product of the whole process. Following this logic, each prototype system developed along the way will also be a model of the operation of the system in the world. But these successive models will not be *structured* in any way — they will have no *function* independently of the process of development. Our point is that in model-based development (and not in rapid prototyping), the process of development itself can have by-products which are models, and whose content will be, in a sense, independent of the target KBS, and further, that these various models can each be seen as useful for the process of development and support. Not only that, of course, but these models can have significance outside the KBS development process.

result in the following two ways. Firstly, there simply is no easy way of discovering redundant code in a system developed via rapid prototyping, whereas with a model-based approach, each bit of code is aimed at various specific parts of the final model, which at least expresses the structure of the system, and can be used to detect redundancies. Code (i.e. lines of a program) is redundant when it can never be used, or when its function is duplicated by other code. When a program is hundreds of thousands of lines long, simply storing it is problematic enough, never mind accessing the right line of code at the right time. Hence it is essential for large-scale projects that redundant code is avoided. In rapid prototyping, code is written specifically to create certain effects; the expert wants some particular output, and a few lines are added which create that output. Once the program is more than a few tens of lines long, it becomes increasingly difficult to discover whether or not code is being duplicated (especially if the program is a software engineering project, i.e. it is being written by a number of programmers under a project manager). In a model-based methodology, the code only gets written at the end, when it is keyed to particular sections of the structured model. Since the model is structured, it is relatively easier to detect redundancies in the model, and if there are no redundancies in the model there should be no redundancies in the code. And the second way in which model-based methods can be more efficient follows from the fact that the method of model refinement will make the development of a system rather more clearly specifiable — corners can't be cut, but equally blind alleys should be avoided.

Hence, we make no claim that all KBSs will have the properties that we have mentioned. Indeed, many developers of model-based systems can be very willing to do great violence to their models in the name of greater efficiency for the final system. So we should be wary even about attaching the reflections following to *all* model-based systems. However, all along I will attempt to make it clear at least roughly how far a system developer can go before these reflections fail to apply. In fact, my belief is that a 'sliding scale' will be appropriate — all systems depart from their conceptual models to some degree, and to that extent they will fail to be fully-fledged psychological competence models. But that is not an all-or-nothing thing.

5.2.2 Conceptual Models and Competence Models

The main business of §5.2 will be conducted in this subsection: we need to show that conceptual models such as we discussed in §2.2.2 are psychological

competence models such as we discussed in §5.1.2. We begin with a reminder about conceptual models.

Conceptual models are intended to model the expertise required to do the job of the target KBS. In general, they will contain a number of elements which are essential for the building of the target KBS. There are a number of theories as to which elements are essential, but a rough consensus has formed, and a good exemplar is the KADS four-layer model discussed in §2.2.2. Recall that the KADS methodology suggested an epistemological division of knowledge into four *layers*, each of which represented a *type* of knowledge which could in principle be acquired separately. These layers pretty well cover the various possibilities, and most major modelling methodologies can be represented without too much violence in a KADS-like format (Karbach et al 1990). As we discussed earlier, we are therefore justified in using KADS models as a standard; if our argument applies to KADS models, it will also apply to models built using other methodologies (and also to hand-crafted models).

The four layers are as follows: the *domain layer* contains the knowledge about the static objects and concepts of the domain; the *inference layer* contains the knowledge required for primitive inferences to be made about the objects of the domain; the *task layer* contains knowledge about how to put together the primitive inferences in order to achieve goals; and the *strategic layer* contains knowledge about how to set up goals and schedule tasks. What we need to show here is that each of these types of knowledge is meaningful in the context of psychological competence models, and that each component of a competence model is meaningful in the context of conceptual models. In particular, if every item in a psychological competence model appears in some guise in a conceptual model, then we can say that all conceptual models are psychological competence models.

We begin by showing that conceptual models are competence models. The components of a competence model are as follows (recall §5.1.1): the inferential primitives, the inferential types, the inferential relations and the competence theory linking the inferential relations to the solution relation. The domain layer of a conceptual model basically describes the domain without reference to inference; hence it can plausibly be assimilated to the inferential primitives (the objects that end up in the solution path). There are, however, two technical points which we must consider before we can make the assimilation; firstly,

there is the problem of ontology, and secondly we must look at how independent of inference the domain representations are. Neither point turns out to be serious, but each deserves a brief discussion. However, the less AI-minded reader may wish to skip the five paragraphs which follow and take it on trust.

The problem of ontology is that there appears to be room for a mismatch, because in the definition of a competence model, all inference is done with first order objects. The inferential relations are binary, taking their input, which must be an object or tuple of objects, and output, which also must be an object or tuple, from named inferential types. Hence all inference in a competence model is performed over objects, the inferential primitives. Whereas in a conceptual model's domain theory, it is quite likely that reasoning will be performed about objects *and their attributes* (recall our discussion in §2.1.2, where objects such as `my-beetle` had attributes such as `no-of-wheels`). Is it the case that this possibility rules out the assimilation of the object-based inferential primitives with the domain layer of a conceptual model?

Fortunately, the answer is 'no'. Generally speaking, in philosophical ontology, a parsimonious ontology is good. However, in computational ontology, what really counts is the ability to get the inferences done efficiently; the ontology of a computer program simply consists of the things the computer reasons about, and that question is much less deep than its Quinean philosophical analogue, the question of what things there are. So what we want to know is whether it makes sense to treat the attributes and values of attributes of objects in a conceptual modelling language as objects such as are reasoned about in competence models. In modelling practice, it turns out that it does make sense. This is impossible to establish in the short space available here, without departing too much from our philosophical brief. But to show how it is done, we can take a quick look at the knowledge representation language (KRL) Ontolingua (Gruber 1992). Ontolingua is particularly suggestive, since its aim is to act as an all-purpose KRL to translate between KBs written in different KRLs (in other words, it may not be efficient, but it is highly expressive). The way it represents an attribute is as a two-place relation, which takes as its first argument a domain object (e.g. `my-beetle`), and as its second argument a value (e.g. `4`). This particular pairing of two objects would be appropriate for the relation `no-of-wheels`. Relations are extensionally characterized, so they are sets of pairs of objects, which, being sets, are objects in their own right. Hence everything that might be reasoned about with respect to the number of wheels of a car, including the

relation itself, can be seen as an object. Even the knowledge that the number of wheels of my beetle is four can be seen as an object, the triple containing `no-of-wheels`, `my-beetle` and `4`. So the fact that competence models' reasoning is represented as being about objects exclusively is not a problem.

The second technical question is the independence from inference. Obviously, which inferential primitives get chosen depends on what inferential relations the solution relation will be decomposed into. Yet in a conceptual model, the domain layer is supposedly independent of the inference layer. Can these points of view be reconciled? Of course, they could be trivially reconciled, since there is nothing in the definition of a competence model that insists that all inferential types or primitives need to be used; however, a good competence model should be such that each part of it is significant, and so the inclusion of redundancy would only answer the question in a most unsatisfactory way.

Fortunately, there is a non-trivial reconciliation. The independence of the domain and inference layers is only partial. The inference layer contains patterns of inference that may be shared across domains (for example, the inference structure of systematic diagnosis that was shown in Figure 2.4, repeated as Figure 5.3). The domain layer is then a 'filling-in' of this inference structure. True independence would be achieved if the domain layer could be kept constant across various *inference structures*. So if the domain was the respiratory disease domain, for example, the same domain layer could be used for a diagnostic KBS and a prognostic KBS, or a diagnostic KBS that used the diagnostic inference structure heuristic classification, and a diagnostic KBS that used the diagnostic inference structure cover-and-differentiate. Domain layers *can* be transferred, but only up to a point.

What actually happens is that when we switch from one inference structure to another, the domain layer needs to be reconceptualized and reorganized somewhat. Hence efficiency is lost, both in the running of the target KBS and in its implementation. The problem lies in the ambiguity of the notion of 'the knowledge required to do the job'. Suppose we had a problem P, and there were two structures of inference, or methods of solving the problem, M and N. Suppose M requires knowledge K to solve P, and N requires knowledge L, and that K and L were not identical (as is typically the case). Suppose also that there are good reasons why M and N might each be chosen to solve P. Now, what is the knowledge required to do the job? There is an argument for saying that the

union of K and L is required, since that is the minimal set of knowledge which allows a choice between the two methods. But then each method would be weighed down with redundant knowledge. On the other hand, if K (or L) were chosen (as a minimal set), then the domain would not have been represented independently of method, since the choice of K (L) to represent the domain would entail the choice of the particular method M (N) to solve P. Hence representing the domain does in practice involve some sort of commitment to particular inference structures. Indeed Bylander and Chandrasekaran (1988) insist that the way that the domain is represented depends on which inferences are going to be used; the development of the system itself will be imperilled if the domain knowledge representation does not take into account the inferences that will be made over it. Therefore, in Chandrasekaran's Generic Task methodology, each Generic Task (= method of solution) has associated with it its own knowledge representation language. Since it has been shown that different Generic Tasks can solve the same problem (O'Hara and Shadbolt 1993b), it follows that choosing a representational format for the domain amounts to the choice of method for problem-solving.

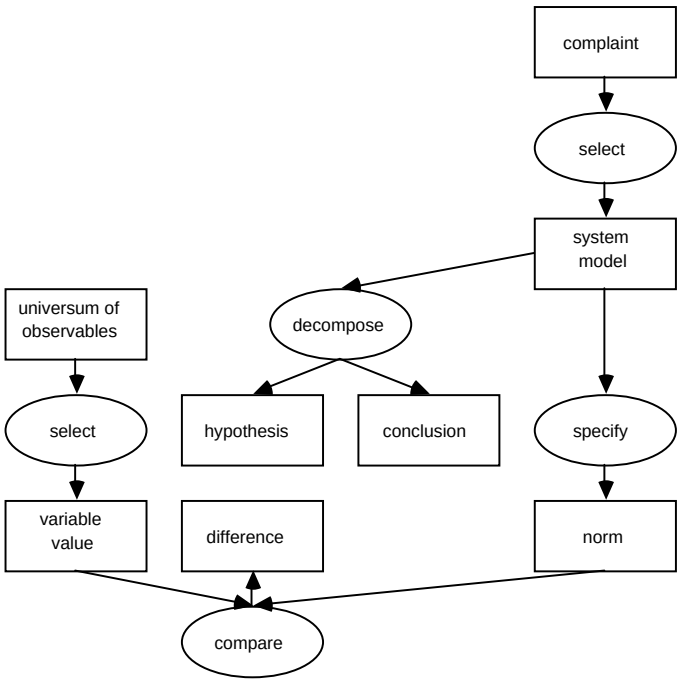


Figure 5.3: The KADS-I Model of Systematic Diagnosis

So, to summarize the outcome of this brief technical discussion, it is reasonable to assume that conceptual models, as well as competence models, reason solely about objects, and it is also reasonable to suggest that the particular objects used

in a conceptual model will depend on the inferences performed over them, just as the inferential primitives (and types, for that matter) of a competence model will depend on the inferential relations. There are philosophical analogues of the problems we have tried to meet here. For the first problem, an analogue might be theories of arithmetic that treat numbers as objects (e.g. they can be quantified over). One can go along with these theories 'as if' numbers were objects, without being committed to the Platonic view of arithmetic. The second problem is perhaps similar to the question of what the basic building blocks are of, say, the human body. The answer here is that it will vary from inferential context to inferential context. For example, for the physicist or chemist, the fundamental parts are atoms and molecules; for the biochemist they are cells; for the anatomist, they are such macro-systems as the lungs, liver, bones, etc.. The basic primitives vary as the inferential context varies. We find the same result with the inferential primitives of conceptual models. Hence there is no obstacle in the assimilation of the conceptual model domain layer with the inferential primitives of a psychological competence model.

Now we want to know what, if a conceptual model is a competence model, can be taken as the inferential types and inferential relations? Here, the answer can be found in the inference layer. Recall the model of systematic diagnosis in Figure 5.3. This is an inference layer structure, and connects classes of domain knowledge indexed by role, with primitive inferences. The whole represents a complex inference from a complaint to a state or process which is responsible for that complaint. Hence the inference structure can be seen as a decomposition of the solution relation which relates complaints with solutions. The metaclasses are the inferential types (so in Figure 5.3, the inferential types are represented by *complaint*, *system model*, *universum of observables*, *hypothesis*, *conclusion*, *variable value*, *norm* and *difference*), while the inference steps are inferential relations (so in Figure 5.3, the inferential relations are *select*, *decompose*, *specify* and *compare*). In KADS theory, these inferential relations are sufficiently primitive to be regarded as understood.

Finally, there is the competence theory, which relates the inferential relations with the solution relation. This can be assimilated to the knowledge in the task and strategic layers, in the following way. What goes on in the task and strategic layers is that decisions are made as to how to navigate through inference structures such as the one seen in Figure 5.3 so as to solve the required problems. One natural way of reading Figure 5.3 is that, given a complaint, you can find

out what state (disease, malfunction) is causing it — this is what we meant when we said in the previous paragraph that the inference structure can be seen as a decomposition of the solution relation which relates complaints with solutions. However, the solution relation could be differently characterized. For example, one may, given a diagnosis (disease, malfunction), wish to work backwards to the complaint, so that one knows what symptoms to expect — after all, the cure may be worse than the disease. This is a different solution relation, and the same inferential relations (and their inverses, which of course can be used in decompositions) would have to be combined in different ways. This is exactly the sort of issue that the competence theory is meant to tackle. Hence we can assimilate the competence theory of the competence model with the task and strategic layers of the conceptual model.

We have now effectively shown that conceptual models are competence models, as defined in §5.1.1. However, we are particularly interested in psychological competence models, a subclass of competence models which we defined in §5.1.2, because it is only psychological competence models which we have shown to be explanatory (§5.1.3). Hence we need to show that the constraints on psychological competence models apply to conceptual models, if we are to show that conceptual models are explanatory.

The first constraint was that the model should be predictive. This is easily met; a good conceptual model will be used in novel situations, and therefore cannot simply be a *post hoc* rationalization of the expertise.

The second constraint was that competence should be a 'smooth curve' through the 'cluster' of performances. Again, conceptual models bear a similar relation to performance of experts; it is not the case that *each* judgment needs to be preserved.

The third constraint was that psychological competence models should rationalize the competence. Conceptual models are developed using KA techniques. These are used with the expert's co-operation, and help to map the structures in the conceptual model onto the expert's ontology. Indeed, if there were not such a mapping, the explanations of the KBS's output provided by the KBS would not be understandable. Hence the conceptual model does help us make sense of the expertise. Of course, some KA techniques are designed to uncover ontological characterizations that the expert is unaware of in her own

reasoning. As we noted above (§5.1.3), the 'constructs' uncovered by techniques such as repertory grid analysis would count under this heading. Furthermore, the terms used in a conceptual model are relatively primitive inferences; hence the complex problem-solving is decomposed into more or less well-understood steps.

The fourth constraint is that various information processing constraints should be respected. This is usual for a conceptual model.

The fifth constraint is that psychological competence models should describe the production of performance, rather than *simply* predicting the performance. The terms of the models themselves need to have psychological significance. It is not enough for the model to be *a* decomposition of the solution relation (always assuming that the model is not intended to be explanatory at level 1); it has to give the inferential relations corresponding to the inferences that the expert actually makes. The level of correspondence then depends on the level of explanation that the model is intended to provide — typically level 1.6. We saw in §4.2.4 how, all things being equal, KBS development requires a model of expertise at level 1.6 — in other words, a conceptual model usually describes the production of performance at level 1.6. We have seen that psychological competence models can also do the same, in §5.1.3.

Finally, the psychological competence model need not be a function, and neither need the conceptual model. Admissible output only need be specified. What needs to be shown is that the expert's problem-solving behaviour in a specific instance will be *a* solution path of the model; it does not need to be shown further that this solution path is unique, or privileged in the model.

There are just a couple more points that we need to make. Firstly, we have to look at the audience for whom a conceptual model would be explanatory. We have seen that an explanation needs an audience, and also that psychological competence models are explanatory. Hence it follows pretty quickly that the sort of people who would find a psychological competence model explanatory would also be interested in a conceptual model — we mentioned programmers, ergonomists and cognitive psychologists. Clearly, as well as these there is a ready-made audience for conceptual models, and that is the class of KBS developers. What is required for KBS development is an explanation of the

operation of the expertise at level 1.6, and that, it turns out, is precisely what conceptual models provide.

And secondly, we noted in §5.1.3 (in the indented part) that we needed to show how conceptual models are contentful. If something is to be explanatory, then it needs to be shown that it is not the case that just any model will do. Of course, a conceptual model is constrained in a number of important ways. Firstly, it must provide an accurate specification of the solution relation. And secondly, associated with all the KA tools used in the KA process will be methodologies for their use. These will act as constraints; if a KA tool uncovers a certain inference pattern or ontological structure, then it should end up in the model; the knowledge engineer cannot just ignore selectively knowledge that he has elicited from the expert.

5.2.3 Off-the-Peg Models

There is one final cavil which we must address, before we can complete our argument, a cavil we first hinted at in §2.2.3. We have shown that conceptual models are psychological competence models, which is the result we wanted. However, we do want our conclusions to apply to more than a vanishingly small set of KBSs; what we want to show is that the class of conceptual models which are also psychological competence models is a significant class.

This is where a potential problem emerges. I think there is no doubt that we can be happy with 'hand-crafted' conceptual models developed for particular systems on particular domains, models that are 'made to measure' for the application. If a knowledge engineer sits down with the expert and puts together a special purpose model, then, assuming that the knowledge engineer is competent, the result will be a model that is explanatory. However, such systems are expensive to build; it is cheaper to develop a model using a skeletal model selected from a model library, as discussed in §2.2.2 (cf also O'Hara and Shadbolt forthcoming).

There are a number of modelling methodologies that include libraries of well-known or frequently-occurring inference structures. The model of systematic diagnosis featured in Figure 5.3 is a model from the KADS library of interpretation models (Breuker et al 1987). This is an inference structure, which comes with a default control structure. Chandrasekaran's Generic Task methodology (Chandrasekaran and Johnson 1993) provides inference *and*

control structures. The Generalised Directive Model (GDM) methodology (Van Heijst et al 1992; O'Hara 1993) provides inference layer components which can be assembled using a GDM grammar. In conceptual model terms, what these off-the-peg skeletal models provide is a specification of the inference layer and a partial (or default) specification of the task layer (in the case of GDMs there is no specification of the task layer). In competence model terms, they provide a specification of the inferential relations and types, and a partial (or default) specification of the competence theory (except in the case of GDMs).

The complaint that we might anticipate is that it might be the case that library-based conceptual models built out of the skeletal models in these various libraries do not fit the problem-solving sufficiently well to be explanatory of it. Even though the resulting models are good enough to build working systems out of, they might not count as actually explanatory. They will slip up either by not properly rationalizing the problem-solving, or by failing to describe the production of the performance at level 1.6. We might say that the expertise is 'shoe-horned' into the shape of the model for the convenience of the system-builders, not explained or even properly described.

It is unknown how many model-based KBSs are built using model libraries, and it is also unknown how many KBSs are built using the particular libraries mentioned here. Nevertheless, I will focus on them as examples, to show how there are two answers to the criticism. I hope that these three libraries can be taken as representative. The KADS library of interpretation models has been very successfully used in many European applications (Wielinga et al 1992, p.46). The Generic Task library may be the oldest; it has certainly been used in a number of American applications. GDMs are a relatively recent invention, and are still at the research phase. However, they have been used to build small systems (e.g. Motta et al 1994).

The early KADS interpretation model library was influential, but in fact was rarely used unalloyed in actual applications. The library contains a couple of dozen models (one of which is systematic diagnosis). What tended to happen was that the models did not quite fit the problem-solving (e.g. it might be that the problem-solving was systematic diagnosis without a system model — the list of hypotheses was known and the diagnostician simply ran through the list). Hence the library model would distort the problem-solving. But what could happen is that the knowledge engineer takes the model from the library as a template, and

simply constructs one that actually fits the problem-solving by adjusting, adding to, or removing, the offending parts. The KADS model is merely taken as a starting point, and is built on rather than being used wholesale. The individual parts of the inference structure — the boxes and ovals — lend themselves to being removed and replaced, so this was not too onerous. The result is a model which *does* fit the problem-solving after all.

Generic Tasks were, to begin with, even more rigid than KADS models (Chandrasekaran 1983). They contained, not only rigid inference structures, but rigid control structures and special purpose KRLs. In competence model terms, only the most general parts of the competence theory were left to the knowledge engineer's discretion. Again, this resulted in a change of tack for the methodology, for, although the six tasks in the library were very useful and ubiquitous, they were not quite flexible enough, but unlike KADS models, were not easy to customize. The result was a change in the methodology (and consequent change in the structure of the library), to a less monolithic approach (Chandrasekaran 1990). The Generic Tasks were replaced by generic Task Structures, loose families of control structures and ways of representing knowledge. Much more leeway was given to the knowledge engineer, and the result was customizable models as in KADS.

However, the customizing of Generic Tasks could have been controlled more strongly (O'Hara and Shadbolt 1993b). In other words, the changes that could be made to large scale abstract models could *also* be encoded in the library; if those changes were sufficiently small scale, it is plausible that even a library model would be fine grained enough to represent the problem-solving. This is the insight behind GDMs, which come with a decompositional grammar to allow any model to be rewritten in a number of minor ways. Space dictates that this cannot be gone into in detail; but an analogy would be with sentences and words. Complete models in a library are analogous to sentences describing the problem-solving. The GDM grammar is analogous to a formal language out of which many thousands of sentences can be constructed; the result is a recursive definition of models which means that many orders of magnitude more models can be expressed in the GDM library. For example, in a recent application, a very small library indeed (three rewrite rules) drawn up to describe teaching strategies was able to represent nearly a million models (Major and O'Hara 1994). The full GDM grammar will have tens of rewrite rules. Hence only *very* idiosyncratic problem-solving will not properly represented by such

decompositional techniques. O'Hara and Shadbolt (forthcoming), discuss the general movement of KBS development methodologies toward this more decompositional position.

Our conclusion, then, is that even off-the-peg models can be expected to describe problem-solving at a sufficient level of detail in a large number of cases. The first reason is that they are often customized by knowledge engineers. The second reason is that methods of representing models are getting more and more fine grained, and consequently more expressive anyway. To repeat, the purpose of this digression was to ensure that our results apply to as wide a class of models as possible.

5.2.4 The Argument Completed: Conceptual Models, Knowledge-Based Systems and Psychological Explanation

We have two pieces of business to transact in this subsection. We need to complete our arguments about the explanatoriness of the AI systems we have been discussing. The first argument is an argument to the effect that conceptual models are explanatory. This argument is very simply made. In §5.1.1 we defined a type of model which we termed a competence model; in §5.1.2 we defined a particular class of competence model which we called psychological competence models. In §5.1.3 we showed that psychological competence models are explanatory of the problem-solving behaviour that they model for a number of different audiences, as long as certain conditions held. In §5.2.2 we showed that conceptual models were psychological competence models and that the conditions did hold, while in §5.2.3 we showed that there need be no serious worry about the use of ready-made skeletal templates for the development of conceptual models. Since conceptual models are psychological competence models, then, it follows that they are explanatory of the problem-solving behaviour they model, for the various audiences we have mentioned, and no doubt others. Hence, on the assumption that expertise is problem-solving ability, KBS development can produce psychological explanations of expertise as by-products.

This shows that conceptual models are explanatory. However, our main interest in this thesis is in the AI systems themselves: can they be seen as explanatory? An examination of KBS development methodology suggests an argument to the effect that they can. The argument runs as follows.

1) Model-based KBS development involves refining a series of models, from the user requirements specification (which may involve models of the users, the used-upon, the computers to be used, etc.) and the conceptual model of the expert, through a more implementation-based design phase, to the implementation itself.
2) Since the development process is one of model refinement, the final system will share a number of characteristics of the conceptual model.
3) Anything that is sufficiently like a conceptual model in structure will be a psychological competence model of the expertise in question.
4) To the extent that the final system respects the conceptual model of the expertise, the final system is explanatory of the expert, by virtue of its being a psychological competence model, which itself is true by virtue of its being similar enough to the conceptual model.

Obviously, this argument can only refer to a subset of the set of KBSs. And the parameters can be argued about — I don't want to specify exactly how much a KBS can deviate from its conceptual model before it ceases to be viewed as a psychological competence model. It is certainly the case that this subset of KBSs will be significantly populated. Often, even if the detailed conceptual model is more fine grained than it need be for the system merely to run, it can happen that the detail of the conceptual model is welcome in other respects. For example, the business of providing decent explanations for the output of KBSs is an important one, and the finer-grained model is of use there for the final system, even if not for the main activity of deriving output from input.

Obviously, similar thoughts apply to design models. They have a lot in common with conceptual models, often being written in the same formal language. They are intended to take computational limitations into account, and so therefore may depart from the expertise. To that extent they fail to model it. But, like KBSs, if they retain sufficiently many aspects of the conceptual model, they can be explanatory too.

5.3 Knowledge-Based Systems, Normative and Causal Explanation

§5.2 has proved our main result of the thesis. Conceptual models are explanatory of the expertise they model, by virtue of being psychological competence

models. One final subsidiary question that we might wish to answer is: what *type* of explanation do they provide? Recall from §3.4 that we discussed two types of explanations, causal explanation and normative explanation. Each of these is a decent candidate for the type of explanation provided by a KBS or a conceptual model. In this final section of the chapter, we will examine the credentials of conceptual models with respect to these types of explanation. Our discussion will be brief and in no way conclusive; however, the conclusions we draw will allow us to make a tentative suggestion about the nature of expertise in Chapter Six.

5.3.1 Normative Explanation?

Recall from §3.4.2 that a normative explanation is a type of non-causal explanation that basically sets out the norms for a particular type of behaviour. One would expect from a normative explanation to be told how to do a certain task; because the basic items in the explanation are norms, it is up to me whether or not I follow them. The main point is that I have an account of the task which tells me that if I want to perform it, this is what I do.

But this is just what a conceptual model provides. A conceptual model is a psychological competence model. Hence it has the formal structure that we defined in §5.1.1, and meets the constraints given in §5.1.2. The solution relation characterizes the task, and the model decomposes the relation into a series of inferential relations, all of which are simpler or more tractable than the solution relation itself. Therefore, if each inferential relation corresponds to a relatively simple inference that I can make, then it follows that if I learn the conceptual model, I will be able to perform the task too (possibly not with the facility of the expert, of course). I will even be able to provide explanations and justifications of my output, exactly as the expert would be able to with respect to her output.

So the conceptual model can be seen as providing a set of norms governing some expert conduct; it tells you that *this is how it is done*. Further, because we are dealing with experts, it is reasonable to suppose that the method outlined in the conceptual model is actually the *best* way of performing the task. 'Best' might have one of two senses here. It could mean that the experts know the quickest/easiest/most efficient/most reliable method, which is enshrined in the model. Or it could mean that the real life task is constrained in a number of ways — so that the quickest route from problem to solution cannot always be used — and the expertise is maximally adjusted to take account of such constraints. An

example of such a constraint would be the necessity for an expert to be able to justify her output in courts of law; such a constraint might have led to a tailoring of the expertise to meet certain evidential conditions.

So we can say that a conceptual model provides a normative explanation of a practice, in the sense that it provides a manual for how to perform the task in some optimal way, and plays a justificatory role too, in that it sets out a series of inferences which show that the output is the right output for the task (even if those steps were not necessarily followed).

5.3.2 Causal Explanation?

Can a conceptual model count as a causal explanation? Recall from §3.4.3 that a causal explanation specifies a cause of the explanandum — the explanandum in this case being expertise, or expert performance — and specifies also the causally relevant property of the cause. We did not define causal relevance in full. One possible way in which a property can be causally relevant is for it to be causally efficacious. But it is difficult to see that a conceptual model outlines causally efficacious properties of an expert.

The components of conceptual models are designed specifically for the development of KBSs, as we have seen. The domain level entities are basically the sorts of things that can be used to model data in a computer — classes, elements, attribute slots, frames, etc. — which we saw in the section on knowledge representation, §2.1.2. At the inference level, what we find are typical domain-independent inferences, and types of data, indexed as to the role they play in problem-solving. Examples of the former are inferences such as *abstract*, or *match*, and of the latter are *observables*, or *hypotheses*. At the task level, we have problem-solving methods such as *heuristic classification* or *design by propose-and-revise*.

Now if we look at what would be the causally efficacious properties typically cited in psychology, we do not find that sort of entity. In this thesis, I would like to remain neutral between various positions on the causal story psychology tells, and neutral between realist and anti-realist notions of causation. But we can do a small survey of the field in the philosophy of psychology, to see if there is a glimmer of hope for the claim that conceptual models are about causally efficacious properties.

We can make a rough sort of distinction between connectionists and anti-connectionists, and consider briefly each case in turn. Anti-connectionists, such as Fodor or Pylyshyn, tend to see the causal entities as being at least something like programming language constructs, whose syntactic form is efficacious both semantically and causally, thereby making the link between reasons and causes. If this view held, then it would possibly be arguable that the domain level items in a conceptual model mapped onto causes relatively directly. The domain level items are pretty low level. They don't tend to be primitive in standard programming languages, but do appear in KRLs, which are after all special purpose programming languages. Moreover they can often appear in basic languages in mathematical logic. So it wouldn't be absurd to argue that they could be related to genuinely causal entities — neither is there any positive evidence that they are so related, of course. So one might reasonably hold the line that some structure in the brain represented, say, knowledge about the number of wheels a car has got, or knowledge about what temperatures can be taken as feverish. The classic example of such a view would be a language of thought account, which posits symbol types corresponding to the notions like 'number-of-wheels', such that these symbols have causally efficacious properties which information processing mechanisms can make use of. Since these would be actually identifiable structures, the claim could be made that they are causally efficacious. So the anti-connectionist view at least can make a relatively strong claim about the causal efficacy of the domain layer of a conceptual model (this would be an *a priori* claim, since there is no empirical evidence).

But if the jury is out for the domain level, it seems pretty unlikely that the inference or task layer entities could have any causal status (O'Hara and Shadbolt 1993a). No-one seems to have suggested that high level specifications of inferences or domain knowledge are somehow causally efficacious. It seems most unlikely that the methods for problem-solving outlined by the various KBS development methodologies feature as brainy structures. Conceptually, the various domain level entities have clear antecedents, but such large scale inference structures such as *heuristic classification* or *propose-and-revise* are recent inventions/discoveries, and it is surely improbable that they represent causally efficacious structures in the brain. The same goes for the *components* of such methods (such as *abstract*, *refine* or *match*). When, say, an observable is abstracted into a finding, it is highly unlikely that this describes any causal process taking the structure corresponding to the knowledge about the observable and producing a structure corresponding to a finding. It is very

unlikely that the causal processes in the brain magically correspond to the inference layer terms of KBS development models.

The position is even worse if one takes the connectionist or eliminativist line, which locates the effective causal entities in the brain as described neuroscientifically. For it is as plain as a pikestaff that there is no heuristic classification neuron, or group of neurons, that can be causally efficacious in ways described by conceptual models. KBS development, as a discipline, is firmly in the standard, logicist camp of AI.

So, whichever philosophy of psychology is taken (at least of the main candidates), it is difficult to see how conceptual models can give causally efficacious properties. So if the only way to be causally relevant (as required for causal explanation) is to be causally efficacious, then conceptual models cannot be seen as causally explanatory of expertise.

However, as we noted in §3.4.4, there is at least one other way to be causally relevant. A property can be causally relevant if its realization ensures that there will be a causally efficacious property in the offing (Jackson and Pettit 1990). Such an abstract property *programs* for the lower order property. Many properties can be efficacious in causing a particular effect; the program property ensures that at least one of them will be realized. In this sense we can suggest that a conceptual model can after all be a causal explanation of the expertise.

Suppose a conceptual model attributes a state to an expert — for example, suppose that the conceptual model contains a rule

IF temperature(patient) > 39 THEN fever(patient).

This rule describes the expert's reasoning about fevers and temperatures . The expert behaves exactly as if she believes a universally quantified version of that statement. Of course, because our example is so simple, it is quite likely a medical expert *would* profess a belief in it. This example is at the domain layer, but equally inference and task knowledge (more usually knowledge how rather than knowledge that) can correspond to beliefs or knowledge that the expert has.

The result is that we have a sort of two-level program property. At the first level, we have in the conceptual model a more-or-less precise technical representation

of knowledge that the expert has. This programs for various beliefs that an expert has. For example, the expert might believe exactly that anyone with a temperature over 39° has a fever. Or she may have a belief which does not have precisely that content (she may believe that anyone who feels very hot to the touch has a fever, or that anyone who displays abnormal discomfiture as a result of high body temperature has a fever), but which is programmed for by the conceptual model. In other words, because she has the belief that she has, any model of her can contain the information that anyone with a temperature over 39° has a fever. At the second level, the belief programs for some causally efficacious property (at least, this is a standard philosophical story). (Davidson 1963) is the classic argument that beliefs can be causes; (Jackson and Pettit 1988) argue that it is natural to see beliefs as causally programming some property that is actually causally efficacious. Hence — since programming in this sense should be transitive — the conceptual model programs for some causally efficacious property in the expert. Hence the conceptual model is causally relevant, and hence the conceptual model provides a causal explanation of the expertise.

We should spend a little time considering the work performed on the psychology of KA (§2.2.5 above). Recall the work done by Barry Silverman using conceptual models to uncover biases in expertise. If biases have been weeded out of the expertise by the knowledge engineer, then it seems clear that, although there would be no problem for a conceptual model as a normative explanation, it could not — at least with respect to the areas of the expertise polluted by the bias — act as a causal explanation of the expertise. KA as critique of expertise is relatively rare, and so it is unlikely that this caveat rules out many existing systems, but it should be made clear.

A second caveat would attach to the work of Burton et al (and others) which shows that experts do not always recognise their expertise. Indeed, many experts do not perceive themselves as manipulating rules at all, whereas most conceptual models and KBSs tend to be rule-based. The program story still stands with respect to such systems. The conceptual model programs for some belief or beliefs that the expert has, that lead her to behave exactly if she were following the rules of the conceptual model, even if this rule-following is not performed consciously at all. The beliefs then program for causally efficacious properties as suggested above.

Of course, these notions of causal relevance and efficacy are at the centre of a great deal of current debate in the philosophy of mind, and it is fair to say that no real consensus has emerged. What our brief discussion, in the context of the theory of program explanation developed by Jackson and Pettit, has shown is that it cannot as yet be ruled out that conceptual models generated by KBS development can figure in causal explanations of expertise.

5.3.3 Conceptual Models and KBSs are Both Normatively Explanatory and Causally Explanatory

So, to conclude this chapter, we can see that conceptual models may be both normatively *and* causally explanatory. They are explanatory because they are psychological competence models (§5.2), which are themselves explanatory (§5.1). They are normatively explanatory (§5.3.1) because they describe the norms of expert behaviour, while they are causally explanatory (§5.3.2) because they program for various beliefs, which themselves program for causally efficacious properties, and are therefore causally relevant to the expert performance.

Further, because KBSs are built from conceptual models with minimal violence to the models, they can be seen as embodying those models (§5.2.4), and therefore are normatively and causally explanatory themselves.

This is a very interesting result — it is unusual to see an explanation touted as being both normative and causal (i.e. causal and non-causal at the same time). We will discuss the import of this dual nature of conceptual models as explanations in our concluding chapter, Chapter Six.

Chapter Six: Conclusion: Expertise and Artificial Intelligence

We have now completed the main body of work of this thesis, and it is the purpose of this final chapter to review the ground we have covered, and to enumerate the take-home messages that follow.

6.1 Review of Chapters One-Five

We began in Chapter One with a discussion of a general split in philosophy between top down and bottom up philosophy. Top down philosophy is concerned with aprioristic reasoning about various fields; bottom up philosophy, in contrast, is a more Baconian philosophy which takes the actual progress in those fields as the subject matter it is to evaluate. It was argued, by way of a number of examples, that most philosophy of AI has been top down in style, and because of that, that a number of interesting opportunities have been missed (which is not to denigrate the interesting results that top down philosophy has achieved). This led to the first major assumption of the thesis: **that the philosophical style of the thesis was to be bottom up**. This entailed that we would be relatively suspicious of *a priori* reasoning about AI, and would concentrate instead on evaluating an existing functioning sub-discipline of AI.

In order to do this, we needed some background to such an area. Chapter Two introduced a relatively mundane field of AI, that of model-based KBS development methodologies. We saw that such methodologies have interesting advantages for knowledge acquisition, explanation, verification and validation, and archiving. We focussed on model refinement methodologies, which move from user requirements to conceptual model to design model to implementation. Examples of such methodologies include KADS, Generic Tasks and GDMs. The aim of the thesis could then be formulated with respect to this field: **conceptual models built as a result of model-based KBS development projects (and, derivatively, design models and KBSs themselves) are explanatory of the expertise they model**.

Chapter Three accordingly introduced the idea of scientific explanation. We extended our preference for bottom up explanation in the philosophy of AI to the philosophy of explanation itself. This led us to reject the top down accounts of

explanation of many important philosophers of science, such as Hempel, Salmon, Lewis and van Fraassen. We did not dispute that these forms of explanation were epistemologically interesting, nor that they delineated and described important classes of explanation in science. But we wanted a more general account. Hence, from our bottom up perspective, we looked at a number of scientific activities that are *prima facie* explanatory, and tried to get a general account of them. This account was very context-sensitive, but we were able to list a series of conditions that an account of an explanatory practice should discuss. Our second major assumption could then be stated: **a good explanation should help a scientist to reach understanding of some phenomenon, where understanding is to be read instrumentally as rendering some achievement with respect to that phenomenon possible**. We also discussed causal and non-causal explanation, and set out one variety of non-causal explanation, normative explanation, and one variety of causal explanation, program explanation.

Since psychology is a science, the results of Chapter Three all applied to psychological explanation, in which we were interested. Chapter Four discussed some further results that applied to psychological explanation in particular. The main assumption reached in this chapter was that: **computational/information processing models in psychology can be explanatory**. This is not made trivial by the results of Chapter Three. Because we refuse (for bottom up reasons) to make hard and fast pronouncements about what can and cannot be explanatory, in one, loose, sense of 'possibility', it is 'possible' that anything could be explanatory. We are not using 'possibility' in this sense. When we say that it is possible that some practice can be explanatory, what we have to show is that there is some context in which the practice *actually is* explanatory. The result will be more significant according to the plausibility or ubiquity of the context. Following the formulation of this third main assumption, the rest of the chapter was devoted to setting out how computational models could be explanatory. We reviewed Marr's three levels, 1, 2 and 3, and Peacocke's level 1.5, and found them unsatisfactory for our purposes. We therefore formulated a new level, level 1.6.

Finally, Chapter Five attempted to put all these three assumptions together in the context of our chosen subfield of AI, model-based KBS development , to produce the main result of the thesis. §5.1 proved a lemma, that a particular type of competence model (defined in §5.1.1) called a psychological competence model (defined in §5.1.2) is explanatory of problem-solving behaviour. §5.2

used this lemma to show that conceptual models are psychological competence models, and therefore that: **conceptual models (and, derivatively, KBSs) are explanatory of problem-solving behaviour at level 1.6**.

After the main result, we developed a subsidiary thesis. §5.3 looked at the content of these models to determine the sense in which they are explanatory, and produced the surprising result that, plausibly, **conceptual models (and, derivatively, KBSs) are both causally *and* normatively explanatory of problem-solving behaviour** (at least pending further research into the whole notion of causal explanation in psychology).

That is a review of the work done in this thesis. For the remainder of this chapter, we will discuss some implications.

6.2 Artificial Intelligence as Explanatory

Our first task is to make sure that we have answered the question posed in the title of the thesis: can computational processes be regarded as explanatory of mental processes? The answer here is clearly 'yes'. We have shown that a certain class of KBS is explanatory of expertise; KBSs are computational systems, even if conceptual models are not, and so therefore we have an existence proof that computational systems can be psychologically explanatory.

Of course, after our discussions of explanation in Chapter Three, we would expect this result to be relativized to a context, and this is indeed the case. We set out, in Chapters Four and Five, various classes of people for whom this sort of explanation would be interesting: knowledge engineers of course trivially leap to mind. Others include ergonomists and people in human factors research. Indeed, one major class of people for whom conceptual models at least can be explanatory are those for whom expertise is a commodity (e.g. industrial corporations). As expertise gets increasingly expensive because of the value it adds to products, its preservation for a corporation is correspondingly important. Hence conceptual models are increasingly used, not for KBS development, but for archiving expertise; a recent article in *Fortune 500* focussed on such work in industry and business. Recall also Barry Silverman's work (1990), which uses conceptual models to explain how expertise could lead to flawed results (e.g. the Challenger disaster, the Bhopal disaster); this of course is of value for accident investigators and safety experts.

Hence although the contexts in which conceptual models are explanatory are limited, they are not vanishingly small. This is to be expected; §§5.1 and 5.2 showed how conceptual models are instances of a general explanatory strategy, decomposing a complex process into simpler ones. The basic processes are, in the case of conceptual modelling, fairly simple, possibly even simple enough to be of interest to academic cognitive psychology. Examples include *selection* (selecting an element from a class on the basis of some requirement) and *abstraction* (stripping inessential information from a proposition). Clearly such a strategy is going to *render the complex process understandable* in many contexts; this is certainly true given our instrumental characterization of understanding, but is also likely to be true given rather more traditional accounts of understanding.

6.3 Expertise in Context

We said in §6.2 that we have shown that a certain class of KBS is explanatory of expertise. However much expertise is context-sensitive (Agnew et al 1994), and does not transfer easily from the areas in which it is tried and trusted. It often consists largely of relatively *ad hoc* propositions about the domain, which are entailed by the underlying laws of the domain given common or standard initial conditions. If the conditions under which the expertise is to function change fairly radically, the expert often flounders. Therefore many people in AI do not want to make an easy assimilation between the content of the conceptual model and "what goes on in the expert's head" (specified in some internal, or context independent, way), since, for expertise to flourish, certain conditions about the expert's *context* have to obtain as well. For example, the influential theorist William Clancey sees a KBS as a model of some system or organization *in the world*, encompassing the KBS *and* its environment. The expert is seen as an '*informant* about some system in the world (therefore knowledge acquisition is primarily concerned with modeling some system in the expert's world, in contrast to modeling his mental processes)' (Clancey 1992, p.6).

The worry here is that the conceptual model inevitably includes information about the environment of the target system — or, perhaps better, is inevitably tainted by environmental influences. Hence we need to be clear about the claims we are making in the thesis. The conceptual model explains the expertise *and its operation in a wider context*. And this is not surprising — it cannot be emphasized too much that the main aim of the conceptual model is to aid the

development of an operational KBS, and the function of explanation is very much secondary. That does not mean that one could envisage a type of model suitable for KBS development that *wasn't* explanatory; rather the point is that the conceptual model promotes KBS development *because* it explains the expertise in its context. The fact that the conceptual model is rooted in the context of the exercise of the expertise means that the expertise is simply not fully understandable without reference to that context, at least for the purposes of AI. Hence that context or situation is, or at least can be, an important factor in the explanation of expertise, and one would *expect* a model of the competence of an expert to include information about the operation of that expertise in the world (or rather, one would expect a purported model of the competence of an expert, that refuses to specify what aspects of the world enable it to operate, to be regarded as incomplete).

For this reason, the conceptual models in this thesis can be seen as shedding some light on the recent debate about *externalism* in psychology, and externalist *explanation* in particular (e.g. Peacocke 1993). Conceptual models of expertise provide externalist explanations, and, furthermore, it would be impossible for a satisfactory internalist explanation of expertise to be given in AI, since AI's requirements are that the operation of the expertise in real-world contexts be explained (Clancey 1992). Furthermore, one vital component of any description of expertise is a description of its *constituency* (Agnew et al 1994), i.e. the society whose interests dictate that they defer to the expert; the way in which the expertise functions outside the area of interest of that constituency should not be covered by the description of the expertise. The expertise is not designed, developed or evolved to function outside that area of interest.

6.4 Normative and Causal Explanations

Recall from §5.3 that conceptual models can be seen as both normatively and causally explanatory of expertise. They are normatively explanatory, because they describe a practice in such a way that, if you went away and learned the model, you would be able to produce the same behaviour (though not necessarily as 'expertly'). They tell you what 'the thing to do' is in various circumstances. They also have a justificatory component, in that they contain sufficient information to allow the KBS output to be explained (justified) to the user. They are causally explanatory, because they program for certain sets of groups of beliefs, which program for causally efficacious properties.

This is an interesting result because of an often unspoken assumption in the philosophy of explanation that causal and non-causal explanations are mutually exclusive; i.e. if an explanation is causal then it cannot be non-causal (and therefore normative) and vice versa. Of course, this is not a universal assumption. Davidson's reason-giving explanations of action are normative (because the reasons rationalise the actions) and causal (1963), while Pettit's normalizing explanations are similarly normative and causal (1986). Our discussion in Chapter Five supports the Davidson-Pettit view. In §3.4, we looked at fairly anodyne definitions of these types of explanation, and §5.3 discussed conceptual models in those terms. Assuming no error in the latter section, the only routes that are apparently open to those who wish to keep the unspoken assumption are (i) to deny that normative explanation is a genuine type of explanation and (ii) to deny that program explanations are causal explanations. The former route would basically entail that a top down view of explanation is being taken; the arguments of §3.2 would therefore have to be countered. The latter route might be taken by denying either that program explanations are causal, or that they are explanations. Denying that they are causal involves asserting that causal efficacy is the only type of causal relevance; this would involve combatting the arguments of Jackson and Pettit, particularly in their (1990), where their point is reduced to its barest essentials. Denying that they are explanations would seem to rule out many apparently legitimate types of explanation; again this is a top down rather than a bottom up point of view, and would need argument.

If we accept that conceptual models are both normative and causal explanations, we have the result that experts' beliefs turn out to map very closely onto the norms of their expertise. Perhaps this is not surprising; one factor of expertise is that those who have it are, all things being equal, better at performing the task (in context) than anyone else. Hence their own practices — which may be very context-dependent or *ad hoc* — still act as a *standard* for everyone else. This thought leads us to suggest that the conceptual relevance of the result that models of expertise are both causally and normatively explanatory is larger than might first be thought. We can make the following tentative suggestion: **we can define expertise (possibly partially) as a psychological faculty for which it is possible to give an explanation which is both causal and normative**. Where causal and normative explanations coincide, there we can expect to find expertise. There may be other conditions attaching to the definition of expertise (even assuming that a precise definition could be given); nevertheless it seems

very plausible that there is a strong conceptual connection between the coincidence of causal and normative accounts and expertise. Our purpose here is only to point to the possibility — whether the condition is necessary, or sufficient, or both, or neither is a question that we leave open here.

One possible problem is that conceptual models can vary in quality as they are perceived first as normative explanations, and then causally (i.e. a model could be a good causal explanation and a bad normative explanation, or vice versa). This is to be expected, and here is one reason why. Recall from Chapter Two that conceptual models can be built with variable numbers of expert informants. Suppose first of all that a model is built using a largish number of experts, and is developed in such a way as to resolve a number of conflicts between the KA testimony of the various experts. Then the model will model 'expertise' which is demonstrably different in some areas from the expertise of each of the experts. Then it seems to follow that, to the extent that the model departs from the expertise of an expert, it is a relatively poor causal explanation of that expert. On the other hand, because the model is very representative of a wide variety of experts, it may be a very good explanation of the discipline of problem-solving in that particular domain; i.e. it may be a very good normative explanation. Secondly, consider a model built using only one expert informant (this is a more usual case). Then the model may be a very good causal explanation of the expert, but, if the expert is idiosyncratic in some ways, it may not be quite so good a normative explanation.

All that is certainly possible. Nevertheless, there will be cases where a conceptual model is a good explanation of the expertise both normatively and causally. And in those cases, our tentative suggestion does seem plausible. An expert is someone who does 'the right thing'; hence a model of what an expert does will be a normative model. But since the expert does actually do those things that are right, there will be a causal explanation too. And if he does 'the right thing' *because* it is the right thing, then we may get a coincidence of those two explanations.

* * *

The arguments in this thesis have been based on actual research in a relatively mundane subfield of AI, yet they have yielded results about psychological explanation, externalist explanation, and the nature of expertise. I take this as

evidence that philosophical argument that is rooted in real-world endeavour can be as interesting and important as highly aprioristic reasoning based on ideal definitions and thought experiments.

References

Peter Achinstein (1981) 'Can There Be a Model of Explanation?' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.136-59

Peter Achinstein (1983) *The Nature of Explanation* (O.U.P., Oxford)

Peter Achinstein (1984) 'The Pragmatic Character of Explanation' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.326-44

W.R. Adrion, M.A. Branstad and J.C. Cherniavsky (1982) 'Validation, Verification and Testing of Computer Software' in *ACM Computing Surveys* vol.14 pp.159-92

Neil Agnew, Ken Ford and Pat Hayes (1994) 'Expertise in Context: Personally Constructed, Socially Selected and Reality-Relevant?' in *International Journal of Expert Systems* vol.7 pp.65-88

Alan R. Anderson (ed.) (1964) *Minds and Machines* (Prentice-Hall, Englewood Cliffs, N.J.)

Anjo Anjewierden, Jan Wielemaker and Catherine Toussaint (1992) 'Shelley — Computer-Aided Knowledge Engineering' in *Knowledge Acquisition* vol.4 pp.109-25

S. Barthélemy, P. Frot and N. Simonin (1988) *Analysis Document Experiment F4* (KADS deliverable D3)

Izak Benbasat and Jasbir S. Dhaliwal (1989) 'A Framework for the Validation of Knowledge Acquisition' in *Knowledge Acquisition* vol.1 pp.215-33

Francesco Bergadano (1993) 'Machine Learning and the Foundations of Inductive Inference' in *Minds and Machines* vol.3 pp.31-51

G. Blaauw (1976) *Digital System Implementation* (Prentice-Hall, Englewood Cliffs, N.J.)

Margaret A. Boden (ed.) (1990a) *The Philosophy of Artificial Intelligence* (O.U.P., Oxford)

Margaret A. Boden (1990b) 'Introduction' in Margaret A. Boden (ed.) *The Philosophy of Artificial Intelligence* (O.U.P., Oxford) pp.1-21

Fergus Bolger, George Wright, Gene Rowe, John Gammack and Bob Wood (1989) 'LUST For Life: Developing Expert Systems for Life Assurance Underwriting' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.128-39

Joost Breuker, Bob Wielinga, Maarten van Someren, Robert de Hoog, Guus Schreiber, Paul de Greef, Bert Bredeweg, Jan Wielemaker, Jean-Paul Billeaut, Massoud Davoodi and Simon Hayward (1987) *Model-Driven Knowledge Acquisition: Interpretation Models* (KADS deliverable D1)

Baruch A. Brody (1972) 'Towards an Aristotelian Theory of Scientific Explanation' in *Philosophy of Science* vol.39 pp.20-31

Rodney A. Brooks (1991) 'Intelligence Without Representation' in *Artificial Intelligence* vol.47 pp.139-59

S. Bromberger (1965) 'An Approach to Explanation' in R.J. Butler (ed.) *Analytical Philosophy* (Blackwells, Oxford), pp.72-105

S. Bromberger (1966) 'Why-Questions' in Robert Colodny (ed.) *Mind and Cosmos: Essays in Contemporary Science and Philosophy* (University of Pittsburgh Press, Pittsburgh), pp.86-111

D.C. Brown and B. Chandrasekaran (1989) *Design Problem Solving: Knowledge Structures and Control Strategies* (Morgan Kaufman, San Mateo)

B.G. Buchanan and E.A. Feigenbaum (1978) 'DENDRAL and Meta-DENDRAL: Their Applications Dimension' in *Artificial Intelligence* vol.11 pp.5-24

Alan Bundy (1987) 'How to Improve the Reliability of Expert Systems' in D.S. Moralee (ed.) *Research and Development in Expert Systems IV* (C.U.P., Cambridge) pp.3-17

Tyler Burge (1986) 'Individualism and Psychology' in *Philosophical Review* vol.95 pp.3-45

A.M. Burton, N.R. Shadbolt, A.D. Hedgecock and G. Rugg (1987) 'A Formal Evaluation of Knowledge Elicitation Techniques for Expert Systems: Domain 1' in D.S. Moralee (ed.) *Research and Development in Expert Systems IV* (C.U.P., Cambridge) pp.136-45

A.M. Burton, N.R. Shadbolt, G. Rugg and A.D. Hedgecock (1988) 'Knowledge Elicitation Techniques in Classification Domains' in *Proceedings of the 8th European Conference on Artificial Intelligence* (Pitman, London) pp.85-90

A.M. Burton, N.R. Shadbolt, G. Rugg and A.D. Hedgecock (1990) 'The Efficacy of Knowledge Elicitation Techniques: A Comparison Across Domains and Levels of Expertise' in *Knowledge Acquisition* vol.2 pp.167-78

Tom Bylander and B. Chandrasekaran (1988) 'Generic Tasks for Knowledge-Based Reasoning: The "Right" Level of Abstraction for Knowledge Acquisition' in B.R. Gaines and J.H. Boose (eds.) *Knowledge Acquisition for Knowledge-Based Systems vol. i: Knowledge-Based Systems* (Academic Press, San Diego) pp. 65-77

B. Chandrasekaran (1983) 'Towards a Taxonomy of Problem Solving Types' in *AI Magazine* vol.4(1) pp.9-17

B. Chandrasekaran (1990) 'Design Problem Solving: A Task Analysis' in *AI Magazine* vol.11(4) pp.59-71

B. Chandrasekaran and Todd R. Johnson (1993) 'Generic Tasks and Task Structures: History, Critique and New Directions' in J.-M. David, J.-P. Krivine and R. Simmons (eds.) *Second Generation Expert Systems* (Springer, Berlin) pp.232-72

B. Chandrasekaran, Michael C. Tanner and John R. Josephson (1989) 'Explaining Control Strategies in Problem solving' in *IEEE Expert* pp.9-24

Jean-Pierre Changeux (1985) *Neuronal Man: The Biology of Mind* (O.U.P., Oxford)

Andrea Chierici, Maria Grazia Filippini and Marco Minati (1989) 'PORTAFOGLIO: A Portfolio Advisor Application' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.140-52

William Child (1994) *Causality, Interpretation and the Mind* (O.U.P. Oxford)

Patricia Smith Churchland (1986) *Neurophilosophy: Toward a Unified Science of the Mind/Brain* (M.I.T. Press, Cambridge, Mass.)

Paul M. Churchland (1970) 'The Logical Character of Action-Explanations' in *Philosophical Review* vol.79 pp.214-36

Paul M. Churchland (1979) *Scientific Realism and the Plasticity of Mind* (C.U.P., Cambridge)

William J. Clancey (1983) 'The Epistemology of a Rule-Based Expert System — A Framework for Explanation' in *Artificial Intelligence* vol.20 pp.215-51

William J. Clancey (1985) 'Heuristic Classification' in *Artificial Intelligence* vol.27 pp.289-350

William J. Clancey (1991) 'The Frame of Reference Problem in the Design of Intelligent Machines' in K. VanLehn (ed.) *Architectures for Intelligence* (Lawrence Erlbaum, Hillsdale, N.J.) pp.357-423

William J. Clancey (1992) 'Model Construction Operators' in *Artificial Intelligence* vol.53 pp.1-115

Andy Clark (1989) *Microcognition: Philosophy, Cognitive Science and Parallel Distributed Processing* (M.I.T. Press, Cambridge, Mass.)

L. Jonathan Cohen (1989) *An Introduction to the Philosophy of Induction and Probability* (Clarendon Press, Oxford)

Neil Cooper (1994) 'Understanding' in *Proceedings of the Aristotelian Society* supp. vol.68 pp.1-26

Robert Cummins (1975) 'Functional Analysis' in *Journal of Philosophy* vol.72, pp.741-64. Page references to abridged version in Ned Block (ed.) *Readings in the Philosophy of Psychology Volume 1* (Methuen, London, 1980) pp.185-90

Robert Cummins (1983) *The Nature of Psychological Explanation* (M.I.T. Press, Cambridge, Mass.)

Donald Davidson (1963) 'Actions, Reasons and Causes' in *Essays on Actions and Events* (O.U.P., Oxford 1980) pp.3-19

Randall Davis (1982) 'Teiresias: Applications of Meta-Level Knowledge' in R. Davis and D.B. Lenat (eds.) *Knowledge-Based Systems in Artificial Intelligence* (McGraw-Hill, New York) pp.227-490

Randall Davis (1987) 'Knowledge-Based Systems: The View in 1986' in W. Eric L. Grimson and Ramesh S. Patil (eds.) *AI in the 1980s and Beyond: An M.I.T. Survey* (M.I.T. Press, Cambridge, Mass.) pp.13-41

J.A. Deutsch (1962) *The Structural Basis of Behaviour* (C.U.P., Cambridge)

Jasbir Singh Dhaliwal and Izak Benbasat (1990) 'A Framework for the Comparative Evaluation of Knowledge Acquisition Tools and Techniques' in *Knowledge Acquisition* vol.2 pp.145-66

John Domingue, Enrico Motta and Stuart Watt (1993) 'The Emerging VITAL Workbench' in N. Aussenac, G. Boy, B. Gaines, M. Linster, J.-G. Ganascia and Y. Kodratoff (eds.) *Knowledge Acquisition for Knowledge-Based Systems* (Springer-Verlag, Berlin) pp.320-39

Hubert L. Dreyfus (1987) 'Misrepresenting Human Intelligence' in Rainer Born (ed.) *Artificial Intelligence: The Case Against* (Croon Helm, London) pp.41-54

Hubert L. Dreyfus and Stuart E. Dreyfus (1988) 'Making a Mind Versus Modeling the Brain: Artificial Intelligence Back at a Branchpoint' in Stephen R. Graubard (ed.) *The Artificial Intelligence Debate: False Starts, Real Foundations* (M.I.T. Press, Cambridge, Mass.) pp.15-43

Hubert L. Dreyfus, Stuart E. Dreyfus and Tom Athanasiou (1986) *Mind Over Machine: The Power of Human Intuition and Expertise in the Era of the Computer* (Blackwells, Oxford)

Gerald M. Edelman (1992) *Bright Air, Brilliant Fire: On the Matter of Mind* (Penguin, Harmondsworth)

Jerry A. Fodor (1987) *Psychosemantics: The Problem of Meaning in the Philosophy of Mind* (M.I.T. Press, Cambridge, Mass.)

Robert M. French (1990) 'Subcognition and the Limits of the Turing Test' in *Mind* vol.99 pp.53-65

Milton Friedman (1953) *Essays in Positive Economics* (University of Chicago Press, Chicago)

B.R. Gaines (1988) 'Knowledge Acquisition and Technology' in *Proceedings of the 3rd AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop* (Banff, Canada) chapter 8

J. Gaschnig, P. Klahr, H. Pople, E. Shortliffe and A. Terry (1983) 'Evaluation of Expert Systems: Issues and Case Studies' in R. Hayes-Roth, D.A. Waterman and D.B. Lenat (eds.) *Building Expert Systems* (Addison Wesley, Reading, Mass.) pp.241-82

Michael R. Genesereth and Nils J. Nilsson (1987) *Logical Foundations of Artificial Intelligence* (Morgan Kaufman, Los Altos)

Joseph Giarratano and Gary Riley (1989) *Expert Systems: Principles and Programming* (PWS-KENT, Boston)

Nelson Goodman (1954) *Fact, Fiction and Forecast* (Athlone Press London)

David Gries (1981) *The Science of Programming* (Springer Verlag, Berlin)

Thomas R. Gruber (1992) *Ontolingua: A Mechanism to Support Portable Ontologies Version 3.0* (Knowledge Systems Laboratory, Stanford)

R.V. Guha and Douglas B. Lenat (1990) 'Cyc: A Midterm Report' in *AI Magazine* vol.11(3) pp.32-59

Stevan Harnad (1990) 'Lost in the Hermeneutic Hall of Mirrors' in *The Journal of Experimental and Theoretical Artificial Intelligence* vol.2 pp.321-7

Stevan Harnad (1991) 'Other Bodies, Other Minds: A Machine Incarnation of an Old Philosophical Problem' in *Minds and Machines* vol.1 pp.43-54

John Haugeland (ed.) (1981) *Mind Design* (M.I.T. Press, Cambridge, Mass.)

Larry Hauser (1993a) 'Why Isn't My Pocket Calculator a Thinking Thing?' in *Minds and Machines* vol.3 pp.3-10

Larry Hauser (1993b) 'The Sense of "Thinking"' in *Minds and Machines* vol.3 pp.21-9

Larry Hauser (1993c) 'Reaping the Whirlwind: Reply to Harnad's "Other Bodies, Other Minds"' in *Minds and Machines* vol.3 pp.219-37

S.A. Hayward, B.J. Wielinga and J.A. Breuker (1987) 'Structured Analysis of Knowledge' in *International Journal of Man-Machine Studies* vol.26 pp.487-98

Carl.G. Hempel (1945) 'Studies in the Logic of Confirmation' in *Mind* vol.54 pp.1-26, 97-121

Carl G. Hempel (1962a) 'Deductive-Nomological vs Statistical Explanation' in Herbert Feigl and Grover Maxwell (eds.) *Minnesota Studies in the Philosophy of Science III* (University of Minnesota Press, Minneapolis) pp.98-169

Carl G. Hempel (1962b) 'Explanation in Science and in History' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.17-41

Carl G. Hempel (1965) *Aspects of Scientific Explanation* (Free Press, New York)

Carl G. Hempel and Paul Oppenheim (1948) 'Studies in the Logic of Explanation' in Carl G. Hempel *Aspects of Scientific Explanation* (Free Press, New York 1965) pp.245-90

C.A.R. Hoare (1984) 'Programs are Predicates' in C.A.R. Hoare and J.C. Shepherdson (eds.) *Mathematical Logic and Programming Languages* (Prentice-Hall, Englewood Cliffs, N.J. 1985) pp.141-54

Robert R. Hoffman (1980) 'Metaphor in Science' in R.P. Honeck and R.R. Hoffman (eds.) *Cognition and Figurative Language* (Lawrence Erlbaum, Hillsdale, N.J.) pp.393-423

Robert R. Hoffman (1985) 'Some Implications of Metaphor for Philosophy and Psychology of Science' in Wolf Paprotté and René Dirven (eds.) *The Ubiquity of Metaphor: Metaphor in Language and Thought* (John Benjamins, Amsterdam) pp.327-80

Frank Jackson and Philip Pettit (1988) 'Functionalism and Broad Content' in *Mind* vol.97 pp.381-400

Frank Jackson and Philip Pettit (1990) 'Program Explanation: A General Perspective' in *Analysis* vol.50 pp.107-17

Frank Jackson and Philip Pettit (1992) 'Structural Explanation in Social Theory' in David Charles and Kathleen Lennon (eds.) *Reduction, Explanation and Realism* (Clarendon Press, Oxford) pp.97-131

Peter Jackson (1986) *Introduction to Expert Systems* (Addison Wesley, Reading, Mass.)

P.N. Johnson-Laird (1988) *The Computer and the Mind* (Fontana, London)

W. Karbach, Marc Linster and Angi Voß (1990) 'Model-Based Approaches: One Label — One Idea?' in Bob Wielinga, John Boose, Brian Gaines, Guus Schreiber and Maarten van Someren (eds) *Current Trends in Knowledge Acquisition* (IOS Press, Amsterdam) pp.173-189

G.A. Kelly (1955) *The Psychology of Personal Constructs* (Norton, New York)

Philip Kitcher and Wesley C. Salmon (1987) 'Van Fraassen on Explanation' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.310-25

R.E. Korf (1985) 'Macro-Operators: A Weak Method for Learning' in *Artificial Intelligence* vol.26 pp.35-78

Henry E. Kyburg, Jr. (1965) 'Probability, Rationality and a Rule of Detachment' in Y. Bar-Hillel (ed.) *Proceedings of the 1964 Congress for Logic, Methodology and the Philosophy of Science* (North-Holland Amsterdam) pp.301-10

Henry E. Kyburg Jr. (1988) 'The Justification of Deduction in Science' in Adolf Grünbaum and Wesley C. Salmon (eds.) *The Limitations of Deductivism* (University of California Press, Berkeley) pp.61-94

R.C. Lacher (1993) 'Expert Networks: Paradigmatic Conflict, Technological Rapprochement' in *Minds and Machines* vol.3 pp.53-71

David Lewis (1986) 'Causal Explanation' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.182-206

Peter Lipton (1990) 'Contrastive Explanation' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.207-27

Nigel Major and Kieron O'Hara (1994) *Using Generalised Directive Models to Represent Teaching Strategy Knowledge: A Description of COCA* E.S.R.C. Centre for Research in Education, Development and Training Technical Report No.9

S. Marcus and J. McDermott (1989) 'SALT: A Knowledge Acquisition Tool for Propose-and-Revise Systems' in *Artificial Intelligence* vol.39 pp.1-37

David Marr (1982) *Vision* (Freeman, San Francisco)

W.A. Martin and R.J. Fateman (1971) 'The MACSYMA System' in *Proceedings of the Second Symposium on Symbolic and Algebraic Manipulation* pp.59-75

J. McCarthy (1980) 'Circumscription — A Form of Non-Monotonic Reasoning' in *Artificial Intelligence* vol.13 pp.27-39

James L. McClelland, David E. Rumelhart and the PDP Research Group (1986) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition Volume 2: Psychological and Biological Models* (M.I.T. Press, Cambridge, Mass.)

Gregory McCulloch (1986) 'Scientism, Mind and Meaning' in Philip Pettit and John McDowell (eds.) *Subject, Thought and Context* (Clarendon Press, Oxford) pp.59-94

Warren S. McCulloch and Walter H. Pitts (1965) 'A Logical Calculus of the Ideas Immanent in Nervous Activity' in Margaret A. Boden (ed.) *The Philosophy of Artificial Intelligence* (O.U.P., Oxford 1990) pp.22-39

J. McDermott (1988) 'Preliminary Steps Toward a Taxonomy of Problem-Solving Methods' in S. Marcus (ed.) *Automating Knowledge Acquisition for Expert Systems* (Kluwer, Boston) pp.225-56

Colin McGinn (1982) 'The Structure of Content' in Andrew Woodfield (ed.) *Thought and Object* (Clarendon Press, Oxford) pp.207-58

Marvin Minsky (1975) 'A Framework for Representing Knowledge' in P.H. Winston (ed.) *The Psychology of Computer Vision* (McGraw-Hill, New York) pp.211-77

Katharina Morik (1989) 'Sloppy Modeling' in K Morik (ed.) *Knowledge Representation and Organization in Machine Learning* (Springer-Verlag, Berlin) pp.107-34

Adam Morton (1993) 'Mathematical Models: Questions of Trustworthiness' in *British Journal for the Philosophy of Science* vol.44 pp.659-74

Enrico Motta, Kieron O'Hara and Nigel Shadbolt (1994) 'Grounding GDMs: a Structured Case Study' in *International Journal for Human-Computer Studies* vol.40 pp.315-47

M.A. Musen (1989a) 'Conceptual Models of Interactive Knowledge-Acquisition Tools' in *Knowledge Acquisition* vol.1 pp.73-88

M.A. Musen (1989b) *Automated Generation of Model-Based Knowledge-Acquisition Tools* (Pitman, London)

Nils J. Nilsson (1980) *Principles of Artificial Intelligence* (Tioga, Palo Alto)

J.C. Nunnally (1967) *Psychometric Theory* (McGraw-Hill, New York)

Kieron O'Hara (1993) 'A Representation of KADS-I Interpretation Models Using a Decompositional Approach' in Christiane Löckenhoff, Dieter Fensel and Rudi Studer (eds.) *3rd KADS Meeting* (Siemens AG, Munich) pp.147-69

Kieron O'Hara and Nigel Shadbolt (1993a) 'AI Models as a Variety of Psychological Explanation' in *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (Morgan Kaufmann, San Mateo, Calif.) vol.i pp.188-93

Kieron O'Hara and Nigel Shadbolt (1993b) 'Locating Generic Tasks' in *Knowledge Acquisition* vol.5 pp.449-81

Kieron O'Hara and Nigel Shadbolt (forthcoming) 'Interpreting Generic Structures: Expert Systems, Expertise and Context' in Paul Feltovich, Ken Ford and Robert Hoffman (eds.) *Expertise and Context*

Christopher Peacocke (1986) 'Explanation in Computational Psychology: Language, Perception and Level 1.5' in *Mind and Language* vol.1 pp.101-23

Christopher Peacocke (1989) 'When is a Grammar Psychologically Real?' in Alexander George (ed.) *Reflections on Chomsky* (Blackwells, Oxford) pp.111-30

Christopher Peacocke (1993) 'Externalist Explanation' in *Proceedings of the Aristotelian Society* vol.93 pp.203-30

Philip Pettit (1986) 'Broad-Minded Explanation and Psychology' in Philip Pettit and John McDowell (eds.) *Subject, Thought and Context* (Clarendon Press, Oxford) pp.17-58

Philip Pettit and John McDowell (eds.) (1986) *Subject, Thought and Context* (Clarendon Press, Oxford)

Karl R. Popper (1972) *Objective Knowledge* (Clarendon Press, Oxford)

Angel R. Puerta, Samson W. Tu and Mark A. Musen (1992) *Modeling Tasks With Mechanisms* (Knowledge Systems Laboratory Technical Report KSL-REPORT 92-30)

Hilary Putnam (1975) 'The Meaning of "Meaning"' in *Mind, Language and Reality* (C.U.P. Cambridge) pp.215-71

Peter Railton (1981) 'Probability, Explanation and Information' in David-Hillel Ruben (ed.) *Explanation* (O.U.P., Oxford 1993) pp.160-81.

Allan Ramsay (1988) *Formal Methods in Artificial Intelligence* (C.U.P., Cambridge)

Han Reichgelt (1991) *Knowledge Representation: An AI Perspective* (Ablex, Norwood, N.J.)

R. Reiter (1980) 'A Logic for Default Reasoning' in *Artificial Intelligence* vol.13 pp.81-132

David-Hillel Ruben (1990) *Explaining Explanation* (Routledge, London)

David E. Rumelhart, James L. McClelland and the PDP Research Group (1986) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition Volume 1: Foundations* (M.I.T. Press, Cambridge, Mass.)

Wesley C. Salmon (1984) *Scientific Explanation and the Causal Structure of the World* (Princeton University Press, Princeton)

Wesley C. Salmon (1988a) 'Rational Prediction' in Adolf Grünbaum and Wesley C. Salmon (eds.) *The Limitations of Deductivism* (University of California Press, Berkeley) pp.47-60

Wesley C. Salmon (1988b) 'Deductivism Visited and Revisited' in Adolf Grünbaum and Wesley C. Salmon (eds.) *The Limitations of Deductivism* (University of California Press, Berkeley) pp.95-127

Wesley C. Salmon (1989) 'Four Decades of Scientific Explanation' in Philip Kitcher and Wesley C. Salmon (eds.) *Minnesota Studies in the Philosophy of Science XIII* (University of Minnesota Press, Minneapolis) pp.3-219

Wesley C. Salmon, Richard C. Jeffrey and James G. Greeno (1971) *Statistical Explanation and Statistical Relevance* (University of Pittsburgh Press, Pittsburgh)

Tom Scutt (1994) 'The Five Neuron Trick: Using Classical Conditioning to Learn How to Seek Light' in D. Cliff, P. Husbands and S.W. Wilson (eds.) *From Animals to Animats 3: Proceedings of the 3rd International Conference on Simulation of Adaptive Behaviour* (M.I.T. Press, Cambridge, Mass.)

John R. Searle (1980) 'Minds, Brains and Programs' in Margaret A. Boden (ed.) *The Philosophy of Artificial Intelligence* (O.U.P., Oxford 1990) pp.67-88

Amartya Sen (1976-7) 'Rational Fools: A Critique of the Behavioural Foundations of Economic Theory' in *Philosophy and Public Affairs* vol.6 pp.317-44

Nigel Shadbolt (1989) 'Expert Systems — A Natural History' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.1-11

Nigel Shadbolt and A. Mike Burton (1990) 'Knowledge Elicitation Techniques — Some Experimental Results' in Karen L. McGraw and Christopher R. Westphal (eds.) *Readings in Knowledge Acquisition: Current Practices and Trends* (Ellis Horwood, New York) pp.21-33

Nigel Shadbolt, Enrico Motta and Alain Rouge (1993) 'Constructing Knowledge-Based Systems' in *IEEE Software* vol.10(6) pp.34-8

Nigel Shadbolt and Bob Wielinga (1990) '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* (IOS Press, Amsterdam) pp.313-38

Mildred Shaw (1980) *On Becoming a Personal Scientist* (Academic Press, London)

Mildred L.G. Shaw and J. Brian Woodward (1990) 'Modeling Expert Knowledge' in *Knowledge Acquisition* vol.2 pp.179-206

E.H. Shortliffe (1976) *Computer-Based Medical Consultations: MYCIN* (Elsevier, New York)

E.H. Shortliffe, S.G. Axline, B.G. Buchanan, T.C. Merigan and S.N. Cohen (1973) 'An Artificial Intelligence Program to Advise Physicians Regarding Antimicrobial Therapy' in *Computers and Biomedical Research* vol.6 pp.544-60

Barry G. Silverman (1990) 'Critiquing Human Judgment Via Knowledge Acquisition Systems' in *AI Magazine* vol.11(3) pp.60-79

Herbert Simon (1983) *Reason in Human Affairs* (Blackwells, Oxford)

Robert Sokolowski (1988) 'Natural and Artificial Intelligence' in Stephen R. Graubard (ed.) *The Artificial Intelligence Debate: False Starts, Real Foundations* (M.I.T. Press, Cambridge, Mass.) pp.45-64

Andrew Stark (1993) 'What's the Matter with Business Ethics?' in *The Harvard Business Review* vol.71(3) pp.38-48

W.R. Swartout (1983) 'Xplain: A System for Creating and Explaining Expert Consulting Programs' in *Artificial Intelligence* vol.21 pp.285-325

Andrew S. Tanenbaum (1984) *Structured Computer Organization* (Prentice-Hall, Englewood Cliffs, N.J.)

Alan M. Turing (1950) 'Computing Machinery and Intelligence' in Margaret A. Boden (ed.) *The Philosophy of Artificial Intelligence* (O.U.P. Oxford 1990) pp.40-66

Vijay Vadhwana (1989) 'A Knowledge-Based System Approach to the Synthesis of Distillation Sequences' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.263-75

Walter van de Velde (1988) 'Inference Structure as a Basis for Problem Solving' in *Proceedings of the 8th European Conference on Artificial Intelligence* (Pitman, London) pp.202-7

Bas C. van Fraassen (1980) *The Scientific Image* (Clarendon Press, Oxford)

Gertjan van Heijst, Peter Terpstra, Bob Wielinga and Nigel Shadbolt (1992) 'Using Generalised Directive Models in Knowledge Acquisition' in Th. Wetter, K.-D. Althoff, J. Boose, B.R. Gaines, M. Linster & F. Schmalhofer (eds.) *Current Developments in Knowledge Acquisition — EKAW '92* (Springer-Verlag, Berlin) pp.112-32

A. Vandierendonck (1975) 'Inferential Simulation: Hypothesis-Testing by Computer Simulation' in *Nederlands Tijdschrift voor de Psychologie* vol.30 pp.677-700

D.A. Waterman (1986) *A Guide to Expert Systems* (Addison Wesley, Reading, Mass.)

John Watkins (1984) *Science and Scepticism* (Princeton University Press, Princeton)

Bob Wielinga and Guus Schreiber (1989) 'Future Directions in Knowledge Acquisition' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.288-301

B.J. Wielinga, A.Th. Schreiber and J.A. Breuker (1992) 'KADS: A Modelling Approach to Knowledge Engineering' in *Knowledge Acquisition* vol.4 pp.5-53

Kathleen V. Wilkes (1991) 'The Relationship Between Scientific Psychology and Common-Sense Psychology' in *Synthese* vol.89 pp.15-39

R. Williams, B. Knight, P. Watts and J. Burns (1989) 'An Expert System for the Control of the Activated Sludge Process' in Nigel Shadbolt (ed.) *Research and Development in Expert Systems VI* (C.U.P., Cambridge) pp.276-87

Robert A. Wilson (1994) 'Wide Computationalism' in *Mind* vol.103 pp.351-72

James Woodward (1979) 'Scientific Explanation' in *British Journal for the Philosophy of Science* vol.30 pp.41-67