An Algebraic Semantics of UML Supporting its Multiview Approach

Extended Abstract

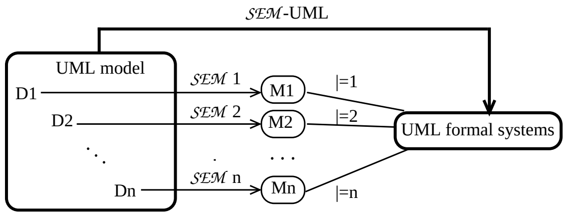
G. Reggio – M. Cerioli – E. Astesiano
DISI Università di Genova - Italy

We aim at using algebraic techniques, and in particular an extension, Casl-Ltl [2], of the Casl basic language [4], in order to produce a formal semantics of the UML, the OMG standard object-oriented notation for specifying, visualizing, constructing, and documenting software systems [5]. Contrary to most cases, this task is far from trivial. Indeed, the UML notation is complex, including a lot of heterogeneous notations for different aspects of a system, possibly described at different points in the development process. Moreover, its informal description is incomplete and ambiguous, not only because it uses the natural language, but also because the UML has the so called *semantics variation points*, that are constructs having a *list* of possible semantics, instead of just one.

A UML model consists of a bunch of diagrams of different kinds, expressing properties on different aspects of a system[1]. Thus a UML model plays the role of a specification, but in a more pragmatic context.

Another analogy that we can establish between UML models and specifications is the fact that the meaning of each diagram (kind) can be given in isolation, as well as the semantics of each axiom, and its effect on the description of the overall system is to rule out some elements from the universe of all possible systems (semantic models). Indeed, both in the case of a UML model and of a collection of axioms, each individual part (one diagram or one axiom) describes a point of view of the overall system.

Therefore, our understanding of the optimal form of a semantics for the UML is illustrated in the picture below.



We have a box representing a UML model, collecting some diagrams of different kinds, and its overall semantics, represented by the arrow labeled by $\mathcal{SEM}$-UML, is a class of UML formal systems. But, each diagram in the model has its own semantics (denoted by the indexed $\mathcal{SEM}$), that is a class of appropriate structures, as well, and these structures are imposing constraints on the overall UML formal systems, represented by lines labeled by $\models$. A sort of commutativity on the diagram has to hold, that is the overall semantics must be the class of UML formal systems satisfying all the constraints imposed by the individual semantics.

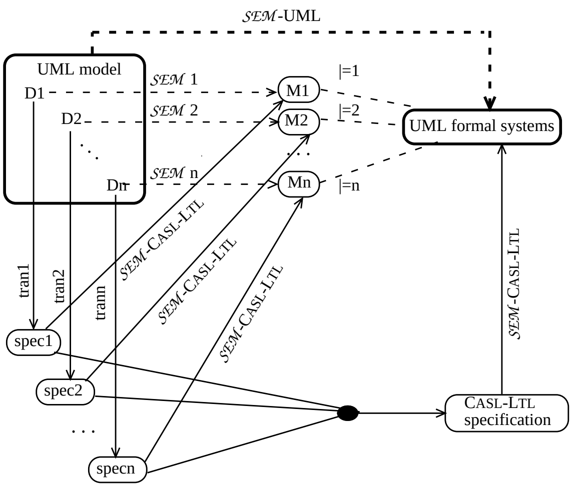
Several attempts at formalizing the UML are currently under development (we omit the references for lack of space), but most of them are taking into account only a part of the UML, with no provision for an integration of the individual diagram semantics toward a formal semantics of the overall UML model. The only exception known to us is the attempt at describing the semantics of the UML within the UML itself (the *meta-model* approach[2]); but even in this case it is difficult to recognize the nature of the semantics of the individual diagrams, as the semantics is given as a sequence of translations into more and more restricted core languages.

Our approach, accordingly with the previous discussion, is an attempt at formalizing UML models as a whole, while simultaneously giving also a formalization of each kind of diagram in an integrated way. In the picture below, we graphi-

[1] In the following we will call UML-*systems* the "real world" systems modeled by using the UML (some instances are information systems, software systems, business organizations) and UML *formal systems* their formal counterparts.

[2] See the site `http://www.cs.york.ac.uk/puml/`

cally summarize our proposal.



From a technical viewpoint, we proceed in two steps: first, we determine the needed semantic structures (the M_i and the UML formal systems in the picture above) through an analysis of the the UML standard [5], and formally describe them as algebraic structures. Then, we translate the diagrams into Casl-Ltl specifications (represented by the downward arrows), whose formal semantics gives, by composition, the semantics of each diagram in the UML model (represented by the dotted horizontal arrows).

Moreover, in the lower part of the diagram, the Casl-Ltl specifications representing the individual diagrams are combined (in a non-trivial way) into an overall specification, whose semantics is (has to be) compatible with the constraints imposed by the individual diagrams and provides a semantics for the overall UML model. This combination is graphically represented by a bullet.

We are currently working on filling the above schema, providing the semantic structures and the translations of the various diagrams into Casl-Ltl. This activity is performed as part of the `CoFI` [3] initiative, within the `CoFI`-reactive task group.

The kinds of diagram considered so far are

- the class diagrams, analyzed and translated into Casl (that is a subset of Casl-Ltl) in [1];
- the statechart diagrams, analyzed and translated into Casl-Ltl in [3];
- the sequence diagrams, currently under development.

Some other kinds of diagrams that we have partly analyzed, and that we conjecture can be added to our schema without major problems, are

- the collaboration diagrams, as they are rather similar to the sequence diagram;
- the activity diagrams, as they are a specialization of the statechart diagrams.

Moreover, we still have to take into account the deployment diagrams, though we do not foresee particular problems for their formalization within our framework, while we are doubtful about the possibility of giving a formal semantics to the use case diagrams, because they are, roughly speaking, natural language descriptions.

We translate diagram annotations as well, currently using the OCL constraints, but we are in some sense parametric w.r.t. such annotations, so that we could easily substitute any other constraint language for OCL.

The mechanisms for self-extension provided by the UML, like stereotypes, are still to be taken into account.

[3] See the site `http://www.brics.dk/Projects/CoFI`.

References

[1] H. Hussmann, M. Cerioli, and H. Baumeister. From UML to Casl (Static Part). Technical Report DISI-TR-00-08, DISI – Università di Genova, Italy, 2000. In preparation.

[2] G. Reggio, E. Astesiano, and C. Choppy. Casl-Ltl : A Casl Extension for Dynamic Reactive Systems – Summary. Technical Report DISI-TR-99-34, DISI – Università di Genova, Italy, 1999. `ReggioEtAll99a.ps` in `ftp://ftp.disi.unige.it/person/ReggioG/`.

[3] G. Reggio, E. Astesiano, C. Choppy, and H. Hussmann. Analysing UML Active Classes and Associated State Machines – A Lightweight Formal Approach. In *Proc. FASE 2000 - Fundamental Approaches to Software Engineering*, LNCS. Springer Verlag, Berlin, 2000. To appear.

[4] The CoFI Task Group on Language Design. Formal Methods '99 - CASL, The Common Algebraic Specification Language - Summary. Available on compact disc published by Springer-Verlag, 1999.

[5] UML Revision Task Force. *OMG UML Specification*, 1999. Available at `http://uml.shl.com`.