A Pattern-Based Constraint Language for Metamodels

Dániel Varró

Nowadays, the Unified Modeling Language (UML) [5] has become the de facto standard modeling language of object-oriented system design. A wide range of software applications is being designed using this *unified* and *visual* formalism serving as the common communication platform between customers, system designers, and programmers. However, both academic and industrial applications have revealed several drawbacks of the language concerning, especially, its imprecise semantics, and the lack of flexibility in domain-specific environments.

The scope of **metamodeling** is broader than UML as *metamodeling is the paradigm of defining the syntax and semantics of modeling languages for arbitrary domains* preferably in a visual notation. In dominant metamodeling approaches (such as MML[1], or GME[3]), the *abstract syntax* of a domain is captured by (variants of) visual UML class diagrams, while the *static semantics* of a modeling language is typically formalized by using the textual Object Constraint Language (OCL) [4]. When specifying the dynamic behavior of a modeling language, the use of **graph transformation** [2, 6] is a powerful solution as it provides a mathematically precise underlying formalism without the loss of visual description techniques of UML.

Obviously, there is a huge abstraction gap that has to be bridged when designing a new domain-specific modeling language between the visual specification of abstract syntax (expressed in UML) and dynamic semantics (in graph transformation rules), and the textual description of static semantic constraints (written in OCL). Since visual specifications drastically increase the "legibility" of a model, we focus on the visual definition of static semantics.

In the paper, we propose *a visual constraint language to express first-order structural constraints by graph patterns* in analogy with the pattern matching concepts of graph transformation. We demonstrate how typical structural OCL constraints can be expressed by patterns, and how such visual constraints can be checked automatically using the graph pattern matching engine of the model transformation system VIATRA [6][7].

References

[1] T. Clark, A. Evans, and S. Kent. The Metamodelling Language Calculus: Foundation semantics for UML. In H. Hussmann, editor, Proc. Fundamental Approaches to Software Engineering, FASE 2001 Genova, Italy, volume 2029 of LNCS, pages 17–31. Springer, 2001.

[2] G. Engels, R. Heckel, and J. M. Küster. Rule-based specification of behavioral consistency based on the UML meta-model. In M. Gogolla and C. Kobryn, editors, UML 2001: The Unified Modeling Language. Modeling Languages, Concepts and Tools, volume 2185 of LNCS, pages 272–286. Springer, 2001.

[3] A. Ledeczi., M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, and P. Volgyesi. The Generic Modeling Environment. In Proc. Workshop on Intelligent Signal Processing, 2001.

[4] Object Management Group. Object Constraint Language Specification Version 1.3, June 1999. `http://www.rational.com/uml`

[5] J. Rumbaugh, I. Jacobson, and G. Booch. The Unified Modeling Language Reference Manual. Addison-Wesley, 1999.

[6] D. Varró, G. Varró, and A. Pataricza. Designing the automatic transformation of visual languages. Science of Computer Programming. In print.

[7] All papers of the author are available from `http://www.inf.mit.bme.hu/~varro`