The Knowledge Engineering Review, Vol. 8:3, 1993, 201–222

Choosing a reasoning style for a knowledge-based system: lessons from supporting a help desk

ANDREW M. DEARDEN AND DEREK G. BRIDGE

Department of Computer Science, University of York, York YO1 5DD, UK

Abstract

In this paper, we present two broad styles of KBS reasoner: those based primarily on some general, explicit model of the knowledge of the domain (whether that model be expressed by heuristic rules or by a deep model of structure and function), which we term *domain model-based reasoners*; and those based primarily on a set of examples of events in the domain, which we term *example-based reasoners* (EBR), of which case-based reasoners are a subset. The aim of this paper is to guide developers in considering the trade-offs between these different styles of reasoning. We believe that this cannot be done in general, but may be possible for specific domains. Thus, the paper provides an example analysis of the usefulness of these reasoning styles. We assess the suitability of these styles against a series of requirements which we have identified that KBSs must fulfil if they are to support help desk operations. We conclude that EBR systems are more likely to meet those requirements (the analysis draws on our earlier work in Bridge & Dearden, 1992).

1 Introduction

After-sales support is increasingly important to the commercial success of many organizations. As part of this, many companies provide telephone-based support for their customers either as a free service or as a separately contracted service agreed at the time of purchase. The support varies from help and advice on the proper operation of equipment, to repair and maintenance contracts with guaranteed response times. Products from washing machines to computer communications services may be supported in this way. The organizational units which provide such support services are variously designated, for example, 'help desks', 'customer service desks', 'customer hot-lines' and 'customer support desks'. In this paper, we confine ourselves to the term help desks (HD).

We have argued in detail elsewhere (Bridge & Dearden, 1992) that the level of service that HDs offer can be improved through the deployment of knowledge-based systems (KBS) technology. In the present paper, we analyse the extent to which different styles of KBS reasoning can be expected to meet the requirements of a KBS for help desk use.

In this paper, we characterize two design 'styles' of KBS reasoning. These two styles represent considerable abstractions over the set of known KBSs. We refer to these styles as *example-based reasoning* (EBR) and *domain model-based reasoning* (DMBR). We are aware that the distinction between EBR and DMBR that we use in this paper is very coarse. We do not wish to suggest that a KBS *must* employ only 'pure' EBR or 'pure' DMBR. Rather, by considering the strengths and weaknesses of these archetypes in relation to the requirements for a KBS for HD support, we hope to guide developers in considering the trade-offs between the different styles in other application domains.

In section 2 we summarize the reference model which we developed at the University of York as part of a project sponsored by BT plc. The model, (presented in detail in Bridge & Dearden, 1992)

describes HD activities and the requirements these place on KBSs designed to support these activities. Section 3 explains the terms domain model-based reasoning and example-based reasoning. In section 4 the relative strengths and weaknesses of these styles of reasoning are assessed against the set of requirements which KBSs must fulfil if they are to support HDs. Our conclusions are presented in section 5.

2 A reference model for help desk activities

The starting point of this discussion must be an analysis of the situation in which a KBS is to be deployed, in this case a help desk. The KADS methodology, for example, lays particular emphasis on an analysis of the context in which the KBS is to be used (Breuker & Wielinga, 1987; Hickman et al., 1989; Land & Mulhall, 1989). Similarly, Clancey (1985) points out that it would be helpful to have a theory of domains to which KBS designers can refer when deciding whether KBSs are likely to be able to solve a particular problem, and against which different KBSs can be evaluated. That no comprehensive, agreed theory yet exists is witnessed by the continuing work in this area (Hickman et al., 1989; Laufmann et al., 1990).

Our own reference model possibly contributes to such a theory of domains. It arose from our initial analysis of a number of HDs (some within BT plc. and some in a variety of other organizations) and gives a reasonably abstract characterization of this subset of advice-giving domains. It was from this reference model that we were able to identify KBS requirements for *front-line* help desks (defined below), and it is against these requirements that we shall evaluate different styles of KBS reasoning.

The model, and some of the empirical work and past literature on which it is based, is described in full in (Bridge & Dearden, 1992). Here we have space for only a summary.

2.1 Definitions

By defining four pieces of terminology, as follows, we simplify the subsequent exposition:

- Help Desk (HD): a unit within an organization which, when requested, offers support (in the form of information or actions) to the day-to-day operations of consumers of certain of the organization's products.
- Help Desk Operator (HDO): a person who works for the help desk and provides a first point of contact for requests for help.
- Expert: a person to whom a HDO may field a request for help.
- Customer: a person or organization, not necessarily external to the HD's organization, who rings the HD with a request for help. Customers may be divided into those who are entitled to receive the help they request, and those that are not (qualified and non-qualified customers).

2.2 Classifying problems

A customer contacts a HD when he or she has a problem which he or she wants resolved. We identified three classes of problem (Bridge & Dearden, 1992):

- A need for *general information*, such as needing to know the cost of using the product, or needing to know the conditions of use of the product.
- A need for *task advice*, such as needing a set of 'how to' instructions which would enable the customer to achieve a particular goal using the product.
- A need for *fault-solving*, such as needing assistance when the product is behaving abnormally.

We must emphasize here that this is *our* classification of customer problems. We do not mean to imply that the customer himself or herself is able to make an *a priori* classification of his or her problem. To illustrate this, consider the case of a customer who believes he or she needs task advice

but who, it turns out, actually has a malfunctioning system and therefore requires a fault-solving service.

2.3 Help desk activities

The HDO, after investigating the request for help, may ultimately take one of the following help actions:

- *Inform*: to give the customer general information concerning the product.
- *Advise*: to give the customer advice on how to carry out some task in the normal operation of the product.
- *Intervene*: to cause some change to the state of the product in an effort to return it to normal operation. Such intervention may be initiated either by instructing the customer to perform the intervention, or by the HDO directly intervening to alter the state of the customer's product (this is particularly applicable to networked digital equipment).
- *Despatch*: to assign the problem to another member of the HD staff, usually an expert who visits the customer's site to solve the problem.
- *Refer*: to pass the problem on to another organizational unit.
- *Disqualify*: to reject the request as being outside the obligations of the HD unit.

Again, we emphasize that this is *our* classification. It is not meant to imply that HDOs straightforwardly select one of the help actions. We recognise, and have analysed (Bridge & Dearden, 1992) the very active roles taken by both the customer and HDO in formulating the problem and agreeing upon a solution. This phenomenon has been extensively studied in the past (Aaronson & Carroll, 1987; Pollack, 1985; Kidd, 1985; Coombs & Alty, 1980).

2.4 Help desk configurations

Companies arrange their HDs in one of two basic configurations:

- *Single entry point configuration*: all customer calls are initially directed to one *front-line* HD. Then, if the call needs to be referred, it is passed on to one of many *second-line* HDs, where more detailed knowledge of a particular product may be available.
- *Multiple entry point configuration*: customers have a choice of HDs which they may contact. Each HD specializes in a particular product (or small range of products).

Obviously, large businesses may have a complex mixture of these two configurations.

2.5 The need for KBS technology

Bridge and Dearden (1992) noted a trend in the organization of HDs toward single entry point configurations.

In such configurations, the front-line HD should refer as few calls as possible to second-line HDs so that the time of second-line HD staff is focused on dealing with the most complicated calls. If this configuration is to work successfully, the front-line HD must offer a service of a high enough quality that customers do not bypass the front-line HD and deal directly with second-line HDs. The deployment of KBS technology to assist the front-line HDOs can help to achieve this (Bridge & Dearden, 1992; Abraham et al., 1991; Kilhoffer & Wisely, 1990; Muns, 1990; Barr, 1991).

Laufmann et al. (1990) describe a methodology for evaluating the suitability of using a KBS in a particular domain. The methodology begins by checking that the organizational goals and objectives in developing a KBS are clearly defined. In the case of a help desk, our earlier analysis

(Bridge & Dearden, 1992), and those of other authors (Barr, 1991; Muns, 1990), show that by providing more knowledge to the help desk, it may be possible to increase the number of calls that are handled by the advise, inform, disqualify and intervene actions, and reduce the number of calls using the refer and despatch actions. This should result in reduced costs and a prompter resolution of customer problems.

The purpose of this paper, however, is not to argue that KBS technology may be applicable to HDs, but to discuss *which* KBS technologies are best suited to the task. In many organizations, two factors make this task even harder and place particular demands on the type of technology that can be deployed:

- the organization may have a wide range of products, and
- the organization may have a high rate of product change.

The presence of these factors imposes particular constraints on the design and maintenance of KBSs for HD use. This led us (Bridge & Dearden, 1992) to specify a set of requirements grouped together under the titles Reasoning and Representation Requirements, Interaction Requirements and Design and Maintenance Requirements. We shall be reviewing these requirements in detail in subsequent sections of this paper, and discussing how well different styles of KBS reasoning fulfil them. But first we must present the two basic styles of reasoning which we are considering.

3 Styles of KBS reasoning

There are many ways of classifying knowledge based systems and expert systems. A widely-accepted but rather coarse classification of expert systems, for example, is to split them by task into those that do *classification tasks* and those that do *construction tasks* (Clancey, 1985). More refined classifications by task exist; e.g., the KADS library of generic tasks (Hickman et al., 1989), or Chandrasekeran's (1983) taxonomy of problem solving types. There are also classifications by role: whether the system acts as monitor, tutor, advisor, etc. (Young, 1989). These classifications are not relevant to us here. We are concerned only with systems whose role is advisor, and whose task is a mixture of classification (working out what the customer's problem is) and construction (formulating a solution) to different degrees.

The classification we need in this paper is based less on *what* the system does, and more on *how* it does it. Even here we could get bogged down in distinctions between systems which represent knowledge using logic, those which use semantic nets, those which use frames, etc., or between systems which use probabilities, certainty factors, etc. for their uncertain reasoning. This is not our wish. We want to formulate a considerable abstraction over the class of KBS. In this paper, we take the position that there are basically two styles of reasoning in KBS (or, at least, in systems that have any real currency at the moment). We call these two styles *domain model-based reasoning* and *example-based reasoning*.

This is obviously a simplification, not least because a system might use both types of reasoning as we will discuss in section 3.2.

3.1 Domain Model-Based Reasoning

In systems which use domain model-based reasoning (DMBR), the most important factor is that the system has an *explicit* domain model. The domain model represents all sorts of knowledge about concepts in the application domain of the KBS. For example, taxonomies of types, causal relationships, co-occurrence relationships, defaults, etc. might all be captured. Various methods of representing such knowledge have been developed (Brachman & Levesque, 1985).

Faced with a problem to solve, a KBS which uses DMBR will try to apply the general knowledge of the domain model to the particular case in hand (henceforth referred to as the current case). In a sense, the current case provides instances for instantiating the general concepts of the domain

model. Then this allows new instances to be generated from (previously) uninstantiated parts of the domain model.

The most common way of implementing DMBR is to use a rule-based system. In a rule-based system all (or most) knowledge (whether it be taxonomic, causal, etc.) is represented as condition-action rules. Examples of such systems applied to the HD advice-giving domain include Abraham et al (1991), Inder (1989a), Taylor (1985) and Register and Rewari (1991). However, it is obvious that rules are but one way of capturing the knowledge of concepts and their inter-relationships needed for a domain model. Other methods include semantic nets and frames. The classification that we are proposing here does not distinguish these systems. Rather, it groups them together because in all cases their method of reasoning is to use a general domain model to instantiate specific instances of concepts.

Before proceeding, we feel it is important to remove the potential for confusion that might arise from our use of the term *domain model-based reasoning*. A term that is in common use is *model-based reasoning* (Steels, 1986). This is used to describe systems which possess a 'deep' model of their domain. By 'deep' is meant a model which explicitly represents all structure and function of the modelled domain in detail.

It should be clear that *model-based reasoning* is for us a sub-class of DMBR. Model-based reasoning is a deep form of DMBR, but we use the term *domain model-based reasoning* to embrace systems which carry out shallower forms of reasoning too. What unites both is that an explicit domain model is used.

For a comparative analysis of approaches to the construction of domain models see the work of Keravnou and colleagues (Keravnou & Washbrook, 1989; Keravnou & Johnson, 1986; Johnson & Keravnou, 1988). The classification we are proposing groups all the systems reviewed in Keravnou and Washbrook (1989), Keravnou and Johnson (1986) and Johnson and Keravnou (1988) as DMBR systems.

Whether the domain model allows deep or shallow reasoning depends on how detailed a domain model can be built. In systems diagnosing faults in digital systems, our understanding of such systems can be detailed enough to allow the construction of a model which captures all the relevant concepts, inter-relationships and precise conditions of causality, etc. However, in most medical domains, for example, our understanding of the underlying causal processes is usually not sufficient to support such a detailed model. Instead, the rough abstraction of causal processes (compiled knowledge) which simple rules coupled with an uncertainty mechanism can afford may be adequate to carry out the task of diagnosing an illness and ranking a set of possible treatments.

Whether the help desk application domain requires shallow or deep models is an open question, depending on the topic of the advice. Help desk systems of which *expertech* (Abraham et al., 1991) is an example, have indeed been constructed using the shallow DMBR style.

3.2 Example-Based Reasoning

In the same way that our term Domain Model-Based Reasoning embraces a range of types of system, so too is our term Example-Based Reasoning supposed to encompass many systems. Possible terms which it might embrace are Case-Based Reasoning, Analogical Reasoning, Memory-Based Reasoning, Experience-Based Reasoning, Instance-Based Reasoning and Prototype-Based reasoning.

In EBR systems the central part of the system's knowledge is contained within a set of instantiated previous cases, rather than in an explicit, complete domain-model. Within our proposed classification, so-called case-based reasoning (CBR) is one particular form of example-based reasoning, which is currently attracting considerable attention (Kolodner, 1988; Hammond, 1989b; Bareiss, 1991).

In CBR, to deal with the current case, the system selects a previous case from its case base, and tries to apply the solution associated with the previous case to the current case. In its simplest form, the previous case chosen is the nearest neighbour to the current case, and the neighbour's solution

is used directly (e.g. Kibler & Aha, 1988). In more complex manifestations of CBR, there may be a more complex matching process and various means of modifying the solution of the previous case before applying it to the new situation (e.g. Hammond, 1989a; Hinrichs & Kolodner, 1991).

In addition to CBR other methods of EBR can be identified.

- *Analogical reasoning*: attempts to transfer knowledge from one domain (the source domain) to another (the target domain) (Prieditis, 1988). The knowledge required to perform this operation is an analogical 'hint' (Greiner, 1988) which links an element, or elements of the source and target domains. Some analogical systems store their knowledge of the source domain in the form of examples, and would therefore be covered by our description of EBR. Others store their knowledge of the source domain in the form of a domain-model (Seifert, 1989; Jones, 1989; Carbonell, 1986).
- *Prototype based reasoning*: this is closely related to CBR but stores elements of a previously seen case, abstracted as general prototypes, which can then be used to analyse new examples (e.g. Rajamoney and Lee, 1991; Thompson and Langley, 1989).
- *Schema based reasoning*: this has been presented as a generalization of CBR, in which groups of examples are abstracted to form partial domain models (models of a particular part of a domain) named schemas (Turner, 1989). Knowledge in a schema is expressed as a declarative set of rules about that part of the domain which can be used for problem solving. A matching process selects the 'best' schema to apply to any particular problem solving situation.

CBR and EBR in general are newer than Domain Model-Based Reasoning, and hence in a greater state of flux. For example, the nature of the similarity metrics that should be applied when looking for a match of cases is an area of ongoing research (Bareiss et al., 1991; Bareiss, 1989). Equally, exactly what is stored by way of a previous case is relatively free. For example, Carbonell (1986) suggests that in many domains one should not simply store symptoms and solutions, but one should also store some record of how, in the previous case, the solution was derived from the symptoms.

We have presented a very 'black and white' distinction between EBR and DMBR systems. Our description of EBR emphasizes the use of previous examples in solving new problems, and noted that the central part of an EBR system's knowledge is contained in its cases. However, we are aware that elements of a domain-model may also form part of an EBR system. For instance:

- If the user's language differs from the language used to index the previous cases then domain knowledge may be applied to perform a translation between the two languages. The CBR system HYPO (Ashley & Rissland, 1988b) demonstrates this use of domain knowledge.
- It is possible that the method used to determine the similarity between cases is not sufficiently sensitive to guarantee the selection of the correct case. To confirm a choice some element of domain knowledge may be attached to the previous case to verify the selection. This method is used by the CASCADE system (Simoudis & Miller, 1991).
- Adaptation of cases may require some knowledge of the domain to be encoded as in the CHEF adaptive planner (Hammond, 1989a).

Our distinction is further blurred by considering two research strands which seek to exploit the strengths of both reasoning styles. For example, in derivational analogy (Carbonell, 1986) the system may have sufficient domain knowledge (in principle) to solve any problem from the domain-model alone. Case retrieval can then be used to support focusing, and thus reduce the need for weak search methods. Alternatively, a CBR system may abstract away from its cases to generate prototypical descriptions (cases) (Bareiss et al., 1991). These might be regarded as forming a part of a domain-model rather than being at the same level as cases.

Many of the EBR systems designed for HD use (e.g. Kilhoffer & Wisely, 1990; Shafer, 1991) have emphasized the matching process using a large case base, indexed by readily identifiable features of the cases. In so doing, these systems have ignored issues of case modification, i.e. adapting previous cases' solutions to the current case, and the complexities of translation between a

problem statement and the terms used to index the stored cases. These systems are examples of the 'pure' EBR style.

In conclusion, we repeat the point we made in section 1. The distinction we are drawing between EBR and DMBR is a coarse one. A system need not deploy only 'pure' DMBR or 'pure' EBR. Rather, our analysis aims to inform trade-off decisions in KBS development.

4 Which style of KBS reasoning suits help desk support applications?

We have now mapped out the activities which HDs carry out and two basic styles of KBS reasoning. We can now look at the requirements which HDs must fulfil.

We emphasize here, as we have emphasized before (Bridge & Dearden, 1992), that we are looking at support for *front-line* HDs. The requirements for second-line HDs might be very different. Front-line HDs act as a filter: they solve the simplest problems and pass the rest on to second-line HDs. We want KBS technology to increase the proportion of calls that can be regarded as trivial and can therefore be solved by the front-line HDs. We also want the technology to ensure that calls that need to be referred to a second-line HD get referred to the right second-line HD.

In Bridge and Dearden (1992) we described the requirements for a KBS under three headings:

- Reasoning and representation requirements
- Interaction requirements
- Design and maintenance requirements.

Within these headings we identified a large number of detailed requirements. We use the same headings and the same requirements below.

4.1 Reasoning and representation requirements

Given that we are focusing on front-line HDs, we can see that many of the reasoning and knowledge representation requirements are likely to be quite different from those that might apply to a second-line HD.

4.1.1 Having a sense of its own scope

No KBS that we know of can truly be said to have a sense of its own scope to enable it to reject problems that fall outside of its expertise. Any system, whether it uses DMBR or EBR, when presented with a problem to solve, will solve it as if it falls within its own scope. A system for diagnosing glaucoma faced with a patient who, in reality, has cataracts will (as likely as not) find evidence for some form of glaucoma, e.g., ranking forms of glaucoma, and proposing some treatment for these.

More extreme cases may not be so bad. For example, the same glaucoma KBS faced with a patient with heart problems is unlikely to conclude anything with enough certainty (its various metrics will fall below some threshold) so the patient might be rejected.

The MDX architecture (Johnson, 1985) attempts to address the problem of peripheral cases by emulating the way in which a community of specialists might co-operate to solve a problem. Within MDX each specialist begins by checking whether the problem lies within its domain. However, as Keravnou and Washbrook point out

> It is not clear though whether MDX can recognise if a case is peripheral to its entire 'expertise'. (Keravnou & Washbrook, 1989, p.223)

It is usually said that it is up to the human user to determine usage of KBSs. This is one reason why KBSs are now almost invariably seen as adjuncts to, or helpers for, human experts or near-experts (e.g., a physician, not a patient, would use the glaucoma KBS) so that considered use is made. We think that this applies just about equally to DMBR and EBR: the human must determine use.

However, EBR may have some advantages over DMBR for this purpose. An EBR system that has a good similarity metric should not only be able to give an indication of the relative similarity

between the current case and different previous cases, but can also give an indication of the *absolute* similarity of the current case to any previous case. If the similarity of all the previous cases to the current case falls below a certain threshold, this may be interpreted as an indication that the current case is beyond the scope of the system. Of course, this argument is entirely dependent on the discriminatory power (Maybury, 1990; Swaminathan, 1988) of the similarity assessments used, which is still an open field of research (Bareiss et al., 1991; Bareiss, 1989).

One might wonder why, if we believe that an EBR similarity metric can help to delimit scope, we do not also believe that the sort of measures of uncertainty (certainty factors, Bayesian probabilities, etc.) used in DMBR systems could do the same job. We would argue that some DMBR architectures may well rival EBR systems in their ability to reject cases as outside their scope, but others fare less well.

The DMBR systems which rival EBR systems are those such as PROSPECTOR (Duda et al., 1979; Johnson & Keravnou, 1988) and CASNET (Weiss et al., 1978; Johnson & Keravnou, 1988), where the initial evidence elicited from the user about the patient can immediately be used to access *all* the hypotheses in the system. If this initial evidence does not immediately give a 'reasonable' level of certainty to one or more hypotheses, then the problem can be quickly rejected. These systems may be regarded as using a *differential model of diagnosis*.

DMBR systems which pursue only one hypothesis at a time (such as those in the MYCIN family) really need to explore exhaustively the whole search space before a case can be rejected. Thus they do not have a sense of their own scope. We will use the term *single operator model of diagnosis* to refer to such systems.

We do not wish to suggest that simple 'speed' of rejection is the main criterion here, but we do feel that systems which entertain and can compare multiple hypotheses simultaneously will ultimately provide the basis for systems that have a sense of their own scope, while those that consider hypotheses serially do not model this human faculty adequately.

Nevertheless, what we have said under this heading is speculative and needs much further research.

4.1.2 Dealing with product teething problems

As new products are released or existing products are modified, which as we indicated earlier is liable to happen a lot in many advice-giving domains, then 'teething problems' are introduced. These are different from other problems that arise. Other problems tend to fall into two classes:

- *The known problems*. These are known 'bugs' that have not been fixed, the design problems which cause users difficulty, but cannot be easily solved except by a lot of training or major changes to the system, the hardware reliability problems that will always be with us, etc.
- *The rare problems*. These are particularly unusual co-occurrences of circumstances that give rise to problems or bugs deep in the system which crop up only on rare occasions.

Teething problems are different from both of these. They are bugs or difficulties of use that arise when a change is made or a new product is launched. They cause enough people enough trouble that the product development team is quickly called in to iron out the problem e.g. fix the bug. Thus, these problems might cause a good deal of trouble for a short period, perhaps one or two weeks, during which time a temporary solution (e.g., a clerical or procedural fix) might be established. Meanwhile, a more permanent fix is made, after which the problems disappear.

Deep reasoning (i.e., deep DMBR) has been proposed as a possible method for dealing with novel situations:

> If the task involves situations that the practicing experts have never seen before, it is probably a candidate for deep reasoning or model based techniques. These approaches are typically expensive. (Laufmann et al., 1990).

In fact, deep reasoning is not a viable option in the case of product teething problems. The problem is that if the deep model is already able to find the bug, then it is likely that the bug would be known

about before the product's release since the deep model is a reflection of the KBS engineer's understanding of the behaviour of the product. This means that the bug is not a teething problem but a known bug.

It would therefore seem that the only way we have of dealing with product teething problems is to wait for them to arise and then update the KBS with some knowledge of the teething problem and its temporary solution. When the teething problem is properly resolved, this extra knowledge can be removed from the KBS. Thus dealing with teething problems involves making changes to the KBS. It follows that the ability to cope with teething problems is closely related to the question of maintenance. We will return to this problem in section 4.3.2.

4.1.3 Dealing with major disruptions

After a major disruption (e.g., a damaging storm or fire) the calls to a HD may well increase. It is undoubtedly the case that many problems might look like normal ones (e.g., 'the screen of my terminal is blank') but the solution is now radically different (e.g., the HDO would not say, as he or she normally would, that the customer should press CTRL-Clear or whatever, rather he or she would tell the customer to sit tight and wait for the computer service to be restored, and maybe to follow some alternative procedures in the meantime, e.g., to write customer orders onto a notepad).

All KBSs are likely to be equally fragile in these cases to the point where it might even be better for the HDO to stop using the KBS. If the KBS *is* to be useful it will need to be updated temporarily in some way, hence, as with product teething problems, this problem relates closely to the problem of KBS maintenance. However, what is clear is that these sorts of circumstances need the common sense of the human operator.

Three aspects of CBR systems (as examples of EBR) may give some advantages here. Firstly, in CBR systems, the ability of the operator to override recommendations of the system and ask for other recommendations is greater than in typical DMBR systems, not least because DMBR systems' dialogues are usually system-initiated. Secondly, the role of an EBR system (Young, 1989) is typically more that of an active memory of previous events rather than a diagnostic expert. This may also make the user of an EBR system more likely to ignore the recommendations made. (We shall return to the question of dialogue requirements in the next section.) Finally, major disruptions typically call for analogical reasoning (Kedar-Cabelli, 1988). We would not wish to suggest that a KBS for help desk use should make extensive use of analogical reasoning. However, if human operators can be supported in searching for previous analogous situations, then the human operators are well equipped to perform the more difficult task of mapping the old solution onto the new situation.

4.1.4 Speaking the customer's language

The requirement that the system should 'speak the same language' as the HDO obviously has much to do with interaction. However, we have included it here in the reasoning and representation section because the ability of a system to communicate at the right level in the right terms has much to do with its knowledge representation.

Those DMBR systems which represent knowledge and reason with it at only one level run the risk that this level is not right for the operator. This might be particularly so in deep systems, where detailed knowledge of the structure and function of entities in the domain are modelled, since this model may not be at the correct level for communication.

To enable DMBR systems to communicate better, systems have been developed that represent the domain knowledge at several layers in detail, e.g., ABEL (Patil et al., 1981). Other systems operate by separating knowledge about findings in the domain from diagnostic reasoning knowledge. This allows an operator to enter data about a particular case at any level that is expressed in the model (Keravnou & Johnson, 1986; Keravnou et al., 1989). Much of this work has addressed the aim of producing explanations of expert system reasoning in natural language (e.g. Swartout & Smoliar, 1987; Neches et al., 1985).

EBR systems are likely to face the same problems as DMBR systems which only use one level of knowledge representation, since a particular level of representation may be used to conduct similarity assessment. If the level of representation used for similarity assessment is different to that used for interaction with the user, then some form of translation must take place between the two. It may be important to ensure that the mechanism used for translation can also support explanation of the choices that the system makes. Architectures similar to those proposed for DMBR in Keravnou and Johnson (1986) may be required to allow EBR to meet this requirement. As yet, little work has been published addressing the problem of communication for EBR systems.

4.1.5 Shallow but broad reasoning

It is clear that for most calls to the help desk, and especially for fault solving requests, diagnosis is required only to the level at which appropriate advise, inform, disqualify or intervene actions can be identified or eliminated, and/or the best unit to refer the problem to can be identified. If a problem has been identified which cannot be repaired by the remote intervention of the HDO, then the customer will have to wait until either an expert arrives at the site, or another organisational unit is able to take on the referred problem. Further diagnosis at the help desk in these circumstances will not result in any better response to the customer (i.e., faster resumption of normal service).

It could be argued that the HDO should attempt to do further diagnosis in the case of a despatch action to ensure that the despatched expert is properly equipped with tools and spares to complete the job. This would indeed be useful, but has associated costs in that longer conversations at the help desk may mean other customers waiting for help. In handling despatch, careful trade-off decisions must be made between the time spent on diagnosing the fault and the danger that the expert would need to make repeated trips. One way in which to resolve the conflict between providing extended diagnosis before despatching an expert and wishing to answer other customer calls quickly, would be to increase the use of referral actions and decrease the frequency of the despatch action. The further diagnosis could then be conducted by the second-line unit, releasing the front-line HDO to answer new calls.

The need for shallow but broad reasoning is taken in the DMBR literature as being a negative indicator for the successful deployment of DMBR systems. for instance Hickman et al. (1989) state that:

> Domains that are deep and narrow are better hosts for KBS than domains that are broad and shallow. (Hickman et al., 1989, p. 63).

Clearly, DMBR systems that use 'deep' reasoning models are unlikely to be appropriate. Constructing such models for a wide, ever-changing range of products is probably not realistic. DMBR systems based on a shallower model (e.g., heuristically-based systems) may provide a more appropriate depth of reasoning; however, we tend to the view that even shallow DMBR is not suited to an application requiring such broad knowledge. Building a heuristic domain-model for every product, and updating the model in response to repeated changes, would require a large amount of work. DMBR systems are better suited to well-defined, well-understood, well-delimited and stable areas of knowledge. The sheer extent of the front-line HD application domain is generally going to be too great.

It is here that EBR systems should fit the bill. The reasoning conducted by some case-based reasoners can be regarded as quite shallow: they search for a previous case similar to the current one and then recommend that the user applies the previous case's solution to the new problem (Shafer, 1991; Kibler & Aha, 1988). This reasoning may be given additional depth by building in a verification step before applying the solution (Simoudis & Miller, 1991), or by including some simple adaptation to the retrieved solution (Bain, 1986). Thus, EBR systems can be developed to provide the required shallow reasoning.

To provide an EBR system with a reasoning capability over a broad domain will require careful consideration of the indexing terms and similarity metrics that are used to discriminate between

cases. One difficulty that may arise if the indexing terms are not sufficiently discriminable (Swaminathan, 1988) is in partitioning sets of cases that appear similar, but require different responses. However, if the interventions required are not potentially damaging it might be reasonable simply to apply them in succession, or if the choice is highly sensitive this would indicate a need to refer the problem to an expert. Alternatively, the use of domain-model fragments as in the CASCADE system (Simoudis & Miller, 1991) may alleviate this problem. Such mechanisms may allow an EBR system to finesse some of the problems of working in a very broad domain.

4.1.6 Refining diagnosis to the right level of detail

The system should be able in all cases to come up with an appropriate help desk action in response to fault solving requests: disqualify, despatch the right expert to the right site, intervene, or a referral to the right second-line HD.

Well-designed systems of either kind ought to be able to do this. Provided the top-level goals in a DMBR system are at this level, a DMBR system will satisfy this requirement. Provided the actions associated with previous cases in an EBR system are at this level, an EBR system will satisfy this requirement.

4.1.7 Reasoning in the presence of incomplete and unreliable data

Any discussion of uncertain reasoning is contentious. We cannot hope to cover all viewpoints here, and so we keep our discussion even briefer than we might otherwise be able to do.

A number of techniques have been applied to the problem of reasoning with uncertainty in DMBR systems. Most of these involve the propagation of numbers, representing some measure of confidence in input data and intermediate propositions, throughout the system to arrive at some number representing a level of confidence in a conclusion. These methods include approximate Bayesian reasoning (Duda et al., 1979), certainty factors (Shortliffe, 1976) and fuzzy logic (Zadeh, 1965).

A debate rages over the use of these measures. Criticisms made (which may not apply to all manifestations of the numerical approaches) include: that the numbers are often 'synthetic' as they rely on an expert's subjective evaluation; to work properly assumptions about the independence of the units of knowledge have to be made, where these assumptions may rarely hold true in practice; the exact numbers used do not matter very much, which suggests that something more qualitative might be sufficient; and the values assigned to different parts of the knowledge base may be inter-dependent, thus making maintenance of the knowledge base even harder than usual.

In EBR approaches, the presence of a large number of exemplars may allow statistical techniques to be rigorously applied to deal with some of these problems. In particular, it might be possible to use only analytic measures. However, difficulties may still arise regarding the confidence in the data collected for individual cases.

There is now a growing body of work which looks at more qualitative approaches to uncertain reasoning (see, for example, the work of Fox et al., 1990, or for an overview of the field see Travé-Massueyès et al., 1992) and this might be applicable to both types of reasoning although for the moment it has been deployed only in DMBR systems.

Our feeling at this stage is that neither style of reasoning possesses a clear advantage over the other when it comes to dealing with uncertainty.

4.1.8 Entertaining multiple hypotheses and switching attention between them flexibly

Switching attention flexibly between multiple hypotheses is essential to support HD operations. This is a consequence of the requirement to support the mixed-initiative interaction between three agents: the customer, the HDO and the KBS (see section 4.2.3). The customer or HDO may wish to volunteer both new facts and new hypotheses.

A system can only fulfil this requirement if its hypotheses are explicated in such a form that the user of the system can refer to them, and the system allows a mixed initiative meta-dialogue concerning these hypotheses.

For DMBR systems to support such switching between multiple hypotheses they must use a differential model of diagnosis (see section 4.1.1). Clearly, systems based on a single operator model of diagnosis can only entertain one hypothesis at a time. Further, to allow multiple hypotheses to be considered simultaneously, DMBR systems must support some forward-chaining from input, using the input observations to update the level of certainty associated with each hypothesis. However, even in systems which offer these possibilities, the user often has no direct control over which hypotheses are being entertained, which are being focused on and the timing and target of a switch. Their only influence is indirect: by supplying facts whose certainties affect the certainties of the hypotheses. An exception is NEOMYCIN, in which it is explicitly possible to add hypotheses to the differential, thus effecting changes in the focus of the system (Clancey & Letsinger, 1981; Keravnou & Washbrook, 1989).

Much the same can be said of EBR systems. There is a close parallel between the forward-chaining process we have just described and the processing that goes on in an EBR system: user-supplied facts activate previous cases (the hypotheses) and then one of these may be chosen for further investigation, i.e., questions are asked to confirm or disconfirm the hypothesis. The user obviously has little direct control again.

Having doubted EBR's ability to satisfy this requirement, we note that in fact certain EBR systems might overcome this problem. We have designed and prototyped a case-based memory system which has a different style of interaction to it, which may prove able to satisfy this requirement. The interaction is much more navigational: the user navigates through a case base which has some rudimentary structure imposed on it. It becomes easier to see several hypotheses on the screen, to look at closely related hypotheses, and even to move off to look at a completely different area of the case base (Dearden, 1993).

In principle, it seems possible for both EBR and DMBR systems to support the requirement for flexible switching between hypotheses, but the nature of a hypothesis in EBR (a previous example or perhaps a set of examples, or an abstraction over such a set) commends itself as a method of explication which is easily understood by the customer and HDO, and would provide a simple communication token for the required meta-dialogue.

4.1.9 The system should be able to support some form of 'what-if' reasoning

This is clearly related to the ability to entertain multiple hypotheses and switch flexibly between them. Thus the comments in the previous section are all relevant. However, the notion of what-if reasoning is more than simply the system supporting multiple hypotheses about a single world situation, it suggests the creation of alternative descriptions of the world by the user.

Various approaches to this form of reasoning have been devised in DMBR research, particularly the use of contexts and environments (DeKleer, 1986) to reason about a set of possible worlds. However, to apply this form of reasoning in a DMBR system using a truth maintenance system would usually involve 'pretending' to the system that some fact was true in the current problem solving episode, and then later retracting that fact.

EBR might handle this problem in the same manner. However the EBR approach is simpler to implement in that it is not necessary to record for each fact the set of all possible contexts in which it is valid, merely to record the variety of views of the current case, and where that places it relative to other cases in the case base. Alternatively, our own prototype system characterizes the problem of finding a previous example as one of skilled search and provides an interface which aims to co-operate in that search. In this system it is possible to construct multiple views of the current case, represented by collections of facts about the case, and then consider the position of any given view within the case space. We believe that this offers a more flexible method of what-if reasoning than that provided in DMBR systems.

4.1.10 Traces of the interaction for second-line HDs

In cases of referral, the second-line HD needs to know the avenues that have already been explored. Therefore, when we talk of traces here we do not necessarily mean the traces of

reasoning which are conventionally given as 'explanations' of a system's conclusions for a hypothesis or reasons for asking a question. The reasoning of HD advisory systems is probably so shallow that good explanations are not really necessary.

Rather, we simply require the HD KBS to show what information has been collected, what actions have been taken, and what results observed. Both DMBR and EBR systems can provide this.

In fact, this may be an area where DMBR systems have an advantage over EBR systems: they might be able to present this information in a more logical way—their more highly constrained interaction and reasoning will impose a structure on the dialogue and hence on the trace of the dialogue.

4.2 Interaction requirements

There are two fairly minor interaction requirements and one major one. We will look at the minor two first.

4.2.1 Integrating with other call management software

We need to ensure that our HD KBS interfaces with other HD software, such as that which logs calls and keeps accounts. There is nothing intrinsic to EBR or DMBR systems that should stop us from interfacing them with other software.

4.2.2 Moving onto the next call

The HDO should be able to move onto the next call even though the present call has not been fully resolved.

In principle, this should be easy to implement in both types of system. However, it can in fact be quite troublesome in DMBR systems. Storing the state of a DMBR system until resumption of its processing involves storing all instances (facts) that relate to the current case. This means storing not just the facts supplied by the user, but also those instantiated through reasoning on the domain model. If the latter were not stored, then resumption will involve re-doing all the reasoning that has been done before. This would take an unacceptably long time while the HDO is on the 'phone to the customer.

On the other hand, in an EBR system, only facts of the present case need to be stored. It should be quite quick and easy to reapply the pattern matching algorithm to take the user back into the same part(s) of the case base.

One minor rider should be place on this argument: if a new case is added to the EBR system in the intervening period, then a re-evaluation of the metrics might result in different previous cases being considered when the problem solving session is resumed. On the one hand, this makes the system's behaviour somewhat unpredictable for the user, but on the other hand, it is to be hoped that adding more information to the case base should result in improved problem solving performance, not deterioration. This situation is equivalent to a change in the domain-model of a DMBR system. If such a change had taken place in a DMBR system, then resumption of the problem solving session would involve repeating all the reasoning that had been done before.

4.2.3 Supporting mixed-initiative interaction

We explored the requirement for mixed-initiative interaction in great detail in Bridge and Dearden (1992). Here we take a higher-level view. Our main observation was that we have an interaction between three agents—the customer, the HDO and the KBS—the likes of which it is hard to imagine arising between three human agents.

A KBS must not artificially constrain the dialogue that goes on between the customer and the HDO. Indeed, it is almost futile to hope that a KBS could be used to *prescribe* the dialogue that the

human agents engage in over the 'phone: the customer will feel free to volunteer facts and hypotheses at any stage, irrespective of when the system wants them (if it wants them at all). The HDO must be able to record this information in the system and have it taken into account immediately. He or she must not be forced to record it on a scrap of paper until the system explicitly asks for it. This places a very strong requirement for mixed-initiative interaction on the KBS.

We also note that in the three way consultation between customer, HDO and KBS, the HDO should exercise considerable control as it is the perceived competence of *the HDO* that will determine the willingness of the customers to use a front-line HD service. Roth et al. (1988) make the distinction between the use of KBS as prosthesis (making up for some deficiency in the user) and as instruments to enhance the powers of a competent user. In a design based on prosthesis, control rests mainly with the KBS and the user works as a data gatherer for the system. It is our view that if a system based on a prosthesis design is deployed to support the work of a HD, this will undermine the customer's perception of the competence of the HDO. Indeed the HDO may appear as an obstruction in the communication between the customer and the available expertise (the KBS). We therefore require a KBS which will support a mixed-initiative interaction where the HDO is able to steer the dialogue with both the customer and the system.

This requirement was certainly not satisfied by first generation DMBR expert systems. These systems almost invariably had system-initiated dialogues: questions were asked of the user in a predefined order, to which the user responded. Beyond responding to questions and occasionally asking for explanation of conclusions and justifications for questions, the user was passive. He or she could not volunteer facts and hypotheses.

Second generation DMBR systems may address these problems by separating out general knowledge about the domain, domain reasoning knowledge and knowledge about diagnostic strategy (Keravnou & Johnson, 1986; Keravnou & Washbrook, 1989). The separation of general domain knowledge allows some opportunistic forward chaining so that observations may be entered at any point in a consultation, and immediately 'taken into account' in the diagnostic process. However, the fundamental aim of this research is to allow the DMBR system to model a competent expert in the chosen domain, and for this reason the DMBR system is given control over the diagnostic strategy. The user of the majority of such systems may be able to interject relevant observations, but is not able to direct the system to consider an alternative hypothesis (Keravnou & Washbrook, 1989). Work by Pople (1985) has specifically addressed the problem of allowing a user to direct the diagnostic path followed by the CADUCEUS system for internal medicine. This would be a significant move towards the kind of interaction required for HD support.

For EBR systems little work explicitly considering the nature of the interface has yet been conducted. However, we believe that the EBR style of reasoning may lend itself more easily to meeting the requirement for the style of interaction we have described. We believe that the way in which the hypotheses of an EBR system are explicated (individual cases) is easy for a HDO to understand and, we hope, manipulate. This should allow the HDO both to enter statements about the current situation and to instruct the KBS to consider particular possible past cases. Studies of co-operative search techniques for database access (Stenton, 1987; Motro, 1987; Fischer, 1990) may provide guidance in the design of appropriate interfaces for EBR systems.

The prototype EBR system that we have constructed for HD use allows an HDO to update the current case description, hypothesize alternative descriptions, or look at sets of past cases at any point during the consultation. The characterization of the HDOs problem as one of co-operative search in a case base gives the HDO an active and controlling role in the dialogue with both system and customer.

4.3 Design and maintenance requirements

Design and maintenance requirements were the most important requirements that BT plc. had for a KBS for front-line HD support. We believe that the majority of front-line HDs in other organizations will have similar design and maintenance requirements.

4.3.1 Limiting the amount of knowledge acquisition

The systems should be easy to build without inordinate amounts of initial knowledge acquisition.

It almost goes without saying that DMBR systems do not satisfy this criterion: building a domain model is time-consuming, difficult and has, in most cases, to be done by a professional knowledge engineer (unless the system is trivial). The deeper the model, the worse this gets and the broader the product range, the worse it gets. Kidd (1987) argues that to develop a comprehensive solution to the problem of constructing a domain model would require answers to some of the fundamental questions of psychology, philosophy and linguistics, such as the precise relationships between language, knowledge and human problem solving.

Having constructed a suitable model of the knowledge required, further problems may arise in moving from a model to an operational DMBR system. Ward and Sleeman (1987) highlight the following problems in actually constructing a DMBR system using a KBS development tool:

- Choosing whether to represent a particular element of knowledge as a property of an object or a new object in its own right.
- Choosing whether to represent a piece of knowledge as a set of separate rules, or to build one complex rule with many antecedent and consequent clauses.
- Developing control structures to model the tactical and strategic problem solving knowledge of a domain expert.
- Modelling uncertainty in reasoning via the use of certainty factors.

On the other hand, advocates of EBR systems stress the limited engineering required to produce them (Shafer, 1991; Hennessy & Hinkle, 1992). Major elements of EBR systems may be generated automatically. Many organizations already have databases of past cases which, with some minor changes of format, might be usable at least as an initial case base[1]. The development of suitable similarity measures for an EBR system may also be automated (Aha, 1991; Aha et al., 1991) by making use of statistical analyses of the case base.

We would not claim that EBR approaches remove the need for knowledge acquisition altogether. Choosing the attributes which are used to describe cases may require specialised knowledge engineering skills. However, the discovery of these attributes would also be required before a domain model for a DMBR system could be constructed.

As we mentioned above in section 3.2, some elements of a domain model may be used at other points within an EBR system. In section 3.2 we mentioned three possible points at which domain model elements might be used: in translating between an input language and a set of indexing terms, in similarity assessment and in case adaptation. Another problem for some domains reported by Pearce et al. (1992) is that prior case records may not be in a standard format and may not contain all the information needed to represent them in the system. This was true of some of the help desks we studied in (Bridge & Dearden, 1992). To correct this deficit may involve obtaining more information about the cases, or constructing domain model fragments to allow the system to make use of this incomplete information. We would argue that such knowledge fragments will represent a (proper) subset of the knowledge required for a full domain model, and we hypothesize that the knowledge elicitation required to obtain such fragments will also be a proper subset of the knowledge acquisition required to build a full domain model.

4.3.2 Low maintenance requirements

The systems should be easy to maintain without lots of intervention by knowledge engineers. If the product range changes a lot then so will the KBS. We have also seen that product teething problems and major changes of circumstances may require changes to the KBS.

Rapid change in the knowledge is taken as a negative indicator for the likely success of KBS development (Laufmann et al., 1990). Our earlier analysis (Bridge & Dearden, 1992) clearly

[1]It may be that cases collected later specifically by and for the EBR system will contain more useful information than those initial cases, in which case the deficient old cases may be weeded out if desired.

identified such rapid change as a feature of many help desk domains, and hence argued that any KBS should have a low maintenance overhead.

It has been convincingly argued that rule-based systems, as one subtype of DMBR systems, are difficult to modify (Keravnou & Johnson, 1986). One reason for the difficulty of maintaining rule-based systems is the way that the knowledge they contain is distributed between a large number of rules. An ideal situation for maintenance of a KBS is:

> . . . if one real 'thing' were to change in the application then only one represented 'thing' would have to be changed in the implementation. (Debenham, 1989)

In a rule-based system if one thing in the application changes this may involve changes to many different rules, and the rules that must be changed cannot be identified easily.

In response to the problem, Debenham proposes a 'normalized model' that records the structure of a rule-based system in a way designed to simplify the identification of sets of rules which need changing in response to a change in the domain. Alternative knowledge representations such as frames (Minsky, 1985) or semantic networks (Brachman, 1985) may be used to group together related elements of a domain model, and thus to simplify the process of identifying which elements of a domain model must be updated. Structured architectures such as those proposed by Keravnou & Johnson (1986) may also assist in simplifying the maintenance task by separating different aspects of the knowledge. However, none of the above authors would suggest that maintaining a DMBR constructed according to their particular guidelines was a simple task.

Systems using a 'deep' model in which the elements of the domain model are more closely linked to the properties of domain objects may come closer to the ideal proposed above, but only at a considerable cost in terms of the effort that must be expended in constructing the initial domain model, and in modelling any new element introduced to the system.

In EBR approaches the main items that require change are the cases themselves. These may need updating, adding or deleting from the case base. It may be possible to automate these processes by using some form of time-stamping mechanism to allow old cases to wither away if they are not regularly used, and adding all successful new cases to the case base. However, we believe a more practical solution for most applications will require the intervention of some domain expertise. For the CLAVIER system, an EBR system for designing loading configurations for an autoclave (a large oven used for firing aircraft parts), regular meetings of the systems users were required to ensure that cases added to the system represented only 'successful' solutions (Hennessy & Hinkle, 1992). Cases were then added in a batch. For updating and deleting cases in a help desk EBR system it may be relatively simple for a domain expert, or even a HDO, to identify cases, or classes of cases, that will be affected by a change in the domain.

If the similarity measures used are automatically generated, then they will be relatively simple to update. If they are engineered, they may only require occasional updating when a large number of changes have occurred in the domain. At this point the intervention of a skilled knowledge engineer may be necessary.

A comparison of the engineering requirements for a case-based reasoning system, CASCADE, and a rule-based system, CANASTA, in the same domain is provided by Simoudis and Miller (1991). The authors compare the time spent maintaining consistency whilst integrating modules of the rule-based system with the time spent on revising previously stored cases and changing the similarity metrics as cases were added to the case-based system. Maintaining consistency on the CANASTA system took up 17.5% of the time in development whereas no alterations were required at all for the CASCADE system. Over the longer term, the authors do not claim that CASCADE will require *no* maintenance, however they clearly expect maintenance to be much less of a problem for the EBR system. In total, CASCADE required only 1/9th of the time and 1/8th of the cost required to develop the rule-based system, although it should be noted that CASCADE's knowledge covered only 60% of the domain of CANASTA, and the economies claimed may not scale directly if CASCADE were to be extended. However, the indications for the relative maintenance needs of EBR systems are encouraging.

On the specific problem of product teething troubles, which we related to maintenance in section 4.1.2, we suggest that it is not cost-effective to handle these teething problems using a DMBR system. A knowledge engineer would be needed to alter the domain model to reflect the presence of the teething problem and its temporary solution, and would then later need to remove this when the permanent fix is made. In particular, the identification of the exact parts of the domain model that need updating, so as to identify the problem whilst not introducing errors to the system, may take as much time and effort as precisely identifying and fixing the bug in the product.

EBR might offer a solution to product teething problems. It will be quite easy to add a prototype (or actual) case with a suitable solution to the case base. In many systems frequency statistics are stored alongside cases. For a short while, the frequency of use of a case representing a product teething problem will be high. Others 'phoning in with the same problem will therefore be quite likely to match against this. Once the permanent fix is made, the frequency statistics will decrease. Matches will be less likely. Eventually, that case can be removed either by an automatic garbage collection routine which deletes cases that fall below a threshold (not necessarily a good idea) or by manual intervention on the basis of a listing of infrequently matched cases, or it might just stay in the case base but match infrequently.

In dealing with cases associated with teething problems for a particular product, the product identity and some time stamping of cases could allow easy identification of cases that require editing when the teething problems are solved. This is exactly the right behaviour and comes for free with an EBR system: popularity grows and then withers (indeed, disappears).

4.3.3 Discussion

The previous two subsections have suggested that EBR systems may be easier to design and maintain than DMBR systems. However, we should note that, due to the relatively recent development of EBR techniques, little empirical support is available for this belief. What is required, in addition to more empirical evidence, is an analytic framework to enable designers to explore the trade-offs between different KBS tools and techniques and the emergent properties (such as ease of design and maintenance costs) of the resulting systems (Johnson & Ghalal, 1992). Some elements of such a framework can be seen in the work that has been done on the learnability of programming languages (see, for example, Soloway & Iyengar, 1986; Olson et al., 1987; Wiedenbeck, 1986; Sholtz & Wiedenbeck, 1992). However, this work also indicates how far we are from understanding the cognitive processes involved in designing and maintaining complex software.

Green (1989) presents a set of cognitive dimensions which may be used to relate cognitive requirements to syntactic and semantic features of notations. Green applies this analysis to the design of spreadsheet interfaces, text processing systems and programming languages. The approach has also been applied to design rationale notations (Shum, 1991). This analysis might also be applied to compare knowledge representation schemes. Three of Green's dimensions seem particularly relevant, and we discuss these below.

- *Hidden/Explicit Dependencies* Design and maintenance of systems will be easier if dependencies between one part of a system and another are made explicit. Green suggests that, to ease the cognitive load on designers and maintainers, cross-referencing and browsing tools should be supplied as a matter of course. Researchers in the engineering of DMBR systems have also made this recommendation (Baroff et al., 1988) and many modern DMBR development environments (e.g. Inder, 1989b) provide this form of support.

  Analysis of so-called first generation expert systems (e.g. Johnson & Keravnou, 1988; Keravnou & Washbrook, 1989) showed how dialogue strategy and question ordering were hidden implicitly within the ordering of clauses within rules and the ordering of rules in the rule base. This resulted in difficulty in predicting the effects of minor changes in the rules. The second generation of expert systems (e.g. Clancey, 1986; Keravnou & Johnson, 1986; Keravnou et al., 1989) sought to explicate some of these dependencies by separating knowledge about diagnostic

strategy, knowledge about dialogue management, and domain-specific knowledge. Clearly, the designers of DMBR systems have tried to address the problems of hidden dependencies.

In EBR systems dependencies may be at their most implicit. The ordering of cases in response to a query may depend on global as well as local properties of a case base (e.g. Ashley & Rissland, 1988a; Tyree et al., 1988). Even where the case ordering is determined only by local factors in the case base, the behaviour of the system is determined by a set of cases (those cases in that neighbourhood), not directly by individual cases. This makes it difficult to predict the precise effect on system behaviour of changes in the case base. However, whilst fine-grained specific changes may be difficult to make, introducing new knowledge where no specific, premeditated change in behaviour is desired is easy. One simply adds new cases.

- *Viscosity/Fluidity* Viscosity characterizes the *number* of dependencies in a system. The more dependencies there are (the more viscous a system is), the more difficult it is to change.

  Relating this to KBS follows the same line of discussion as the discussion above of hidden dependencies. Even where dependencies are made explicit, DMBR systems necessarily contain many dependencies (Debenham, 1989).

  Green points out that different parts of a system may have different levels of viscosity. In considering a spreadsheet, changes to an individual cell's contents are easy to effect, but changes to the overall structure of the spreadsheet are very difficult. The same should be true of EBR systems. Adding, deleting or modifying cases should be simple, but changes to the similarity metric and indexing terms might be difficult as such changes may fundamentally alter the reasoning of the system. With this in mind, it should be noted that the use of global case base data in similarity assessment introduces extra viscosity to editing the case base.
- *Role Expressiveness* This dimension characterizes the degree to which functionally different parts of a system may be recognized by static analysis of textual features. The separation of different types of knowledge in second generation DMBR systems clearly improves the role expressiveness of these systems. In EBR systems it can be argued that the precise role of indexing terms may be hard to discern, however the roles of the cases themselves could not be clearer.
- *Cognitive Requirements of KBS development* As research in DMBR systems design has progressed and matured it is apparent that tools and techniques have had to be developed to respond to some of the difficulties which we have highlighted using Green's analysis. In particular, architectures have evolved to make dependencies more explicit, and to present these dependencies graphically to the KBS developer.

  Because EBR systems are a more recent innovation, we may as yet be unaware of problems in their maintenance. As more systems are fielded we may expect to discover new problems, and require new tools to support EBR development. A better understanding of the cognitive requirements of software design and maintenance in general, and specifically of the difficulties in building and maintaining knowledge based systems, may help us in predicting some of the requirements for EBR development tools. Clearly, this is an important field for research.

5 Preliminary conclusions

From the analysis we have given above, we conclude that EBR systems are likely to prove more appropriate as KBS solutions for supporting front-line HDs than DMBR systems. Below we summarize the arguments in their favour.

5.1 Reasoning and representation

- EBR systems may offer a slightly better method of recognising the limits of their own scope.
- Product teething problems should be simpler to handle in an EBR system. We noted here that an appeal to 'deep' DMBR models does not address this particular problem.

- For dealing with major disruptions neither type of system is likely to perform well, but an EBR system working with an active human operator might offer some help in such situations.
- To speak the same language as the users, particular architectures have been proposed for DMBR systems. Similar architectures may be necessary for EBR systems to address this problem.
- To perform shallow but broad reasoning EBR systems are preferable. DMBR systems are usually associated with reasoning in a deep but narrow domain.
- The correct level of diagnosis should be possible in either type of system.
- Uncertainty will present similar problems for either type of system.
- Multiple hypotheses can only be supported if the notion of hypothesis is explicitly represented in the system, and can form a part of the user/system dialogue. EBR systems provide a simple notion of hypothesis (a previous case) which should be easily understood by a user.
- We are experimenting with designs for an EBR system which supports a form of 'what-if' reasoning. Current DMBR systems do not seem to be able to do this: the user has to 'pretend' that some fact is true and then retract that fact.
- DMBR is likely to provide slightly more informative traces of reasoning to second-line help desks.

5.2 Interaction

- Integration with other call management software is achievable by either type of system.
- Moving onto a new call in a DMBR system may involve more information being stored than in an EBR system.
- Mixed-initiative interaction can, in principle, be supported by either type of system. However, EBR systems may allow the HDO a more active role, and thereby improve the customer's perception of the HDO's competence.

5.3 Design and maintenance

It is in respect of design and maintenance requirements that EBR (potentially) offers the greatest benefits. DMBR systems are complex systems which require considerable effort to design and maintain (Breuker & Wielinga, 1987; Laufmann et al., 1990; Hickman et al., 1989; Debenham, 1989).

In contrast, major elements of the knowledge acquisition of EBR systems, particularly the collection of cases and some of the knowledge for similarity assessment, may be automated. Also, such knowledge acquisition as will have to be manually conducted, should be only a subset of the acquisition required for a DMBR system. Other authors (e.g. Simoudis & Miller, 1991; Hennessy & Hinkle, 1992) have indicated major savings in the amount of knowledge engineering required to produce EBR systems in comparison to DMBR systems.

5.4 Cautionary notes

The argument above shows that EBR systems should be investigated in preference to DMBR systems for *front-line* HD applications. However, this does not necessarily mean that DMBR solutions will not be useful in some circumstances.

We noted above that EBR systems may have varying amounts of domain knowledge either distributed with the cases or separately stored for use in similarity assessment. Thus there is a continuum of systems between the pure DMBR and pure EBR models that will need to be investigated.

We also related all our analysis to the problem of front-line HDs, and it is our opinion that DMBR systems may be more appropriate for the specialist work of second-line HDs.

Finally, it should also be noted that research into EBR systems is a much younger field than research into DMBR systems, and we should expect new problems with the design of EBR systems

to be identified as that research progresses. We are now investigating a class of EBR systems which we term Interactive Case Memories, as appropriate systems for use in front-line HDs.

Acknowledgements

This article is based on work conducted at the University of York in association with BT plc. We wish to express our thanks for their constructive comments to Professor Michael Harrison of the University of York, to Richard Oppenheimer and Rob Davis of BT plc, and to the referees of this paper. Andrew Dearden is sponsored by a CASE studentship from SERC and BT plc.

References

Aaronson, A and Carroll, JM, 1987, "The answer is in the question: a protocol study of intelligent help" *Behaviour and Information Technology* **6**(4) 393–402.

Abraham, DM, Spangler, WE and May, JH, 1991, "Expertech: Issues in the design and development of an intelligent help desk system" *Expert Systems with Applications* **2** 305–319.

Aha, DW, 1991, "Case-based learning algorithms" In: *DARPA Case Based Reasoning Workshop* pp 147–158. Morgan Kaufmann.

Aha, DW, Kibler D and Albert, MK, 1991, "Instance based learning algorithms" *Machine Learning* **6** 37–66.

Ashley, K and Rissland, E, 1988a. "Waiting on weightings: A symbolic least commitment approach" In: *AAAI 88*, pp 239–244. Morgan Kaufmann.

Ashley, KD and Rissland, EL, 1988b, "A case-based approach to modeling legal expertise" *IEEE Expert* **3**(3) 70–77.

Bain, WM, 1986, *Case-Based Reasoning: A Computer Model of Subjective Assessment* PhD thesis, Yale University.

Bareiss, R (ed.), 1991, *DARPA Case Based Reasoning Workshop 1991* Morgan Kaufmann.

Bareiss, ER, Porter, BW and Wier, CC, 1991, "Protos: an exemplar-based learning apprentice" In: Y Kodratoff and R Michalski (eds.), *Machine Learning: An Artificial Intelligence Approach Vol 3*, pp 112–127. Morgan Kaufmann.

Bareiss, R (chair), 1989, "Panel discussion of indexing vocabulary" In: *DARPA Case based reasoning workshop 1989*, pp 66–84. Morgan Kaufmann.

Baroff, J, Simon, R, Gilman, F and Shneiderman, B, 1988, "Direct manipulation user interfaces for expert systems" In: J Hendler (ed.), *Expert Systems: The User Interface*, pp 99–125. Ablex.

Barr, A, 1991, "Transforming support at the help desk" *Information Centre Quarterly* Spring 20–27.

Brachman, RJ, 1985, "On the epistemological status of semantic networks". In: RJ Brachman and HJ Levesque (eds.), *Readings in Knowledge Representation*, pp 191–216, Morgan Kaufmann.

Brachman, RJ and Levesque, HJ (eds.), 1985, *Readings in Knowledge Representation* Morgan Kaufmann.

Breuker, J and Wielinga, B, 1987, "Use of models in the interpretation of verbal data" In: AL Kidd (ed.), *Knowledge Acquisition for Expert Systems*. Plenum Press.

Bridge, DG and Dearden, AM, 1992, "Knowledge based systems support for help desk operations: A reference model" *Int. J. Systems Research and Information Science* **5** 217–234.

Carbonnel, JG, 1986, "Derivational analogy: A theory of reconstructive problem solving and expertise acquisition" In: RS Michalski, JG Carbonell and TM Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach, Volume II*, pp 371–392. Morgan Kaufmann.

Chandrasekeran, B, 1983, "Towards a taxonomy of problem solving types" *The AI Magazine* Winter/Spring 9–17.

Clancey, WJ, 1985, "Heuristic classification" *Artificial Intelligence* **27** 289–350.

Clancey, WJ, 1986, "From Guidon to Neomycin to Heracles in twenty short lessons: Orn final report 1975–1985" *The AI Magazine* August 40–60, 187.

Clancey, WJ and Letsinger, R, 1981, "Neomycin: Reconfiguring a rule based expert system for applications to teaching" In: *Proc. IJCAI 81*, pp 829–836.

Coombs, MJ and Alty, JL, 1980, "Face to face guidance of university computer users 2. Characterising advisory interactions" *Int J. Man Machine Studies* **12** 389–405.

Dearden, AM, 1993, "Interacting with a case memory" In: *Proc IEE Colloquium on Case-Based Reasoning*. IEE, Computing and Control Division, Professional Group C4 (Artificial Intelligence).

Debenham, J, 1989, "The normalized model and expert systems maintenance" In: N Shadbolt (ed.), *Research and Development in Expert Systems VI*, pp 78–88. Cambridge University Press.

DeKleer, J, 1986, "An assumption-based TMS" *Artificial Intelligence* **28** 127–162.

Duda, R, Gasching, J and Hart, P, 1979, "Model design in the PROSPECTOR consultant system for mineral exploration" In: D Michie (ed.), *Expert Systems in the Microelectronic Age*, pp 153–167. Edinburgh University Press.

Fischer, G, 1990, "Communications requirements for co-operative problem solving systems" *Information Systems* **15**(1) 21–36.

Fox, J, Clarke, DA, Glowinski, AJ and O'Neil, MJ, 1990, "Using predicate logic to integrate qualitative reasoning and classical decision theory" *IEEE Trans. on Systems, Man and Cybernetics* **20**(2) 347–357.

Green, TRG, 1989, "Cognitive Dimensions of notations" In: Sutcliffe, A and Macaulay, L (eds.) People and Computers V, Proceedings of HCI 89, pp 443–460, Cambridge University Press.

Greiner, R, 1988, "Learning by understanding analogies" In: A Prieditis (ed.), *Analogica*, pp 1–36. Pitman.

Hammond, K, 1989a, *Case Based Planning* Academic Press.

Hammond, K (ed.), 1989b, *DARPA Case Based Reasoning Workshop 1989* Morgan Kaufmann.

Hennessy, D and Hinkle, D, 1992, "Applying case based reasoning to autoclave loading" *IEEE Expert* October 21–26.

Hickman, FR, Killin, JL, Land, L, Mulhall, T, Porter, D and Taylor, RM, 1989, *Analysis for Knowledge Based Systems a practical guide to the KADS methodology*. Ellis Horwood.

Hinrichs, TR and Kolodner, JL, 1991, "The roles of adaptation in case based design" In: *AAAI 91*, pp 28–33. MIT Press.

Inder, R, 1989a, "Experience of constructing a fault localisation expert system using an AI toolkit" In: S Vadera (ed.), *Expert System Applications*, pp 151–167. Sigma Press.

Inder, R, 1989b, "The state of the art" In: S Vadera (ed.), *Expert Systems Applications*, pp 151–167. Sigma Press.

Johnson, L, 1985, "The MDX diagnostic system" *Int. J. Systems Research and Info. Science* **1** 163–171.

Johnson, L and Ghalal, GH, 1992, "Methodological issues in knowledge based systems development" In: *IEE Colloquium on Software Engineering and AI*.

Johnson, L and Keravnou, ET, 1988, *Expert Systems Architectures*. Kogan Page, 2nd edition.

Jones, EK, 1989. "Case-based analogical reasoning using proverbs" In: K Hammond (ed.), *Proceedings: Case Based Reasoning Workshop*, pp 275–279. Morgan Kaufmann.

Kedar-Cabelli, ST, 1988, *Formulating concepts and Analogies According to Purpose*. PhD thesis, Rutgers University.

Keravnou, ET and Johnson, L, 1986, *Competent Expert Systems*. Kogan Page.

Keravnou, ET and Washbrook, J, 1989, "What is a deep expert system: An analysis of the architectural requirements of second-generation expert systems". *Knowledge Engineering Review* **4**(3) 205–233.

Keravnou, ET, Washbrook, J, Dawood, RM, Hall, CM and Shaw, D, 1989, "A model based diagnostic expert system for skeletal dysplasias". In: J Hunter, J Cookson and J Wyatt (eds.), *AIME 89: Proceedings of the second European Conference on Artificial Intelligence in Medicine*, pp 47–56.

Kibler, D and Aha, DW, 1988, "Case based classification" In: *AAAI 88 Case Based Reasoning Workshop*, pp 62–67.

Kidd, AL, 1985, "What do users ask? Some thoughts on diagnostic advice" In: *Proceedings of the 5th Technical Conference of the BCS Special Interest Group on Expert Systems*, pp 9–19.

Kilhoffer, AR and Wisely, CL, 1990, "HELDA: The help desk assistant". *AI Expert* February 57–59.

Kolodner, JL (ed.), 1988. *DARPA Case Based Reasoning Workshop 1988*. Morgan Kaufmann.

Land, L and Mulhall, T, 1989, "Developing co-operative knowledge based systems" In: N Shadbolt (ed.), *Research and Development in Expert Systems VI*, pp 90–103. Cambridge University Press.

Laufmann, SC, DeVaney, DM and Whiting, MA, 1990, A methodology for evaluating potential KBS applications. *IEEE Expert* December 43–62.

Maybury, MT, 1990, *Planning Multisentential English Text Using Communicative Acts*. Technical Report RADC-TR-90-411, Rome Air Development Center, Griffiss Air Force Base, NY.

Minsky, M, 1985, "A framework for representing knowledge" In: RJ Brachman and HJ Levesque (eds.), *Readings in Knowledge Representation*, pp 245–263. Morgan Kaufmann.

Motro, A, 1987, "Extending the relational database model to support goal queries" In: L Kerschberg (ed.), *Expert Database Systems*, pp 129–150. Benjamin/Cummings.

Muns, RJ, 1990, "The expert help desk .. 28 factors for success" In: *Conference on Artificial Intelligence and the Help Desk*. The Help Desk Institute.

Neches, R, Swartout, WR and Moore, JD, 1985, "Enhanced maintenance and explanation of expert systems through explicit models of their development". *IEEE Transactions of Software Engineering* **11**.

Olson, G, Sheppard, S and Soloway, E (eds.), 1987, *Empirical Studies of Programmers*. Ablex.

Patil, RS, Szolovits, P and Schwarz, WB, 1981, "Causal understanding of patient illness in medical diagnosis" In: *Procs IJCAI 81*, pp 893–899.

Pearce, M, Goel, AK, Kolodner, JL, Zimring, C, Sentosa, L and Billington, R, 1992, "Case based design support: A case study in architectural design", IEEE Expert, October, pp. 14–20.

Pollack, ME, 1985, "Information sought and information provided: An empirical study of user/expert dialogues" In: *Proceedings of the ACM/CHI San Fransisco*, pp 155–159.

Pople Jr, H, 1985, "Evolution of an expert system: from Internist to Caduceus" In: I DeLotto and M Stefanelli (eds.), *Artificial Intelligence in Medicine*. Elsevier.

Prieditis, A (ed.), 1988, *Analogica*. Pitman.

Rajamoney, SA and Lee, H-Y, 1991. "Prototype based reasoning: An integrated approach to solving large novel problems" In: *AAAI 91 Transformation in Design Workshop*, pp 34–39. AAAI/MIT Press.

Register, MS and Rewari, A, 1991, "CANASTA: The crash analysis troubleshooting assistant" In: RG Smith and AC Scott (eds.), *Innovative Applications of Artificial Intelligence 3*, pp 195–212.

Seifert, C (chair), 1989, "Panel on analogy and CBR" In: K Hammond (ed.), *Proceedings: Case Based Reasoning Workshop 1989*, pp 125–159. Morgan Kaufmann.

Shafer, D, 1991, "CBR Express: Getting down to cases". *PC AI* July/August 42–45.

Sholtz, J and Wiedenbeck, S, 1992. "The use of unfamiliar programming languages by experienced programmers" In: A Monk, D Diaper and MD Harrison (eds.), *People and Computers VII*, pp 45–56. Cambridge University Press.

Shortliffe, EH, 1976, *Computer-based Medical Consultations*. Elsevier.

Shum, S, 1991. "Cognitive dimensions of design rationale" In: D Diaper and N Hammond (eds.), *People and Computers VI*, pp 331–344. Cambridge University Press.

Simoudis, E and Miller, JS, 1991, "The application of CBR to help desk applications" In: *DARPA Case Based Reasoning Workshop 1991*. Morgan Kaufmann.

Soloway, E and Iyengar, S (eds.), 1986, *Empirical Studies of Programmers*. Ablex.

Steels, L, 1986, "Second-generation expert systems" In: M Bramer (ed.), *Research and Development in Expert Systems III*. Cambridge University Press.

Stenton, SP, 1987, "Dialogue management for co-operative knowledge based systems". *Knowledge Engineering Review* **2**(2) 99–122.

Swaminathan, K, 1988. "Properties of an indexing scheme" In: *AAAI Case Based Reasoning Workshop*, pp 132–136.

Swartout, WR and Smoliar, SW, 1987, "On making expert systems more like experts". *Expert Systems* **4**(3).

Taylor, JM, 1985. "An expert system for terminal fault diagnosis" In: *First International Expert Systems Conference*, pp 193–199.

Thompson, K and Langley, P, 1989, "Organisation and retrieval of composite concepts" In: JL Kolodner (ed.), *DARPA Case Based Reasoning Workshop 1989*, pp 329–333. Morgan Kaufmann.

Travé-Massueyès, L et al., 1992, "Qualitative reasoning". *Special Issue of Knowledge Engineering Review* **7**(1).

Turner, RM, 1989, "Case-based and Schema-based reasoning for problem solving" In: *Proceedings of the Case-Based Reasoning Workshop*, pp 341–344. Morgan Kaufmann.

Tyree, AL, Greenleaf, G and Mowbray, A, 1988, "Legal reasoning: The problem of precedent" In: JS Gero and R Stanton (eds.), *Artificial Intelligence Developments and Applications*, pp 231–245. Elsevier.

Ward, RD and Sleerman, D, 1987, "Learning to use the S.1 knowledge engineering tool" *Knowledge Engineering Review* **2** 265–276.

Weiss, SM, Kulikowski, CA, Amarel, S and Safir, A, 1978, "A model-based method for computer-aided medical decision-making". *Artificial Intelligence* **11** 145–172.

Wiedenbeck, S, 1986. "Processes in computer program comprehension" In: E Soloway and S Iyengar (eds.), *Empirical Studies of Programmers*, pp 58–79. Ablex.

Young, RM, 1989, "Human interface aspects of expert systems" In: LA Murray and JTE Richardson (eds.), *Intelligent Systems in a Human Context*, pp 20–34. Oxford University Press.

Zadeh, LA, 1965, "The role of fuzzy logic in the management of uncertainty in expert systems". *Fuzzy Sets and Systems* **11** 199–227.