A Conceptual and Formal Model of a Diagnostic Reasoner

Richard Benjamins\textsuperscript{1} and Manfred Aben\textsuperscript{2}

\textsuperscript{1} SWI, University of Amsterdam, Roetersstraat 15, NL-1018 WB Amsterdam, the Netherlands, richard@swi.psy.uva.nl

\textsuperscript{2} Unilever Research Laboratories, Olivier van Noortlaan 120, NL-3133 AT Vlaardingen, the Netherlands, Manfred.Aben@2488taux.uml.com

Abstract. The knowledge acquisition process can be supported by distinguishing different models in the development process of knowledge-based systems, where each model is dedicated to a specific phase. Two of these models are the conceptual model and the formal model. Conceptual models of knowledge-based systems facilitate initial system specification because they are easy to understand and construct. However, such models are often ambiguous and inconsistent, and contain hidden assumptions. The use of formal methods is a way to overcome these problems, and formalization becomes essential when we have to guarantee that system specifications are met, such as in safety critical systems. This paper presents a conceptual model and a formal model of a diagnostic reasoner, and includes a proof which shows that the high-level specification of our reasoner is ensured by the formal model presented.

1 Introduction and motivation

Conceptual modeling is a widely recognized activity in the process of knowledge-based system (KBS) development. A conceptual model (CM) is an informal, though highly structured, model for initial system specification [14]. In a CM, one specifies the required functionality of the system at a high level, without loosing oneself in details concerned with design or implementation. Conceptual models in Knowledge Engineering share the same motivation as the structured models in Object-Oriented Modeling [12] and Structured Analysis [20]. Due to its high-level, on the one hand, a CM facilitates initial system development but, on the other hand, leaves room for ambiguities and inconsistencies (e.g., inferences can be interpreted in more than one way [3]). A CM is only a first step in the development process of a knowledge-based system, and a long way is still to go before it can be implemented.

Current knowledge engineering approaches acknowledge this gap between a conceptual model and its implementation, and, to bridge it, they view KBS development as an incremental (though nonlinear) process, in which subsequently more detailed models are constructed. Several knowledge engineering approaches such as KADS [14], MIKE [2] and Vital [16] advocate the use of the following, increasingly more detailed, models: conceptual model (informal, but structured), formal model (language with a mathematically defined syntax and semantics), design model (specifies the system architecture, and the data structures), and

\*** Richard Benjamins is supported by the Netherlands Computer Science Research Foundation with financial support from the Netherlands Organisation for Scientific Research (NWO). The work has partly been supported by the HCM program, financed by the CEC.
implementation model (implementation of data structures in a computer language). The formal model is an essential model in this process when we deal with safety-critical systems, because it enables consistency checking and correctness proving of the models [13].

Formal models are a means to get rid of the ambiguities and inconsistencies in conceptual models. However, constructing formal models is difficult and time consuming. Recently, both in software and in knowledge engineering, several efforts have been made to connect the conceptual model with the formal model [15, 11, 18].

Diagnostic systems often operate under safety-critical conditions, for example in airplanes or in chemical process industries, where finding the right causes of symptoms is crucial. Diagnosis is a complex task for which many approaches and techniques have been put forward. It is therefore not easy to develop diagnostic knowledge-based systems for a particular application. Recently we have presented a conceptual analysis of diagnosis [4] which integrates many approaches and diagnostic systems in a uniform way. The analysis identifies the relevant goals (tasks) in diagnosis, and the way in which these goals can be achieved (by problem-solving methods). We showed how the analysis can be used to semi-automatically generate conceptual models of diagnostic strategies [5].

The aim of this paper has to be seen in the context of integrating conceptual and formal models. We will show how to construct a conceptual and a formal model of a diagnostic reasoner (a hypothesis generator). During the formalization process, we have to be precise about what we exactly mean with the conceptual model.

The structure of the paper is as follows. In Section 2, we present the conceptual model of our diagnostic reasoner. In Section 3, we describe its formalization, and prove that our model indeed describes the intended behavior. Section 4 discusses the formalization, and puts it in a broader perspective. Finally, Section 5 concludes the paper.

2 Conceptual modeling

In this section, we briefly explicate the ingredients of our analysis, and then present the conceptual model of our diagnostic reasoner, which is a hypothesis generator.

2.1 Conceptual analysis

**Task** A task has a goal and is characterized by the type of input it receives and the type of output it produces. A task is a specification of what needs to be achieved. For example, the goal of diagnosis is to find a solution that is consistent with all observations. A task can be decomposed into subtasks (by a problem-solving method).

**Problem-Solving method** A problem-solving method (or method, PSM) describes how the goal of a task can be achieved. It has inputs and outputs and decomposes a task into subtasks and/or primitive inferences. In addition, a method specifies the data flow between its constituents in terms of the input and output types. Control knowledge determines the execution order and iterations of the subtasks and inferences of a PSM.

**Primitive inference** A primitive inference (or inference) defines a reasoning step, with its inputs and outputs, that can be carried out using domain knowledge to achieve a goal. Inferences form the actual building blocks of a problem solver. When all tasks are decomposed, through PSMs, into inferences, and we connect inferences based on shared inputs and outputs, the corresponding