Assumptions of Problem-Solving Methods and their Role in Knowledge Engineering

Richard Benjamins and **Dieter Fensel** and **Remco Straatman** [1]

Abstract. A problem-solving method describes a reasoning process that efficiently achieves a goal by applying domain knowledge. However, a problem-solving method cannot directly be applied because of the existence of a gap between, on the one hand, a problem-solving method and the domain knowledge it uses, and, on the other hand, a problem-solving method and the goal that it is supposed to achieve. In this paper, we distinguish two types of assumptions based on an architecture of problem-solving methods, that are able to bridge the gap: one type of assumption is used to strengthen a problem-solving method, and the other to weaken the goal to be achieved. We also show how the effect of one assumption type can be substituted by the effect of the other type, and refer to this as "the law of conservation of assumptions".

1 Introduction

The notion of *problem-solving method* (PSM) is present in many current knowledge engineering frameworks such as Generic Tasks [6], Role-Limiting Methods [15], KADS [19], CommonKADS [20], the Method-to-Task approach [16], Components of Expertise [21], GDM [24]. Libraries of PSMs are described in [1, 2, 6, 18].

Problem-solving methods can be used to efficiently achieve goals of tasks through the application of domain knowledge [11]. There are however two relations that need to be understood before PSMs can actually be used: from the PSM to the goal it is supposed to achieve, and from the PSM to the domain knowledge that it uses. Figure 1 illustrates these two relations, which can be considered as gaps to be bridged before a PSM can be applied. The basic idea is that a PSM is applicable to achieve a particular goal in a particular domain, only under some assumptions. In other words, assumptions can bridge the gaps.

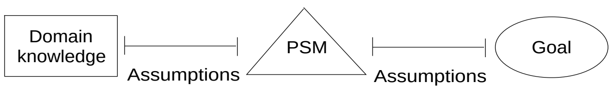

Figure 1. The two gaps that isolate a PSM, and that are bridged by assumptions.

Well-known examples of assumptions in diagnosis include the single-fault and the independence of causes assumptions, and the availability and completeness of fault models.

A PSM describes the reasoning steps and the types of knowledge needed to perform a task, in a domain and implementation independent manner. Most problems tackled with knowledge-based systems are inherently complex and intractable, that is, their time complexity is NP-hard [4, 3, 17]. PSMs make such hard problems practically feasible by making assumptions about the domain knowledge and the possible inputs, or about the precise definition of the functionality.

The aim of this paper is to show that *assumptions* form good candidates to bridge the identified gaps around a PSM, and thus solve the problem of applying a PSM to efficiently achieve a goal in a particular domain. As we will see, the architecture of a PSM gives rise to the identification of two types of assumptions. A main insight of the paper is that the joint effort of the two types of assumption is constant, and that they can compensate each other (called "The law of conservation of assumptions"). This provides a new view on the application of PSMs, and on knowledge-based system development in general.

The structure of the paper is as follows. In Section 2, we present the architecture of a PSM, and based on that, identify two types of assumptions. In Section 3, we describe properties of the assumption types, and suggest a view on assumptions which describes how they relate to, and can compensate each other. Section 4 concludes the paper by pointing out implications for Knowledge Engineering.

2 Architecture of PSMs

In this section, we first discuss the parts of a PSM and their internal relationships, and then the PSM's relation with its environment. These relations naturally lead to the identification of two types of assumptions.

2.1 The different parts of a PSM

A PSM consists of three interrelated parts (see Figure 2).

Functional specification PSM_{fun} is a declarative description of the input-output behavior of the PSM and describes what can be achieved by the PSM.

Requirements $PSMreq$ describes the domain knowledge needed by the PSM to achieve its functionality. Examples of such requirements in a parametric design task [10] include the availability of heuristics that link violated constraints to possible repair actions, the fact that a preference relation must describe a complete ordering, the existence of a causal relationship, etc. The requirements describe what a PSM expects in return for the functionality it provides.

Operational specification PSM_{op} specifies a reasoning process which delivers the specified functionality if the required knowledge is provided. It represents the link between the functionality and the knowledge requirements of a method, and consists of inferences

[1] SWI, University of Amsterdam, Roetersstraat 15, NL-1018 WB, Amsterdam, the Netherlands, e-mail {richard,dieter,remco}@swi.psy.uva.nl

ECAI 96. *12th European Conference on Artificial Intelligence*
Edited by W. Wahlster
Published in 1996 by John Wiley & Sons, Ltd.

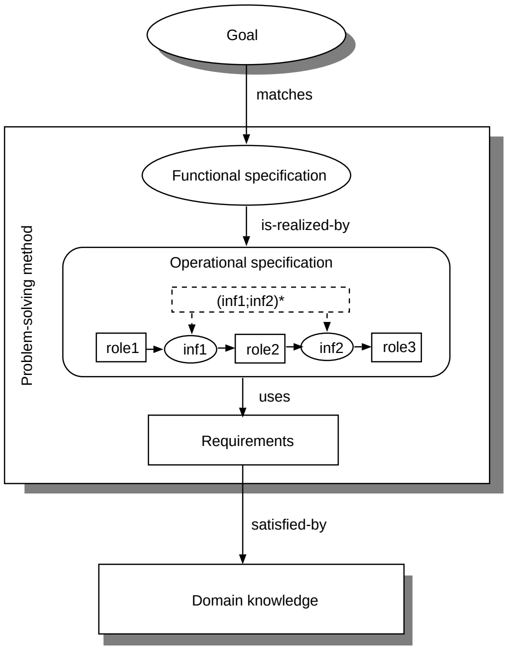

Figure 2. The architecture of a PSM.

and the knowledge and control-flow between them. The inferences specify the reasoning steps that are used to accomplish the functionality of the method. They are described by their input/output relation and can be achieved by either a method (which means that a PSM can be hierarchically decomposed) or a primitive inference (an atomic reasoning step which is not further decomposed). The knowledge flow takes place through roles, which are stores that act as input and output of inferences. Finally, the control of a PSM describes the order of execution of the inferences.

Relations between the parts The internal relationship between the functional and operational descriptions of the method has to be established. One has to ensure that, assuming that the knowledge requirements are satisfied, the operational description describes a way to achieve the functionality ("is-realized-by" in Figure 2). Because the description of the operational specification requires a logic over states, we use dynamic logic [14] to formalize this obligation.[2]

$$\models PSMreq \rightarrow (< PSM_{op} > true \wedge [PSM_{op}]PSM_{fun}). \quad (1)$$

This obligation states that, given that the requirements hold, the operational specification will terminate and its input-output behavior will be identical to the functional specification. It means that, given the requirements, the operational part of a PSM can be used to realize the functional specification.

2.2 External relations of a PSM - two assumption types

Based on the different parts of a PSM, two interfaces to its environment can be distinguished: the requirements on domain knowledge, and the functionality which relates it to the goal that the PSM can achieve. Each interface leads to the identification of an assumption type.

Knowledge requirements: ontological assumptions In downward direction, a PSM has knowledge requirements, specifying the needed domain knowledge ("satisfied-by" in Figure 2). We will refer to them as *ontological* assumptions, because they are reminiscent of ontological commitments [25]. The domain knowledge has to imply the ontological assumptions:

$$\models domain\ knowledge \rightarrow ontological\ assumptions \quad (2)$$

Functionality: teleological assumptions In upward direction, the relation between the PSM functionality and the goal to be achieved has to be established ("matches" in Figure 2).

Ideally, one would like that the goal is implied by the PSM functionality:

$$\models PSM_{fun} \rightarrow goal \quad (3)$$

However, for intractable problems such as design, planning or diagnosis, this is not realistic, so sub-optimal solutions have to be accepted. Sub-optimal solutions are solutions for weakened versions of the original goal. This means that a goal is implied by the PSM functionality only under some assumptions, which we will refer to as *teleological* assumptions.

$$\models PSM_{fun} \rightarrow (teleological\ assumptions \rightarrow goal) \quad (4)$$

Such assumptions describe a translation of the original goal into a weaker one with a smaller complexity, or which can be solved more efficiently for typical cases.

3 Assumptions of PSMs

In this section, we first present some examples of assumptions as they appear in the literature (Section 3.1). Then, we discuss some interesting properties of assumption types (Section 3.2) and show how they relate to each other (Section 3.3).

3.1 Assumptions in the literature

Assumptions of reasoning methods play an important role in AI-research. Table 1 shows some examples taken from the literature on PSMs, along with a short explanation.

3.2 Strengthening of PSMs and weakening of goals

Ontological assumptions embody a *strengthening* of the functionality of a PSM, in the sense that, the more domain knowledge is available, the stronger the PSM can be (see Figure 4). Ontological assumptions reduce the complexity of the part of the problem that is solved by the PSM. In terms of complexity analysis, the domain knowledge or the user of the system is used as an "oracle" that provides a solution for complex parts of the problem. For example, the availability of good "propose" knowledge for elevator design, enables better initial designs, and thus the "revise" step can be simplified [10].

Teleological assumptions embody a *weakening* of the goal that can be achieved, by introducing assumptions about the precise problem type. The desired functionality of the system is thus reduced and therefore also the complexity. For example, in diagnosis a particular PSM might only be able to find single faults. If the original task goal

[2] The two standard modal operators of dynamic logic are used here: the box-operator, $[\alpha]\phi$, defines ϕ to hold in every terminal state of program α, and the diamond-operator, $\langle\alpha\rangle\phi$, defines ϕ to hold in at least one terminal state of α.

assumption name	*explanation*
Independence of hypotheses	An individual hypothesis explains a set of observations regardless of the other hypothesis [1].
Non intermittency	Observations do not change during diagnostic reasoning [1].
Fault models complete	All possible faults are represented in fault models, which allows to derive that a component is correct if none of its fault models holds (GDE+ [23]).
Rule out knowledge	Knowledge that reduces the number of individual hypotheses. [4] shows that an abduction problem becomes tractable if this number is reduced from n to log n.
Unique effects	A causal relation uniquely describes its effects (i.e. "X causes Y1 or Y2" is not allowed) (hierarchical abduction [7]).
Ordered monotonic abduction problem	A composite hypothesis explains at least as much as its elements, each individual hypothesis has a different plausibility and there is a total ordering among plausibilities. This assumptions makes abduction tractable [4].
Belief in observations	Observations can be trusted [22].
Design is correct	The original design of the system to be diagnosed is correct [8].
Fixes do not interact	The results of applying fixes do not cancel each other, so they can be applied in any order [10].
Untangled concept hierarchy	A concept should not be a subconcept of more than one superconcept and the hierarchy must be more or less balanced (hierarchical activation [13]).

Table 1. Some examples of assumptions taken from the literature on PSMs.

is to find any kind of fault (including multiple faults), then the PSM can achieve this goal under the single-fault assumption. Thus in fact the PSM achieves a weaker goal.

In this paper, we deal with modal formulae such as

$$\phi \rightarrow [\alpha]\psi \tag{5}$$

where ϕ, ψ are first-order formula, "$\rightarrow$" denotes implication, and α state-transitions. We require a modal extension of first-order logic to formalize the operational description of a problem-solving method. Such an operational specification defines transitions from initial states into successor states. $\phi[\alpha]\psi$ can be read as: whenever α starts from an initial state satisfying ϕ, it must terminate (if it terminates) in a state satisfying ψ. Formally, the semantics of this formula is the set of all initial states that either do not fulfill ϕ, or lead to successor states of α that fulfill ψ.

If we instantiate this formula with our expressions of Formula 1 and 4 we get:

$$\models ontological\ assumptions \rightarrow \\ [PSM_{op}]\ (teleological\ assumptions \rightarrow goal) \tag{6}$$

Both types of assumptions *weaken* the demands on $[PSM_{op}]$. The *ontological assumptions* limit the set of initial states for which we define restrictions on the behavior of PSM_{op}. The *teleological assumptions* weaken the formula that defines restrictions for the terminal states of PSM_{op}. Only terminal states that fulfill these assumptions must fulfill the goal.

Formally, a formula ϕ is *weakened* to a formula $\phi\prime$ if each model of ϕ is also a model of $\phi\prime$, i.e. $\phi \models \phi\prime$. A formula ϕ is *strengthened* to a formula $\phi\prime$ if each model of $\phi\prime$ is also a model of ϕ, i.e. $\phi\prime \models \phi$.

Formula 6 can therefore be weakened by strengthening the *ontological assumptions* or by strengthening the *teleological assumptions*. In the first case, the requirements on domain knowledge increase. In the second case, we increase the teleological assumptions under which the goal must be achieved. That is, we restrict the set of problems that must be solved by the method. Strengthening of Formula 6 can be analogically defined in the reverse way. Figure 3 illustrates an example for the necessity to weaken the definition of an operational specification of a PSM. The state transition $(w_{1.2}, w_{2.2})$ violates the formula $\phi \rightarrow [\alpha]\psi$. We can remedy this problem either by strengthening the *teleological assumptions* and therefore weaken the formula "*teleological assumptions* $\rightarrow$ *goal*", or by strengthening the *ontological assumptions*.

As a consequence, weakening (respectively strengthening) of one type of assumptions can be compensated by strengthening (respectively weakening) of the other type of assumptions. This effect is what we call later the *conservation law of assumptions*.

In the following, we use a rather informal but intuitive notion of "$+$" and "$-$" to denote respectively strengthening and weakening of formulae. When necessary, the formal semantics can always be given in the terms of this subsection.

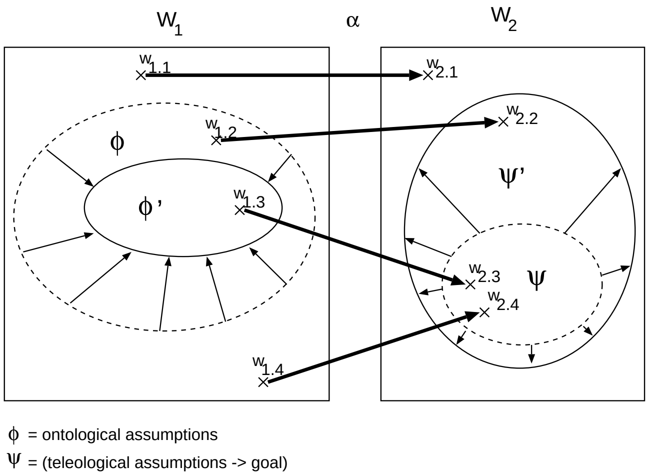

Figure 3. Illustrating the weakening of $\phi \rightarrow [\alpha]\psi$.

3.3 The law of conservation of assumptions

The main insight of our theory of assumptions can be seen in Figure 4, namely that the required net effect of strengthening and weakening is constant with respect to the goal to be achieved. This is apparent from the fact that both PSMs are applied to the same goal (the ellipses), but the first PSM makes more ontological assumptions, less teleological ones, and is thus stronger than the second PSM. In the following, we will show that the joint effort of ontological and teleological assumptions is constant.

Using our notation we can rephrase Formula 4 to:

$$goal - teleological\,assumptions = PSM_{fun} \tag{7}$$

Similarly, Formula 1 can be rephrased to:

$$PSM_{fun} = PSM_{op} + ontological\,assumptions \tag{8}$$

Substituting Formula 8 in 7 yields the formula:

$$\begin{aligned} &goal - teleological\,assumptions = \\ &PSM_{op} + ontological\,assumptions \end{aligned} \tag{9}$$

Formula 9 states that the goal minus the teleological assumptions is equal to the PSM plus its ontological assumptions. This formula is valid in the framework of tasks, PSMs and assumptions and opens interesting possibilities in Knowledge Engineering. In the following, we will focus on one of them and in Section 4, we will briefly suggest other possibilities.

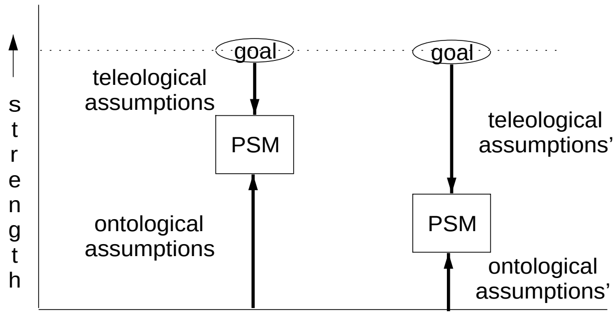

Figure 4. The law of conservation of assumptions.

We reformulate 9 to:

$$\begin{aligned} goal - PSM_{op} = \\ teleological\,assumptions + ontological\,assumptions \end{aligned} \tag{10}$$

In the knowledge engineering situation which we are interested in, we have a goal to be achieved and a particular PSM (selected from a library with reusable PSMs) that might achieve the goal. In other words, $goal$ and PSM_{op} are constants. Consequently, their difference is also constant, which we will call Δ:

$$goal - PSM_{op} = \Delta \tag{11}$$

Substituting 11 in 10, we get

$$ontologial assumptions + teleological assumptions = \Delta \tag{12}$$

Thus, the sum of ontological and teleological assumptions is constant for a fixed $goal$ and PSM_{op}. That is, we can replace an assumption of one type by an assumptions of the other type, requiring that they deliver both the same strength. We refer to this as "the law of conservation of assumptions".

We illustrate the law with the single-fault assumption, which can be regarded as a typical example of a teleological assumption in diagnosis [12]. Some diagnostic problem-solving methods only work for problems with at most one fault. By employing this assumption they can reason more efficiently than methods which deal with multiple faults [9]. The single-fault assumption can also be viewed as a requirement on domain knowledge [8]. If each possible set of faults is represented as a single fault, a method which finds only single faults is still able to solve each diagnostic problem in the given application domain. For example, the malfunction of a single chip with four gates can appear as a set of four faults (one for each gate), or as a single fault if the chip itself is the represented entity. Thus, it depends on our assumptions about the domain knowledge whether we have to weaken the goal in order to guarantee its achievement with a PSM based on single faults.

Another illustration is the assumption "fault models complete" (see Table 1). As an ontological assumption, this states that we can diagnose all possible faults since they all are represented in fault models. As a teleological assumption, it states that the PSM will work for cases where only the faults mentioned in the fault models occur, which could be enough to cover routine diagnosis.

A final illustration of the law is the assumption "fixes do not interact". As an ontological assumption it means that fixes can be applied in any order. As a teleological assumptions it states that we accept that some possible solutions cannot be found.

In our knowledge engineering situation where we want to achieve a goal by a PSM (which is selected from a library) in a particular domain, the conservation law is relevant because we can, within the boundaries of PSM_{op}, weaken or strengthen a reusable PSM. This can be done according to (a) the available domain knowledge, (b) the effort required to model further domain knowledge, (c) how much we are willing to weaken the original goal. Teleological assumptions have to be made if the ontological assumptions cannot be satisfied to the extent that the operational PSM description enables the achievement of the original goal.

4 Conclusion and future work

In this paper, we proposed a solution to the problem of how to bridge the gap between a problem-solving method and, on the one hand, the goal to be achieved, and, on the other hand, the particular domain that it uses. The solution consists of the identification of two types of assumptions of PSMs: ontological and teleological assumptions. The gap can be bridged by strengthening the PSM (i.e. adding more ontological assumptions), or by weakening the goal it has to achieve (adding teleological assumptions). We have shown how assumptions can be moved around from teleological to ontological according to the law of conservation of assumptions, while the PSM remains applicable to the same goal.

Making explicit the assumptions of a PSM about the domain knowledge is a way to deal with the interaction problem. The interaction problem [5] states that domain knowledge cannot be represented independently of how it will be used in reasoning. Vice versa, a PSM and its specific variants cannot be constructed independently of assumptions about the available domain knowledge.

Besides bridging the identified gaps, our view on assumptions has other interesting implications for the knowledge engineering practice which set out interesting lines for future research.

Methodology The first implication relates to a methodology for the construction of KBSs. A KBS can be seen as a configuration of several cooperating PSMs, where each PSM has to achieve a particular goal. The ontological assumptions of a PSM indicate the requirements on (domain) knowledge, and thus define goals for the knowledge acquisition process. According to the methodology first the needed domain knowledge has to be acquired (thus fulfilling the ontological assumptions), and only if this cannot be done, then some ontological assumptions are moved towards the goal and become teleological assumptions, that weaken the goal. This strategy guarantees that the strongest possible PSMs are built given a particular domain and goal.

Reuse Another implication relates to reuse. By distinguishing ontological and teleological assumptions, we enhance the reusability of PSMs. Ontological assumptions explicit the relations between a PSM and domain knowledge in domain-independent terms. That is, they speak about meta-characteristics of domain knowledge, and not about the domain knowledge itself. Teleological assumptions enable a PSM to achieve a variety of (stronger and weaker) goals, and thus a PSM can be used to achieve more than one particular goal. Moreover, the two types of assumptions are interchangeable, allowing for even more flexibility.

In the context of a PSM library, a method can only be chosen if its ontological assumptions are fulfilled by domain knowledge. Ontological assumptions can be viewed as indices to access the library, and we can view the assumptions as proof obligations for the domain knowledge. Moreover, if a PSM is not applicable due to unsatisfiable ontological assumptions, another PSM can be selected from the library.

PSM construction Formula 10 can also be used in the construction of PSM_{op} for a given goal. For instance in [11], we propose a process of method construction that starts with a weak problem-solving method (e.g. generate & test) and transforms this until an acceptable problem solver is constructed. Some of these transformation steps do not change the functionality, while others introduce new assumptions. The formula we presented here can be used as an invariant for this process: it should hold in each state of the construction process. In this way it also defines which combinations of assumptions and changes in PSM_{op} are allowed. Because we introduced the formula into this process, we are able to perform steps which do not preserve the functionality (i.e. introduce assumptions) of the PSM_{op} and still reason about them. This kind of construction steps are essential in constructing PSMs and are missing from comparable frameworks in software engineering.

ACKNOWLEDGEMENTS

This research was supported by the Netherlands Computer Science Research Foundation with financial support from the Netherlands Organization for Scientific Research (NWO).

REFERENCES

[1] V. R. Benjamins, 'Problem-solving methods for diagnosis and their role in knowledge acquisition', *International Journal of Expert Systems: Research and Applications*, **8**(2), 93–120, (1995).
[2] *CommonKADS Library for Expertise Modeling*, eds., J. Breuker and W. van de Velde, IOS Press, Amsterdam, The Netherlands, 1994.
[3] T. Bylander, 'Complexity results for planning', in *International Joint Conference on Artificial Intelligence*, (1991).
[4] T. Bylander, D. Allemang, M. C. Tanner, and J. R. Josephson, 'The computational complexity of abduction', *Artificial Intelligence*, **49**, 25–60, (1991).
[5] T. Bylander and B. Chandrasekaran, 'Generic tasks in knowledge-based reasoning: The right level of abstraction for knowledge acquisition', in *Knowledge Acquisition for Knowledge Based Systems*, eds., B. Gaines and J. Boose, volume 1, 65–77, Academic Press, London, (1988).
[6] B. Chandrasekaran, T. R. Johnson, and J. W. Smith, 'Task-structure analysis for knowledge modeling', *Communications of the ACM*, **35**(9), 124–137, (1992).
[7] L. Console and D. Theseider Dupré, 'Choices in abductive reasoning with abstraction axioms', in *Proc. of the workshop on foundations of knowledge representation*, ed., G. Lakemeijer, Vienna, (August 1992). ECCAI.
[8] R. Davis, 'Diagnostic reasoning based on structure and behavior', *Artificial Intelligence*, **24**, 347–410, (1984).
[9] J.H. de Kleer and B.C. Williams, 'Diagnosing multiple faults', *Artificial Intelligence*, **32**, 97–130, (1987).
[10] D. Fensel, 'Assumptions and limitations of a problem-solving method: A case study', in *Proceedings of the 8th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, eds., B. R. Gaines and M. A. Musen, Alberta, Canada, (1995). SRDG Publications, University of Calgary.
[11] D. Fensel, R. Straatman, and F. van Harmelen, 'The mincer metaphor for problem-solving methods: Making assumptions for reasons of efficiency', in *Proceedings of the Knowledge Engineering: Methods and Languages Workshop (KEML-96)*, Paris-Orsay, (January 15-16 1996). (to appear).
[12] M. R. Genesereth, 'The use of design descriptions in automated diagnosis', *Artificial Intelligence*, **24**, 411–436, (1984).
[13] A. Goel, N. Soundarajan, and B. Chandrasekaran, 'Complexity in classificatory reasoning', in *AAAI–87*, pp. 421–425, (1987).
[14] D. Kozen, 'Logic of programs', in *Handbook of Theoretical Computer Science*, Elsevier Science Publ., B. V., Amsterdam, (1990).
[15] *Automatic knowledge acquisition for expert systems*, ed., S. Marcus, Kluwer, Boston, 1988.
[16] M. Musen, 'Overcoming the limitations of role-limiting methods', *Knowledge Acquisition*, **4**(2), 165–170, (1992).
[17] B. Nebel, 'Artificial intelligence', in *Essentials in Knowledge Representation*, ed., G. Brewka, (1995).
[18] F. Puppe, *Systematic Introduction to Expert Systems: Knowledge Representation and Problem-Solving Methods*, Springer-Verlag, Berlin, 1993.
[19] *KADS: A Principled Approach to Knowledge-Based System Development*, eds., A. Th. Schreiber, B. J. Wielinga, and J. A. Breuker, volume 11 of *Knowledge-Based Systems Book Series*, Academic Press, London, 1993.
[20] A. Th. Schreiber, B. J. Wielinga, R. de Hoog, J. M. Akkermans, and W. Van de Velde, 'CommonKADS: A comprehensive methodology for KBS development', *IEEE Expert*, **9**(6), 28–37, (December 1994).
[21] L. Steels, 'Components of expertise', *AI Magazine*, **11**(2), 28–49, (Summer 1990).
[22] P. Struss, 'Diagnosis as a process', in *Readings in Model-based Diagnosis*, eds., L. Console, J.H. de Kleer, and W.C. Hamscher, Morgan Kaufmann, (1992).
[23] P. Struss and O. Dressler, 'Physical negation – integrating fault models into the general diagnostic engine', in *Proc 11th IJCAI*, pp. 1318–1323, Detroit, (1989).
[24] P. Terpstra, G. van Heijst, B. Wielinga, and N. Shadtbolt, 'Knowledge acquisition support through generalised directive models', in *Second Generation Expert Systems*, eds., Jean-Marc David, Jean-Paul Krivine, and Reid Simmons, 428–455, Springer-Verlag, Berlin Heidelberg, Germany, (1993).
[25] G. van Heijst, *The Role of Ontologies in Knowledge Engineering*, Ph.D. dissertation, University of Amsterdam, May 1995.