Chapter XIV

Seamless Formalizing the UML Semantics Through Metamodels

José Luis Fernández Alemán and Ambrosio Toval Álvarez
University of Murcia, Spain

Evolution...is – a change from an indefinite, incoherent homogeneity, to a definite coherent heterogeneity.
Herbert Spencer (1820-1903), First Principles (1862) ch.16

Despite the fact that the Unified Modeling Language (UML) has been adopted by the Object Management Group (OMG) as the standard notation for use in Object-Oriented (OO) Systems Development, it still does not have a truly formal semantics. There is currently much effort directed towards formalizing particular aspects or models of UML. However, the literature gives little insight into the appropriate strategy for tackling this problem within an integrated basis including the language evolution. This chapter identifies and discusses three feasible strategies which can be applied to formalize UML. One of these strategies is selected to underpin the four-layer architecture on which UML is based. The approach is based on the soundness of algebraic specification theory, which, in addition, provides suitable theorem-proving capabilities for exploiting the UML formal model obtained. The formal models proposed are specified using an executable algebraic specification language called Maude.

INTRODUCTION

The UML (Object Management Group, 1999a; 1999b) has been adopted by the OMG as the standard OO Systems Development language. This language is experiencing a growing popularity, and it is an unavoidable reference in analysis and design of information systems, both in academia and industry. The UML stems from the merging of six earlier proposals and the invaluable contributions of about 40 authors.
In spite of that, the UML has often come under strong criticism since its appearance. The main reason for this has been the ambiguity and the lack of a formal definition of its semantics, which is still not sufficiently precise. For example, the UML static semantics is described by a semi-formal constraint language, the object constraint language (OCL), and the UML dynamic semantics is expressed in natural language. Particular problems concerning ambiguity, inconsistency and incompleteness have been identified (Reggio & Wieringa, 1999). This situation has hindered the construction of rigorous methods, precise UML artifacts and software tools to animate and formally manage UML models.

Unlike other modeling languages, UML has a distinguishing characteristic called extensibility, i.e., the ability to extend the language with new syntax and semantics features using its basic set of elements or building blocks. The UML modeling language supports extensibility by three extension mechanisms: **Stereotype**, **Constraint** and **TaggedValue**. Extensibility is sometimes considered to be a feature used mostly by academic or sophisticated users. Beware that mechanism extensions comprise less than 10% of the effort in building class diagrams (Fowler & Scott, 1997). In addition, the lack of CASE tools including this capability and the difficulty of precisely defining additional semantics to the UML result in the poor use of extensibility by most analysts.

The formalization of a modeling language can help to identify and remove these problems and also allows us to rigorously verify and manipulate the system models constructed. Thus, the goal is to combine the intuitive appeal of visual notations with the precision of formal specification languages. This chapter identifies and discusses three feasible strategies, which can be applied to formalize the UML. One of these strategies is selected and tailored to the UML four-layer metamodeling architecture. In contrast to other related approaches, we present a proposal for the integral formalization of the UML diagrams and their interrelations.

The UML syntax, static semantics and dynamic semantics are included in the same formal model in a seamless way. This means that the evolution of the UML metamodel is supported and formally described, which is a novelty from a theoretical viewpoint compared with other approaches. Therefore, the semantic framework provided faces up to the unpredictable and changeable nature of the UML, without limiting its ability to adapt the UML for needs of a particular domain.

In order to illustrate the approach, we present some examples concerning one of the UML diagrams, the Class Diagram. However, this approach is applicable to any other UML diagram or model, for example the UML Statechart Diagram (Fernández & Toval, 2000a).

The rest of the chapter is organized as follows: the second section discusses some issues that provide rationale for the framework presented, and presents a brief review of related work. We then proceed by introducing the main features of Maude through some examples that are used to present our proposal in the next section, which identifies three feasible strategies to formalize the UML. This section describes a procedure to formally support the extensibility mechanisms. The fifth section focuses on emerging trends in formalizing UML, and outlines the effort to be made in the future. The final section presents some concluding remarks. Previous background in algebraic specifications will help to understand the remaining sections, but this is not essential, provided that the reader refers to Appendix A.
Related Content

Modular Rule-Based Programming in 2APL
[www.irma-international.org/chapter/modular-rule-based-programming-2apl/35853/](http://www.irma-international.org/chapter/modular-rule-based-programming-2apl/35853/)

Specification and Checking of Dependency Relations between UML Models
[www.irma-international.org/chapter/specification-checking-dependency-relations-between/30544/](http://www.irma-international.org/chapter/specification-checking-dependency-relations-between/30544/)

Using Rules in the Narrative Knowledge Representation Language (NKRL) Environment
[www.irma-international.org/chapter/using-rules-narrative-knowledge-representation/35854/](http://www.irma-international.org/chapter/using-rules-narrative-knowledge-representation/35854/)

Formalizing UML Class Diagrams
[www.irma-international.org/chapter/formalizing-uml-class-diagrams/30541/](http://www.irma-international.org/chapter/formalizing-uml-class-diagrams/30541/)

UML Modeling Support for Early Reuse Decisions in Component-Based Development
[www.irma-international.org/chapter/uml-modeling-support-early-reuse/30572/](http://www.irma-international.org/chapter/uml-modeling-support-early-reuse/30572/)