Logical Object as a Basis of Knowledge Based Systems

Xu Dianxiang (徐殿祥) and Zheng Guoliang (郑国梁)

Department of Computer Science and Technology, Nanjing University, Nanjing 210093

Received August 19, 1994.

Abstract

This paper presents a framework called logical knowledge object (LKO), which is taken as a basis of the dependable development of knowledge based systems (KBSs). LKO combines logic programming and object-oriented programming paradigms, where objects are viewed as abstractions with states, constraints, behaviors and inheritance. The operational semantics defined in the style of natural semantics is simple and clear. A hybrid knowledge representation amalgamating rule, frame, semantic network and blackboard is available for both most structured and flat knowledge. The management of knowledge bases has been formally specified. Accordingly, LKO is well suited for the formal representation of knowledge and requirements of KBSs. Based on the framework, verification techniques are also explored to enhance the analysis of requirement specifications and the validation of KBSs. In addition, LKO provides a methodology for the development of KBSs, applying the concepts of rapid prototyping and top-down design to deal with changing and incomplete requirements, and to provide multiple abstract models of the domain, where formal methods might be used at each abstract level.

Keywords: Knowledge based system, logic programming, object-oriented programming, specification, verification.

1 Introduction

Knowledge Based Systems (KBSs) are being now developed at an increasing rate, but there is no uniform approach to dependable KBSs which is a critical issue that could drastically limit the use of KBSs to safety critical applications\(^1\). So what we need is a framework for KBSs that should be used both for the modeling of the system behavior and as a means to an identification of tools and techniques to be used in the development. For the time being, people have not paid much attention to the problem of how to transfer to KBSs development and verification the experience gained in the last two decades in the area of software engineering. Accordingly, we argue that we can benefit from incorporating certain methodologies for KBSs as found in traditional software engineering.

Logic programming, not only as an implementation tool, but also as a formalism of knowledge, has been very popular in the research and development of KBSs

---

\(^1\) This work is supported by the High Tech Research and Development Programme of China.
because of its clear semantics, more expressive power, and capability of automatic reasoning. Current researches have shown, however, it is rather weak as a formalism for describing complex structures. In the last decade, many attempts have been made to combine it with object-oriented paradigm\cite{5-11}. The combined systems therefore support such notions as inheritance, modularity, reusability of knowledge, which have turned out to be extremely useful for the design and maintenance of large scale KBSs. But much of this work has introduced certain undesired features while accommodating object-orientation, and none is proposed for the specification and verification of KBSs.

In this paper, we present a framework called logical knowledge \textit{object} (LKO), which is taken as a basis for the dependable development of KBSs. LKO combines logic programming and object-oriented programming, where logical objects are viewed as abstractions with states, constraints, behaviors and inheritance. Logical objects in LKO have both declarative and operational interpretations. The process of problem solving is formally defined in the style of natural semantics and proven very simple and clear. A hybrid knowledge representation amalgamating rule, frame, semantic network and blackboard is available for both most structured and flat knowledge and the management of knowledge bases has been specified in the formal language \textit{Z}. Accordingly, LKO is well suited for the formal representation of knowledge and requirements of KBSs. Based on the framework, verification techniques are also explored to enhance the specification analysis and the validation.

But how does LKO support the development of KBSs? This is clearly one of the most significant problems. Typically, when developing a KBS, we may start with a small part of the available knowledge, implement this subset, observe the outcome, and tune the knowledge base. It is not what software engineering doctrine has been advocating, but is a natural human mode of operation. Based on this, we apply the concepts of rapid prototyping and top-down design to develop a KBS. The LKO development methodology is designed to deal with changing and incomplete requirements and to provide multiple abstract models of the domain. At each abstract level, there are six correspondent stages: informal requirement, conceptual model, formal LKO specification, specification verification, prototype and implementation. When errors are found in an abstract level, we should return to the previous stage or to the higher level. On the other hand, for any stage, there are a lot of correspondent models in different abstract levels. To support this paradigm, it is imperative that requirements represented in LKO be executable and be easily verified at all levels. Obviously, the requirement specification is incremental modification and refinement of that of the higher level. Thus the object-oriented methodology brings its ability to unifying the description of entities and their protocols into full play.

This paper is organized as follows. In Section 2, we shall describe the concepts of LKO, then in Section 3, we shall introduce the operational semantics in the style of natural semantics. In Section 4, we shall investigate the formal requirement specifications of KBSs. In Section 5, we shall explore the verification techniques based on the operational semantics. Finally, we shall give our conclusions and future work in Section 6.