An algebraic semantics for objects in a rule-based language

Hubert Dubois, Hélène Kirchner

An algebraic semantics for objects in a rule-based language

Hubert Dubois and Hélène Kirchner
LORIA-Université Nancy 2 & LORIA-CNRS
BP 239
54506 Vandœuvre-lès-Nancy Cedex, France
{Hubert.Dubois|Helene.Kirchner}@loria.fr

1 Introduction

Object-oriented programming and rule-based programming are two programming paradigms that have been developed in last decades. On one hand, object oriented programming languages like Ada [Ros92], Smalltalk-80 [GR83] or Java [AG96] are well-known and largely used; but often lack of semantical basis. On the other hand, rule-based systems, initially used in the artificial intelligence community, have gained interest with the development of efficient compilers like the ELAN system [BCD+00,KM01].

Recently, the combination of object-based languages and rule-based languages has been proved to be quite relevant to formalize and solve industrial problems. Among many other languages, let us mention three of them: CLAIRE [CL96], Oz [HSW95] and Maude [CDE+00,DELM00].

Compared to these existing languages, extending ELAN with objects [DK02] has several advantages. First of all, the fact that ELAN provides a strategy language and strategy constructors to define control on rules, appears as essential for many applications [DK00]. Second, the main features that characterize object languages (definition of classes composed of attributes and methods; simple inheritance; method call on objects) can be defined in a reflective way, since the extension is defined in ELAN itself [DK02].

Finally, our last, and surely more important, point is that the operational semantics of the object extension is based on the object ρ-calculus defined in [CKL01]. This is an object extension of the ρ-calculus [CK99] (also called Rewriting Calculus) that encompasses both λ-calculus and term rewriting. In this calculus, terms, rewrite rules and application of a rule on a term can be represented at the same level. The ρ-calculus already provides an operational semantics for ELAN. In this work, we show how to map the theory of objects into the ρ-calculus.

In the object-oriented extension of ELAN, presented here, special modules called OModules (for Object Modules) are available to define classes and objects of that class. An object base representing the current state of the objects can be modified by application of special rules called object rules. These rules are defined to delete, modify or add objects in the object base.

2 An algebraic encoding of objects

In order to implement the definition of classes and objects embedded in OModules in a first-order algebraic language such as ELAN, we choose an approach where objects and classes are represented uniformly by objects. This approach is also followed in object-based languages like Smalltalk-80 where the unique entity is the object and where a class is itself an object.

For several reasons, the chosen encoding approach is the following: each object is composed of pairs *(attribute, value)* and of method's names referencing methods which the object has access to. The objects are represented by terms; methods are represented as functions defined by rewrite rules.

The algebraic encoding is composed of operators associated to the object representation, to the object manipulation (creation, method calls, modification), to each attribute definition and to each method declaration.

This definition of operators is completed by a definition of rewrite rules used in the object manipulation and in the method definition. Indeed, to each method body corresponds a rewrite rule right-hand side, with possibly local assignments and boolean tests. Rules used to evaluate objects compose a rewrite system $\mathcal{R}$. This system is based on the representation of objects and inspired from the object ρ-calculus. We proved in [Dub01] that $\mathcal{R}$ is confluent and terminating.

The $\mathcal{R}$ rewrite system is extended with the set of rules that correspond to each user defined method. When the user defines his own methods, it is his concern to check that method definitions generate, together with $\mathcal{R}$, a set $\mathcal{R}'$ of confluent and terminating rewrite rules.

3 An operational semantics based on the ρ-calculus

In the algebraic theory of objects, an object is represented as a term of sort `Object`. By considering the system $\mathcal{R}'$, each term t of sort `Object` has a normal form denoted by $NF_{\mathcal{R}'}(t)$, or $NF(t)$ for short.

We prove that each reduction of a term t of sort `Object` in the algebraic theory of objects to its normal form $NF(t)$ corresponds to a reduction in the ρ-calculus from a ρ-term t_0 to another one t'_0, where t_0 and t'_0 are the respective translations of t and $NF(t)$ considering a translation τ that we have defined.

4 Conclusion

Following the approach presented here, objects, rewriting rules and strategies can be integrated in a same formalism. This consists in adding OModules where classes with attributes and methods are declared together with the standard ELAN modules. In OModules, rules that apply on objects can be defined.

Furthermore, thanks to the unique algebraic representation that we have defined for both objects and classes, we have proved that these objects can also be represented in the ρ-calculus. This naturally provides an operational semantics for the object extension of ELAN based on the ρ-calculus, which already gives an operational semantics to ELAN.

Interests of the "objects+rules+strategies" paradigm are illustrated by several examples such that a multi-elevator controller and a print controller, described in [Dub01].

References

[AG96] K. Arnold and J. Gosling. *The Java programming language.* Addison Wesley, 1996.

[BCD+00] P. Borovanský, H. Cirstea, H. Dubois, C. Kirchner, H. Kirchner, P.-E. Moreau, C. Ringeissen, and M. Vittek. ELAN *V 3.4 User Manual.* LORIA, Nancy (France), fourth edition, January 2000.

[CDE+00] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martì-Oliet, J. Meseguer, and J.F. Quesada. Towards Maude 2.0. In K. Futatsugi, editor, *WRLA2000, the 3rd International Workshop on Rewriting Logic and its Applications, September 2000, Kanazawa, Japon.* Electronic Notes in Theoretical Computer Science, 2000.

[CK99] H. Cirstea and C. Kirchner. Combining higher-order and first-order computation using ρ-calculus: Towards a semantics of ELAN. In D. Gabbay and M. de Rijke, editors, *Frontiers of Combining Systems 2*, Research Studies, ISBN 0863802524, pages 95–120. Wiley, 1999.

[CKL01] H. Cirstea, C. Kirchner, and L. Liquori. Matching Power. In A. Middeldorp, editor, *Proceedings 12th Conference on Rewriting Techniques and Applications, RTA 2001, Utrecht, The Netherlands*, volume 2051 of *Lecture Notes in Computer Science*, pages 77–92. Springer-Verlag, 2001.

[CL96] Y. Caseau and F. Laburthe. CLAIRE: Combining objects and rules for problem solving. In *Proceedings of the JICSLP'96 workshop on multi-paradigm logic programming,TU Berlin, Germany*, 1996.

[DELM00] F. Durán, S. Eker, P. Lincoln, and J. Meseguer. Principles of Mobile Maude. In D. Kotz and F. Mattern, editors, *Procs. ASA/MA 2000*, volume 1882 of *Lecture Notes in Computer Science*, pages 73–85. Springer-Verlag, 2000.

[DK00] H. Dubois and H. Kirchner. Objects, rules and strategies in ELAN. In *Proceedings of the second AMAST workshop on Algebraic Methods in Language Processing, Iowa City, Iowa, USA*, May 2000.

[DK02] H. Dubois and H. Kirchner. Object Programming in a Rule Based Language with Strategies. In *Proceedings of the ECOOP 2002 Workshop on Multiparadigm Programming with OO Languages MPOOL'02, Malaga, Spain*, June 2002.

[Dub01] H. Dubois. *Systèmes de Règles de Production et Calcul de Réécriture.* PhD thesis, Université Henri Poincaré - Nancy 1, 2001.

[GR83] A. Goldberg and D. Robson. *Smalltalk-80: The Language and Its Implementation.* Addison-Wesley, 1983.

[HSW95] M. Henz, G. Smolka, and J. Wurtz. Object-oriented concurrent constraint programming in oz. In V. Saraswat and P. van Hentenryck, editors, *Principles and Practice of Constraint Programming.*, pages 27–48. MIT Press, 1995.

[KM01] H. Kirchner and P.-E. Moreau. Promoting Rewriting to a Programming Language: A Compiler for Non-Deterministic Rewrite Programs in Associative-Commutative Theories. *Journal of Functional Programming*, 11(2):207–251, 2001.

[Ros92] J.-P. Rosen. What Orientation Should Ada Objects Take? *Communications of the ACM*, 35(11):71–76, 1992.