

# A Case-Based Answer to Some Problems of Knowledge-Based Systems

Agnar Aamodt

*University of Trondheim, Department of Informatics*  
*N-7055 Dragvoll, Norway*  
agnar.aamodt@ifi.unit.no  
+47-7-591838

**Abstract.** Among the most important challenges for contemporary AI research are the development of methods for improved robustness, adaptability, and overall interactiveness of systems. Interactiveness, the ability to perform and react in tight co-operation with a user and/or other parts of the environment, can be viewed as subsuming the other two. There are various approaches to addressing these problems, spanning from minor improvements of existing methods and theories, through new and different methodologies, up to completely different paradigms. As an example of the latter, the very foundation of knowledge-based systems, based on a designer's explicit representation of real world knowledge in computer software structures, has recently been questioned by prominent members of the KBS community. In the present paper, some foundational issues of the knowledge-based paradigm are reviewed, and the main arguments of the critiquing position are discussed. Some of the deficiencies of current approaches pointed out by the critics are acknowledged, but it is argued that the deficiencies cannot be solved by escaping from the knowledge-based paradigm. However, an alternative to the mainstream, generalisation-based, approach is needed. An approach is advocated where reasoning from situation-specific knowledge - captured as a collection of previously solved cases, is combined with generalised domain knowledge in the form of a densely connected semantic network. The approach enables continuous, sustained learning by updating the case base after each new problem has been solved. The paper shows how an example of this approach - the CREEK system - can provide an answer within the knowledge-based paradigm to the problems pointed out by the critics.

## 1. Introduction

A goal of AI is to reach the necessary understanding of intelligence and intelligent behaviour to be able to develop what may be referred to as intelligent artefacts. The most successful paradigm for research and development has so far been the *knowledge-based systems* paradigm - or just knowledge-based paradigm. This paradigm evolved from earlier knowledge-independent approaches, and is characterised by the strong role of explicit, domain-dependent knowledge in problem solving. The basis for explicit knowledge modelling rests upon a few central assumptions, such as the Knowledge Representation Hypothesis [Smith-82] and the Physical Symbol Systems Hypothesis [Newell-80]. This foundation of AI systems, and its operationalization into methods aimed to explicitly represent real world knowledge as computer software structures, has recently been questioned by prominent members of the KBS community [Clancey-89, Winograd-90]. It is

claimed that for an AI system to exhibit the interactions with its environment which are

necessary for real world problem solving and learning - i.e. to become more 'situated' – its knowledge can not be pre-defined and explicitly represented by a designer. This view has lead to new suggestions for building robots as well as user-interactive decision support systems, e.g. expert systems. Various approaches to the development and realisation of intelligent behaviour in computer systems have been proposed as alternatives to the knowledge-based paradigm [Winograd-86, Churchland-86, Clancey-89, Brooks-91a, Rademakers-92]. Although there are substantial differences between these approaches, they tend to agree that intelligent behaviour of artificial systems should emerge from a set of basic mechanisms and repeated interactions with the environment. This view has lead to new suggestions for building robots as well as user-interactive decision support systems.

Although the critiquing position has pointed out several deficiencies of current knowledge-based approaches, it fails to provide a convincing argument for the need to replace the knowledge-based paradigm with something else. A major weakness in the argumentation is that it is mainly based on the deficiencies of well-established, and hence somewhat 'old', methods and systems. Typically, the 'MYCIN school' of systems (i.e. simple knowledge representation schemes, rule-based and shallow reasoning) are used to exemplify weaknesses of the knowledge-based approach - and even of present practice within this paradigm. The potential of recent research results - addressing more competent, robust, and adaptive problem solving behaviour - has not been sufficiently taken into account. Examples of promising recent results include work on knowledge-level modelling methodologies [Steels92, Wielinga-92], case-based methods for problem solving and adaptive learning [Aamodt-90, Kolodner-92], representation languages for weak-theory domains [Guha-90, Aakvik-91], and integrated architectures for learning and problem solving [Ram-92, Plaza-93].

This paper briefly reviews the major points in the critique against the knowledge-based paradigm from the perspective of highly interactive ('interactional') systems - systems which interact heavily with the user and other parts of its environment during problem solving and adaptive learning. It is suggested that deficiencies of current systems should be addressed by extending our methodological basis *within* the knowledge-based paradigm, instead of abandoning it. The paper starts with a summary of the foundation of the knowledge-based paradigm. This is followed by a list of the main critiques against this platform, related to the interactiveness<sup>1</sup> of systems. A suggestion is then given of a knowledge-based approach to the problems attacked by the critics. A case-based method is advocated, where the reasoning from case-specific knowledge is combined with reasoning within a body of densely coupled general domain knowledge. This enables robust problem solving, adaptive learning, and the overall openness required for interactive problem solving. New experiences are retained in a way coherent with the general knowledge, and reused when solving future problems. The final part of the paper describes an instantiation of this approach, based on the CREEK architecture [Aamodt-91a] for knowledge-intensive case-based problem solving and learning.

## 2. The Knowledge-Based Paradigm

### 2.1 *The Knowledge Representation Hypothesis*

A fundamental assumption of the knowledge-based paradigm is that knowledge - as we know it and daily use it - can be identified and explicitly represented. This is captured by the Knowledge Representation Hypothesis, e.g. as expressed in [Smith-82]:

---

<sup>1</sup>The term 'interactiveness' as used here captures the continuous and active impact from the environment upon an agent, and vice versa. Another term capturing essentially the same thing is 'situatedness'.

*Any mechanically embodied intelligent process will be comprised of structural ingredients that*

*a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and*

*b) independent of such external semantical contribution, play a formal but causal and essential role in engendering the behaviour that manifests that knowledge.*

The core points are that within intelligent systems there exist 'structural ingredients' which we as observers take to represent knowledge, and that this knowledge not only exists, but also plays a crucial role in producing intelligent behaviour.

A related hypothesis, the Physical Symbol Systems Hypothesis [Newell-80], takes this view a step further by establishing the notion of *physical symbols* - symbols realised by physical mechanisms - as a necessary and sufficient means to generate intelligent behaviour. Based upon these two hypotheses the 'Knowledge Principle' emphasises the strong role of *specific domain knowledge* - in relation to more general methods for reasoning and problem solving. As expressed in [Lenat-87]: "A system exhibits intelligent understanding and action at a high level of competence primarily because of the specific knowledge that it can bring to bear: the concepts, facts, representations, methods, models, metaphors, and heuristics about its domain of endeavour. "

The Knowledge Representation Hypothesis, extended with the notion of physical symbol systems and the importance of application specific domain knowledge, builds the foundation of the knowledge-based paradigm in AI<sup>1</sup>.

Commonly used words like, 'knowledge-engineering', 'knowledge-level modelling', 'knowledge bases', etc., clearly give a central role to the term 'knowledge'. However, the widespread and frequent use of this term in AI and computer science in general seems to take for granted that we all know what it means. Knowledge is a difficult term to define, and it is probably no exaggeration to say that a lot of confusion and misinterpretation of arguments can be traced back to different views on what knowledge is. In the following, an attempt to clarify this issue is made, by focusing on two essential questions. The first is related to the *frame of reference* of knowledge. That is, when we use the term knowledge, whose knowledge are we referring to - who owns the knowledge? This question is a crucial one, and a major issue underlying the critique of the knowledge-based paradigm [Clancey-89]. The second is related to the use of knowledge as a term; for example why is it called knowledge, and not information or data. The perspective taken here is that of the *role* of knowledge - vs. information and data - in intelligent decision making.

## 2.2 *Knowledge - its Frame of Reference*

The term 'knowledge-based system' refers to a system that is *based on* knowledge. In order for the term to have some meaning distinct from 'information system', such a system is usually assumed to be based on knowledge in the sense that it possesses knowledge - owns knowledge and ways to use this knowledge. A knowledge level description [Newell-82] of a system - human or other - ascribes to the system the property of having knowledge - i.e. its own knowledge - and means to utilise this knowledge to achieve its goals. So, in conformance with the knowledge-based paradigm, a computer system contains 'structural ingredients' - i.e. symbol structures - associated with interpreters which capture the semantical contents of the symbols. The symbols, the underlying interpreters, and the actual implementation which 'grounds' the symbolic representation to phenomena of the real world, is jointly referred to as the system's knowledge.

A typical picture of how it is possible for a computer system to acquire knowledge so

---

<sup>1</sup>This does not mean that all three hypotheses are fully subscribed to by all researchers within the KBS community.

that it becomes its own, is as follows: At the outset, the system has no knowledge, knowledge exists only within the human designer. On the basis of this knowledge, initial structures are being built up within the system, structures which are to become the system's own knowledge. This is an iterative process of knowledge modelling, development of representations and inference methods, and testing within the system's target environment. Modifications and extensions to the system's knowledge is made, by manual or automated acquisition methods, on the basis of how well the system behaves within its environment. This process continues throughout the life time of the system. The goal of knowledge engineering methods, then, are to acquire and represent data structures and interpretation methods which the system will be able to use as *its* knowledge in *its* tasks of problem solving and interactions with the environment.

The important point is that when we ascribe knowledge to a system, we give it the property of possessing knowledge, and the frame of reference for that knowledge should therefore be the system itself. It may be argued, and rightfully so, that our knowledge acquisition and representation methods are too weak to enable a realisation within a computer of the knowledge we want the system to have. The system's knowledge, i.e. the system's interpreted meaning of its symbol structures, may therefore be different from the meaning that the same structures have for a human being. It is a primary goal of AI research to develop methods which enable the shrinking of this gap. Anyhow, these problems are related to the *contents* of the knowledge, not its frame of reference.

### 2.3 Knowledge - its Role in Decision Making

From the frame of reference discussion above, knowledge is always subjective, and only exists within a particular system through this system's reasoning and interpretation methods. This means that, if we want to be precise in our use of terms, it does not make sense to talk about 'knowledge within a book', unless the book has reasoning capabilities. *Knowledge* is always *within* a reasoning agent - human or artificial - and nowhere else. When we, correspondingly, view *information* as interpreted data, it only makes sense to talk about *data* in a book. The information itself has to come from an interpretation process who uses its knowledge in order to understand and thereby 'transform' data into information<sup>1</sup>.

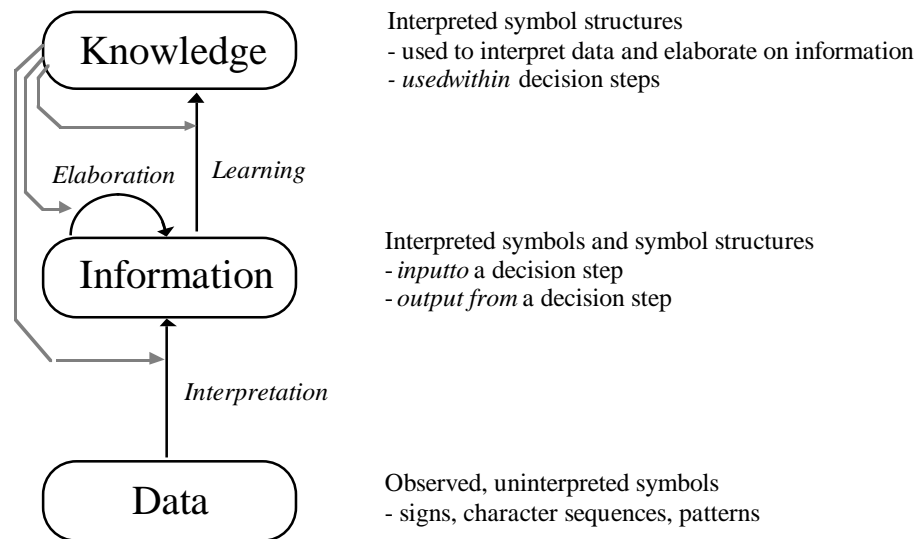
The role of knowledge is an *active* one, i.e. to act in the processes of transforming data into information (interpretation), deriving other information (elaboration, e.g. understanding or active problem solving), and acquiring new knowledge (learning). A way to illustrate these dependencies is shown in figure 1<sup>2</sup>. The context of the figure is a decision making process in general, irrespective of the type of decision to make, and the type of reasoning system (human or artificial) that makes a decision. A distinction between data and information, as already indicated, is that data are uninterpreted characters, signals, patterns, etc., i.e. they have no meaning for the system concerned. Data may become information after having been interpreted to give meaning for the system. In order to interpret data into information, the system needs knowledge. For example, "av)45\#TQ@0@", or a series of signals from a sensor, is data to most human beings, while "interest rate", "increased body temperature", and "monkey business" have meaning, and is therefore information. Their meaning may be different for different people - or other intelligent agents, and it is our knowledge about particular domains, and the world in general, that enables us to get

---

<sup>1</sup>Hence, when we in conversation with other human beings sometimes refer to "the information in a book", or even "the knowledge in a book", we implicitly assume that the interpreter is ourself or another human being with a similar cultural (and therefore interpretative) background.

<sup>2</sup>Although simplified and abstracted, the model presented is intended to capture the essential distinction between knowledge, information, and data. Even if it is rarely made explicit, I think this view reflects how most people in knowledge-based systems research actually use these terms. At least, it provides a framework for clarifications of opinions.

meaning out of the data strings. Information always assumes an interpreter.



**Figure 1: Data, Information, Knowledge.**

The role of knowledge in an intelligent system's decision process. The stippled lines show essential roles of knowledge in data interpretation, information derivation and elaboration, and learning of knowledge. The figure also illustrates the distinction between knowledge, information, and data.

In order to get meaning out of data, knowledge is needed. Knowledge therefore has - at least to some degree - to exist in the system when the interpretation process starts. The important distinction between information and knowledge is, therefore, that information is something which is taken as input to a decision process, and something which is produced as output from it, while knowledge is something which partly is pre-existing and anyhow is brought to bear within the decision process itself.

A second role of knowledge - in addition to data interpretation - is in deriving new information from other information. This process is here referred to as elaboration<sup>1</sup>, and includes the deriving of consequences and expectations, forming of hypotheses, generation of explanations, inferring of new values, etc. For example a medical decision support system, given the information "temperature has increased from 37 to 41.5 degrees during the last hour", may use its knowledge to infer "strongly and rapidly increased temperature", from which "possible life-threatening condition" may be concluded, in turn leading to the proposed action "check for life threatening condition".

A system's knowledge grows by learning. A learning process can also be viewed as a problem solving process, where the problem is what and how to learn. A system may learn by being explicitly told - e.g. by 'traditional' knowledge acquisition methods, or by being implicitly told, and from that infer its own modified knowledge structure. A knowledge-based learning method would in this case use the entered descriptions as data, try to interpret it into information (for the learning process), elaborate on it by checking for conflicts, deriving expectations, attempting to confirm expectations, etc., eventually leading to learning of new or modified knowledge. An increasing amount of research in machine learning as well as knowledge acquisition now focuses on learning by active integration of new information [Murray-88, Eggen-90].

<sup>1</sup>Elaboration is what typically is meant by the term reasoning, but the scope of reasoning in this discussion is broader, and cover parts of the interpretation and learning processes as well.

In a knowledge-intensive decision process (problem solving process), the notion of integration captures an essential aspect of both interpretation, elaboration, and learning: Data gets interpreted by being integrated into the existing information structure, which in turn is integrated into the knowledge structure. Elaboration adds new information by integrating it into existing information, and learning adds new knowledge by integrating it into the existing knowledge structures. Integration ensures coherence (if not always consistence) between new and existing information and knowledge.

### 3. Recent Critique of the Knowledge-Based Paradigm

#### 3.2 *The Main Arguments*

Over the last years, basic assumptions behind the knowledge-based paradigm - as described in the previous sections - have been subject to increased questioning. Critique of the fundamental hypotheses of the knowledge-based paradigm is not a new phenomena, of course (e.g. [Dreyfus-86, Searle-80]). The new thing is that it now comes from people who used to be - and to some extent still are - part of the knowledge-based systems community (e.g. [Winograd-86, Clancey-89, Rademakers-92]). In particular, Winograd and Flores' book [Winograd-86] seems to have been very influential. Bill Clancey - one of the foremost contributors to the KBS field - has also turned into a somewhat critiquing position, mainly based on Winograd's book. Another type of critique comes from the robotics field, as expressed in, e.g., [Brooks-91a, Brooks-91b]. This is the most extreme position, but since its alternative approach - emerging of intelligence in a purely bottom-up manner - only have been applied to autonomous robots, it is less relevant in the present context of interactive decision support systems. A third type of critique comes from the neural network area (e.g. [Churchland-90]). This is the less extreme of the critiquing positions, and may be viewed as closer to pushing the borders of the knowledge-based paradigm, than tearing it all down. The following discussion is mainly related to the first position, although some points and arguments to some extent also will apply to the other two.

Although there are different opinions among the critiquing positions, their major arguments are captured by the following three points:

- *Symbol structures* can not give meaning to a computer system, a computer can only manipulate syntax. The only way a computer system can develop intelligent behaviour is through repeated interactions with its environment. Intelligent behaviour must emerge, it can not be pre-specified and captured in explicit symbol structures.
- *The frame of reference* of knowledge is always a human being. Knowledge can not be from the point of view of a computer, it will always be with respect to an external designer or observer. That is, knowledge is a property which the observer ascribes to an agent. We may still talk about the knowledge of an artificial system, but then this is only a way to explain the agent's behaviour by using a metaphorical concept.  
According to this view, artificial intelligent agents can not possess - own - knowledge. A knowledge level model is always a descriptive model, not a prescriptive one.
- *The role of knowledge* in decision making is still to enable intelligent problem solving based on learned experience. But this knowledge is not something which can be 'separated out' and distinguished from information. Knowledge used in a situation does not reside within the agent, ready to be retrieved and used. Knowledge is constructed in the act of problem solving, within and from the situation in which it is used.  
This implies that a knowledge level model always will be a description of a system's interaction with the world, not of the system alone.

### 2.3 Discussion

The frame of reference of knowledge, and the role of knowledge level models, are important points of disagreement between the knowledge-based and the critiquing positions. The view that "knowledge can not be with respect to a computer", "a computer manipulates symbol structures which only can have meaning to a human observer", etc., is based on a completely different philosophical and psychological tradition than the current knowledge-based view. Claiming that knowledge can not be possessed by, i.e. represented and interpreted within, an artificial agent - and through this lead to meaning and understanding within the agent, takes a 'human chauvinistic' stance to knowledge and intelligence that almost *per definition* excludes the use of this concepts for other type of systems. Given the current state-of-the-art in AI and other sciences studying the phenomenon of intelligence, this seems to be a position with very weak scientific grounding.

There seems to be a general agreement that the role of knowledge is to enable data interpretation, information elaboration, and learning within a system. The question is partly what the nature of this knowledge is, i.e. what its properties are, and what the 'system' is. While the knowledge-based paradigm talks about knowledge and intelligence within a computer system, the critics claim that the only computer-related system for which these concepts make sense, is the total system of the computer and its operating environment (including human beings). The approach of pulling the environment more strongly into the analysis and design of the artefact is a sound one. Although current knowledge-engineering practise to an increasing degree is focusing on this aspect, both for problem solving and learning behaviour (e.g. [VandeVelde-92, Aamodt-92, Steels-92]), we would be wise to look more deeply into methods coming from the critiquing community (e.g. as proposed in [Rademakers-92]). However, although it is helpful to develop knowledge level descriptions of the *behaviour of total systems* - consisting of user, computing system, interaction media, and other relevant parts of the real world - it is useful to build prescriptive models at the knowledge level of the *computer system, as well*. The relations between this local model and the model of the total interacting environment gives important guidelines for an improved design methodology.

In summary, the critiquing position has contributed to the current state of knowledge-based systems research by highlighting some important areas of further research. The following three problems of current knowledge-based systems are of utmost importance to solve, in order for knowledge-based paradigm to continue as the leading AI paradigm: First, systems need to become more *robust* and more *competent* within their operating environment. What we need are decision support systems that are parts of their problem solving situation, not external to them. Second, the purpose of a knowledge-based decision support system is not to automate as many tasks as possible. Whatever intelligent behaviour the system may be able to achieve, what counts is the *total usefulness* of the system within its real world environment. That is, how the user and the system together, interactively, are able to improve the overall decision process. Finally, systems need to be able to *adapt*, gradually, to their problem solving environment. After each problem solving session, the system must learn from the experience. A minimal criterion is that a mistake once made will not be repeated.

## 4. Extending the Current Knowledge-Based Approach

The critique of the established knowledge engineering practise raises several important issues, and needs to be seriously examined. However, although major weaknesses of some

existing methodologies have been pointed out, no alternative paradigm has been proposed to date which is ready to completely replace the knowledge-based paradigm<sup>1</sup>. For decision support systems, what is needed is an alternative to - or an extension of - the main stream, top down approach to systems development. In particular, more emphasis must be given to methods for knowledge modelling within open domains, and for continuous adaptation through learning from experience. This should lead to a stronger focus on methods to develop more *knowledge-dense* domain models. For example, our knowledge models should not contain *either* causal, functional, structural, or associative relationships, but *all* of them. Further, we should not let systems start to solve problems 'from scratch' each time. We should take advantage of the powerful storage and retrieval properties of computers, and let them store and reuse previous concrete experience - past cases - as well. Knowledge-intensive methods are methods that are able to represent and utilise *combined* knowledge, i.e. the power of a knowledge model for real world problem solving lies just as much in the interrelationships between different knowledge types, as within single model types alone.

Hence, the problem with knowledge-based methods in AI is not that they rely on explicitly encoded knowledge. On the contrary, *the problem is that they have to rely on too little knowledge*, leading to too scarce knowledge models<sup>2</sup>. This renders them less competent and less robust than desired, and - equally important - results in too weak a basis for learning.

The learning issue is crucial: We can no longer afford to wipe the learning issue under the carpet - learning methods has to be an inherent part of any design method for AI systems. Since the learning of generalisations have shown to be a difficult and slowly advancing area [Shavlik-90], we should look more strongly into the learning of specialised knowledge - learning and re-use of situation-specific cases instead of generalised rules. Learning, in the form of a sustained adaptation to the ever-changing environment, is an essential quality of intelligent behaviour that very few AI systems so far have demonstrated. Although a lot of continued work needs to be done to achieve these properties, the current state-of-the-art in knowledge-acquisition [Wielinga-92, Weintraub-92], knowledge-based integrated systems architectures [VanLehn-91, Ram-92, Althoff-92, Plaza-93] and knowledge-intensive methods for problem solving and experiential learning [VandeVelde-88, Aamodt-90], clearly indicate that such a goal is feasible. Unfortunately, a significant amount of the critique that has been raised against the knowledge-based paradigm, has been based on an assessment of well established methods and systems, instead of the results and potentials of recent research.

#### 4.1 *Knowledge-Intensive Case-Based Reasoning and Learning*

The Creek system [Aamodt-91a] addresses the above problem by providing a coherent framework and a system design for knowledge-intensive, case-based problem solving and learning. The underlying architecture emphasises robustness by incorporating multiple knowledge types and reasoning methods, and through its ability to continually learn from experience. This section summarises the architecture, as an example of an approach to situation-specific, user-co-operative, open-world decision support.

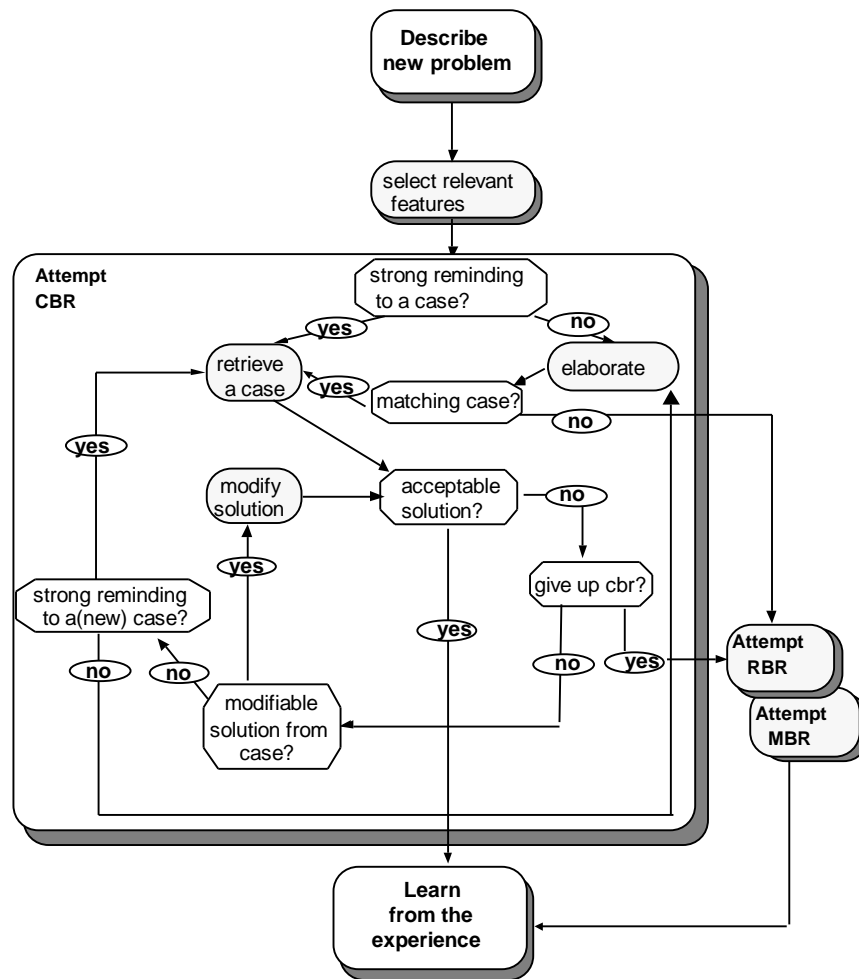
The main method of CREEK is case-based [Riesbeck-89, Aamodt-91b, Kolodner-92], but its case-based method relies heavily on an extensive body of general domain knowledge in its problem understanding, similarity assessment, case adaptation, and learning. In

---

<sup>1</sup>This is the case for user-interactive decision-support systems, at least. For robotics, where other paradigms have been worked out to a larger depth, the situation may be somewhat different.

<sup>2</sup>There may even be a kind of threshold effect here, in that a certain amount of knowledge is needed for a system to show any kind of intelligent behaviour. The fact that it is possible to achieve convergence in a large and growing knowledge base has been demonstrated in the CYC project [Lenat-90].

addition, if reasoning from case-specific knowledge fails - for example when no similar situation is found in the case base - general knowledge may be used in an attempt to solve the problem. This knowledge is typically build up by rather 'deep' relationships - for example a combination of causal, structural, and functional relations (depending on what is most appropriate within the application domain). It may also contain more shallow associations, in the form of If-Then rules. Figure 2 shows the top level combined reasoning algorithm.



**Figure 2: Combined Reasoning in Creek**

The three reasoning methods in Creek are illustrated by the three "Attempt XXX" boxes, of which the Case-Based Reasoning box has been expanded. (RBR = Rule-Based Reasoning, MBR = Model-Based Reasoning).

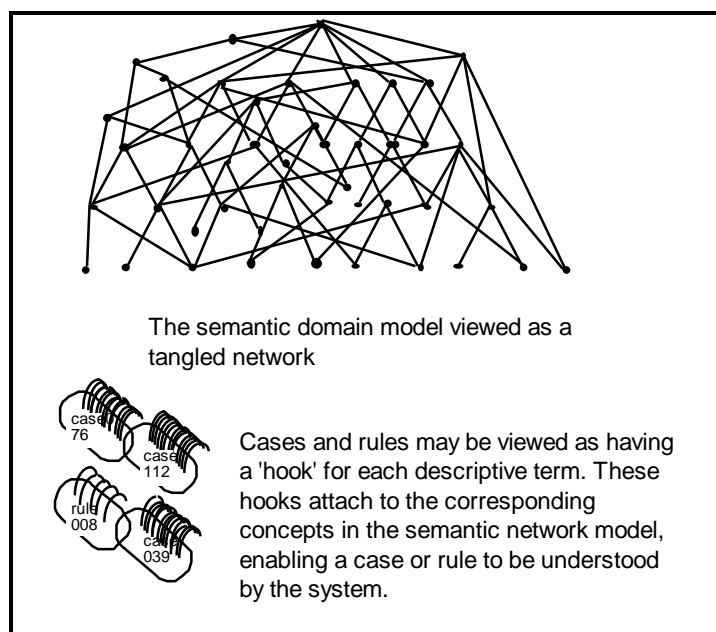
The process of selecting the initial reasoning paradigm starts after relevant problem features has been extracted from the problem description, and halts when either a reasoning method has been chosen, or a plausible solution to the problem has been found. Referring back to the discussion in chapter 2.3, the problem description is entered as data to the system, which interpretes the data into information by use of its knowledge and reasoning methods. Extraction of relevant problem features, and the subsequent case-based and generalisation-based processes up to the learning process, will be combinations of elaborations and interpretations.

If the problem features give a reminding to a previous case that is above a certain

threshold, Creek's case based problem solving method is triggered. The value of this threshold level depends on the relative contents and power of the case base. It is set manually to begin with, and adjusted over time according to how well the chosen reasoning paradigm performs. If this method fails for some reason, the rule based method is tried. As a last resort, problem solving is attempted by a model-based, 'deep-level', reasoning. Typically, the problem is solved by the case-based method, while the other reasoning methods serve to support the case-based processes.

At the meta-level, there are explicit control models of problem solving strategies, for controlling the combined case-based, model-based and rule-based reasoning, and for sustained learning. The problem solving is goal driven, based on an explicit representation of the application's task structure. The intensive use of knowledge in all processes is captured by an underlying 3-step reasoning cycle of ACTIVATE - EXPLAIN - FOCUS: Activate relevant concepts and generate plausible hypothesis, attempt to justify an hypothesis by producing the strongest possible explanation for it, and focus by selecting one hypothesis as the most plausible one.

All knowledge concepts are defined within a single, integrated semantic network. Thus, diagnosis task concepts, such as "symptom" and "diagnostic-hypothesis", as well as learning task concepts, such as "case-indexing" and "failure-generalisation", are defined within the same unified structure as general domain concepts such as "appendicitis" and "fever", and specific case-related domains terms as "Patient#123456" and "current-radiation-dosage". The dual view of cases (and rules) as separate object structures, and at the same time integrated at the substructure level into the common knowledge model, is illustrated in figure 3.



**Figure 3: Integrating Cases and General Knowledge**

Cases are frame structures whose terms are defined within the model of deeper knowledge. In that model, concepts and relations are all represented as frames, interlinked into a dense semantic network.

The frame-based CreekL language defines a concept in terms of a *prototype* - i.e. as a set of typical properties. It is an extension of SFL, the Sintef Frame Language [Aakvik-91]. As for similar frame languages (e.g. [Bobrow-77, Greiner-80, Lenat-89]), *typical properties* gets

inherited by more specialised concepts and instances. The inheritance is not absolute, however, and an inherited property may be overridden by a specific, local property (default inheritance). Thus, this frame system is different from frame representations based on predicate logic, like the KL-ONE [Brachman-85] and KRYPTON [Brachman-83] systems, which assume a closed world, and do not have default inheritance.

It is important to note that the methods in CREEK assume an open, changing world. The basic inferencing method is abductive rather than deductive, which means that the system will always try to produce the strongest possible explanation to justify its hypotheses. It will search for - or construct - explanations by using cases or general domain knowledge together with contextual information from the environment. The generation and evaluation of explanations are based on the assignment of different values of explanatory strength to different relationships - and combinations of relationships - within the knowledge model. A sequence of "causes" relationships, for example, will typically have a greater explanatory strength than a sequence of "cooccurs-with" relations.

The CREEK architecture addresses the interactiveness problem by assuming a competent user, who takes an active role in the problem solving process. When no satisfactory explanation can be produced within the system, the user is questioned. This may be a request to confirm an hypotheses, to supply additional information, or to test out a proposed conclusion. As in the PROTOS system [Porter-90], if deficiencies or errors are discovered within the general knowledge, the user will be requested to solve the incoherence (or log it onto a file for later inspection) and update the knowledge base.

The main target for the learning process in CREEK is the case base. But, as previously mentioned, it may also learn general knowledge through interaction with the user during problem solving. There is no inductive learning of explicit, generalised concept definitions or rules in CREEK. If a solution was derived by modifying a previous solution, a new case is stored and difference links between the two cases are established. A new case is also created after a problem has been solved from rules or the deeper knowledge model. In figure 4, the learning algorithm is shown.

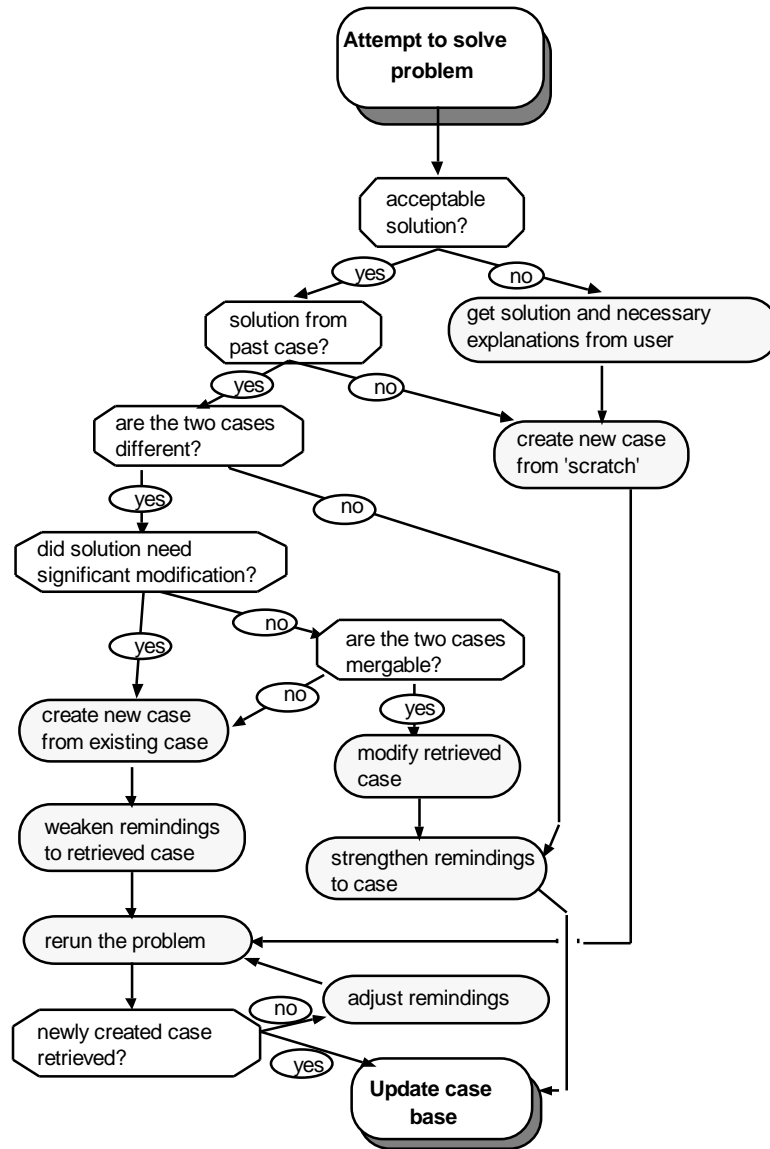
Learning by retaining specific problem situations seems to be the 'right' learning approach for interactive and adaptive knowledge-based systems, for two reasons. By learning specific knowledge, the system makes as few assumptions about future problems as possible. Traditionally, learning in AI has been synonymous with learning of generalisations. Learning of generalisations is more difficult and less useful for problem solving in a changing environment. The problem is, of course, what to leave out when forming generalisations. The case based approach do not 'generalise away' specific knowledge during the learning process, but leaves that to the problem solving step, where the case matching process normally will perform an implicit generalisation. But this is a generalisation within an actual problem solving context, where the current situation determines what should be generalised away and what should be kept as specific properties of the problem. The user can actively take part in this process.

The aim of a Creek system is to serve as an active assistant for the user in a decision-making process. Through its specific and general knowledge, and the corresponding inference and reasoning methods, the system is able to serve as an interactive discussion partner to the user. It is hard to see that such a behaviour can be realised by other than knowledge-based methods. A case-based approach ensures that the system continually becomes more competent, and is able to adapt to a changing environment, by retaining and learning from each problem solving experience.

## 4.2 *Case-Based Systems Applications*

A rapidly growing number of application systems, using the case-based reasoning approach,

are under development [DARPA-91, IEEE-92]. Some systems have also been fielded and are in regular use (e.g. [Hennesy-91, Brown-91]). At the University of Trondheim we are working together with the research institute SINTEF to develop a case-based decision



**Figure 4: Case Learning in Creek**

The figure shows the main steps of the learning algorithm. Rounded boxes are processes while polygons are branching points.

support system for drilling operations [Nordbø-92]. During a drilling process a lot of data is registered, and a lot of information is available for the persons that plan and supervise the drilling operations. The problem is, given a particular situation in planning, supervision, or fault-finding, to be able to remember and utilise the right information from the past. The task of the case-based decision-support system is to assist in this process by comparing a current situation to previous ones, and suggest actions, criticise a user's suggestions, predict consequences, etc., on the basis of earlier experience. A major advantage of the case-based approach is that information is kept *within its total context*, and not split up into pieces. Together with a model of general domain knowledge, these chunks of information and

knowledge - i.e. the cases - enables the *system and user together* to better understand a particular situation, and *jointly* suggest, critique, justify and take appropriate actions.

## 5. Conclusion

In order to improve problem solving competence and robustness in computer-based decision-support systems, and to enable sustained learning within a complex and changing real world problem solving environment, we should give the systems as good a start as possible. There is still no other way to do this than to let them share essential parts of *our* knowledge about their task domains and interaction environments. The methods to achieve this is captured within the knowledge-based approach to AI. A system's learning methods should then use the existing knowledge, together with information from the environment (e.g. interacting with a user), to learn from its failures and successes. Real world learning usually take place at the 'fringe' of what is already known, and this approach assures a strong guidance of the learning process, both from the system's existing knowledge and from its interaction with the environment. Adaptability should be ensured by capturing the specific knowledge that lie in new experiences. This is precisely what a knowledge-intensive, case-based method enables. The CREEK system has shown how this can be realised in practise.

It is hard to see how the problems identified by the critics can be solved *outside* the knowledge-based paradigm. It is plenty of room for extending and improving current methods, based on recent promising research. The case-based approach presented here is a way to achieve this.

Case-based reasoning is a rather novel, but rapidly growing area of research and development. We should expect to see an increasing number of successful case-based decision support systems in the not-so-far future, and thereby also get a case-based support for the paradigm advocated in this paper.

## Acknowledgements

The research reported here was initiated while working in the Artificial Intelligence Laboratory, Free University of Brussels - VUB. Filip Rademakers challenged my view through many stimulating discussions, and by commenting on an initial draft of the paper. Helpful comments and view points has also been given by Walter Van de Velde, Cuno Duursma, Inge Nordbø, Eric Monteiro, Dag Svanes and Tor Busch.

## References

### Aakvik-91

Geir Aakvik, Agnar Aamodt, Nordbø: A knowledge Representation Framework Supporting Knowledge Modelling. *Proceedings EKAW-91*, Fifth European Knowledge Acquisition for Knowledge-based Systems Workshop, Crieff, Scotland, May 1991.

### Aamodt-90

Agnar Aamodt: Knowledge-intensive case-based reasoning and learning. *Proceedings of ECAI-90*, Ninth European Conference on Artificial Intelligence, Stockholm, August 1990.

### Aamodt-91a

Agnar Aamodt: *A knowledge-intensive, integrated approach to problem solving and sustained learning*. Ph.D. Dissertation, University of Trondheim. May 1991.

### Aamodt-91b

Agnar Aamodt: *Problem Solving and Learning from Experience; An Introduction to Case-Based*

Reasoning. *NAIM - Nordisk AI Magasin*, Årgang 6, no. 1, 1991. pp.19-25.

**Aamodt-92**

Agnar Aamodt, Bart Benus, Cuno Duursma, Christine Tomlinsen, Ronald Schrooten, Walter Van De Velde: Task features and their use in CommonKads. *KADS-II Report*, KADS-II/TI.5/VUB/TR/014/1.0. Free University of Brussels - VUB, 1992.

**Althoff-92**

Klaus-Dieter Althoff: Machine learning and knowledge acquisition in a computational architecture for fault diagnosis in engineering systems. Proceedings of the ML-92 Workshop on Computational Architectures for Machine Learning and Knowledge Acquisition. Aberdeen, Scotland, July, 1992.

**Bobrow-77**

Daniel Bobrow, Terry Winograd: An overview of KRL, a knowledge representation language. *Cognitive Science*, Vol. 1, no. 1, 1977. pp 3-46. (Also in R.J Brachman, H.J. Levesque: Readings in knowledge representation. Morgan Kaufmann, 1985. pp 263-285.)

**Brachman-83**

Ronald Brachman, Richard Fikes, Hector Levesque: KRYPTON, A functional approach to knowledge representation. In: Ronald Brachman, Hector Levesque: *Readings in knowledge representation*. Morgan Kauffman. 1985. pp 411-430.

**Brachman-85**

Ronald Brachman, S. Schmolze: An overview of the KL-ONE knowledge representation system. *Cognitive Science*, Vol 9, no 1, pp 3-28. January 1985.

**Brooks-91a**

Rodney Brooks: Intelligence without representation. *AI Journal*, Vol 47, no 1-3, January 1991, pp. 139-160.

**Brooks-91b**

Rodney Brooks: Intelligence without reason. *Proceeding of IJCAI-91*, Sydney. Morgan Kaufmann, 1991.

**Churchland-90**

Paul Churchland, Patricia Churchland: Could a machine think? *Scientific American*, Vol 262, no.1, January 1990.

**Clancey-89**

W.J. Clancey: The frame of reference problem in the design of intelligent machines. *Commentary on the Twenty-Second Carnegie-Mellon Symposium on Cognition*. 1989.

**Dreyfus-86**

Hubert L. Dreyfus, Stuart E. Dreyfus: *Mind over machine; The power of human intuition and expertise in the era of the computer*. Free Press, New York, 1986.

**Eggen-90**

Jorun Eggen, Astrid M. Lundteigen, Mette Mehus: Integration of knowledge from different knowledge acquisition tools. In *EKA-90, Fourth European Knowledge Acquisition for Knowledge-Based Systems Workshop*, Amsterdam, June, 1990. pp 123-142.

**Greiner-80**

Russel Greiner, Doug Lenat: A representation language language. *Proceedings AAAI-80*, Morgan Kaufmann, 1980. pp. 165-169.

**Lenat-89**

Doug Lenat, R. Guha: *Building Large Knowledge-Based Systems; Representation and Inference in the CYC Project*. Addison-Wesley, 1989.

**Guha-90**

R. Guha, Doug Lenat: CYC; A midterm report. *AI Magazine*, Fall 1990, pp 33-59.

**Lenat-87**

Doug Lenat, Edward Feigenbaum: On the thresholds of knowledge. *AI Journal*, Vol. 47, no 1-3, January 1991, pp 185-250.

**Murray-88a**

Kenneth Murray, Bruce Porter: Developing a tool for knowledge integration; initial results.

*Proceedings from the Third Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, November 1988. 13 pgs.

**Newell-80**

Allen Newell: Physical symbol systems. *Cognitive Science*, 4, pp. 135-183, 1980.

**Newell-82**

Allen Newell: The knowledge level. *Artificial Intelligence*, Vol.18, 1982. pp 87-127.

**Nordbø-92**

Inge Nordbø, Pål Skalle, Jostein Sveen, Geir Aakvik, Agnar Aamodt: *Reuse of experience in drilling - Phase 1 Report*. SINTEF DELAB and NTH, Div. of Petroleum Engineering. STF 40 RA92050 and IPT 12/92/PS/JS. Trondheim 1992.

**Plaza-93**

Enric Plaza, Agnar Aamodt, Ashwin Ram, Walter Van de Velde, Maarten Van Someren: Integrated learning architectures. To appear in: Proceedings of the ECML-93, European Conference on Machine Learning. Vienna, 1993.

**Porter-90**

Bruce Porter, Ray Bareiss, Robert Holte: Concept learning and heuristic classification in weak theory domains. *Artificial Intelligence*, vol. 45, no. 1-2, September 1990. pp 229-263.

**Rademakers-92**

Filip Rademakers, Rolf Pfeifer: The role of knowledge-level models in situated adaptive design. *Proceedings, ECAI-92*.

**Ram-92**

Ashwin Ram, Michael Cox, S. Srinivasan: An architecture for integrated introspective learning. Proceedings of the ML-92 Workshop on Computational Architectures for Machine Learning and Knowledge Acquisition. Aberdeen, Scotland, July, 1992.

**Riesbeck-89**

C. Riesbeck, R. Schank: *Inside Case-based reasoning*. Lawrence Erlbaum, 1989.

**Searle-80**

Minds, Brains, and Programs. *Behavioral and Brain Sciences*, 3, 3, pp. 417-424. 1980.

**Shavlik-90**

Jude Shavlik, Tom Dietterich: *Readings in Machine Learning*. Morgan Kaufmann, 1990.

**Smith-82**

Brian C. Smith: Reflections and semantics in a procedural language. *MIT Technical Report, LCR-TR-272*, 1972.

**Steels-92**

Luc Steels: Reusability and configuration of applications by non-programmers. *VUB AI Memo, 92-4*. Free University of Brussels - VUB. 1992.

**Van de Velde-88**

Walter Van de Velde: *Learning from experience*. Ph.D Dissertation, Free University of Brussels - VUB. 1988.

**Van de Velde-92**

Walter Van de Velde, Agnar Aamodt: Machine learning issues in CommonKADS. *KADS-II Report, KADS-II/TII.4.3/TR/VUB/002/3.0*, Free University of Brussels -VUB, 1992.

**Van Lehn-91**

Kurt Van Lehn (ed): *Architectures for Intelligence*. Lawrence Erlbaum, 1991.

**Weintraub-92**

Michael Weintraub (ed): *Proceedings of the ML92 Workshop on Computational Architectures for Machine Learning and Knowledge Acquisition*. Aberdeen, Scotland, 1992.

**Wielinga-92**

Bob Wielinga, Walter Van de Velde, Guus Screiber, Hans Akkermans: Towards a unification of knowledge modeling approaches. *Knowledge Acquisition Journal* (Forthcoming)

**Winograd-86**

Terry Winograd, Fernando Flores: *Understanding computers and cognition*. Ablex, 1986

**Winograd-90**

Terry Winograd: Thinking machines. Can there be? Are we? In D.Partridge, Y.Wilks: *The foundations of artificial intelligence*. Cambridge University Press. 1990.