The New ICom

– Demo Paper –

Pablo Fillottrani, Enrico Franconi, Sergio Tessaris
Free University of Bozen-Bolzano, Italy
lastname@inf.unibz.it

ICom is an advanced CASE tool, which allows the user to design multiple UML class diagrams with inter- and intra-model constraints. Complete logical reasoning is employed by the tool to verify the specification, infer implicit facts, devise stricter constraints, and manifest any inconsistency.

For the ontology creation and maintenance tasks, ICom interface supports ontology engineers in engineering ontologies that meets clear and measurable quality criteria. Indeed, recently we observe the development of large numbers of ontologies. These ontologies have, however, usually been developed in an ad hoc manner by domain experts, often with only a limited understanding of the semantics of ontology languages. The result is that many ontologies are of low quality—they make poor use of the languages in which they are written and do not accurately capture the author's rich knowledge of the domain. This problem becomes even more acute as ontologies are maintained and extended over time, often by multiple authors. Poor quality ontologies usually require localised "tuning" in order to achieve the desired results within applications. This leads to further degradation in their overall quality, increases the brittleness of the applications that use them, and makes interoperability and reuse difficult or impossible. To overcome these problems tools are needed which support the design and the development of the basic infrastructure for building, merging, and maintaining ontologies.

The Ontology Editor works on *projects*, which may contain one or more UML class diagram. The diagrams are referred as *models*. Multiple projects can be opened at the same time, but objects cannot be moved across them. Only one project is visible at a time and the editing of each project is independent. The user can switch between different projects using the tabs at the bottom of the project area.

Figure 1 shows the main window of the Ontology Editor editing a single model. Classes are represented by boxes and n-ary associations by diamonds. Associations have so-called *association classes* specifying their name and attributes. Isa relationships are represented as arrows with a disc in the middle (e.g. see `Enterprise` and `Business_Organization`).

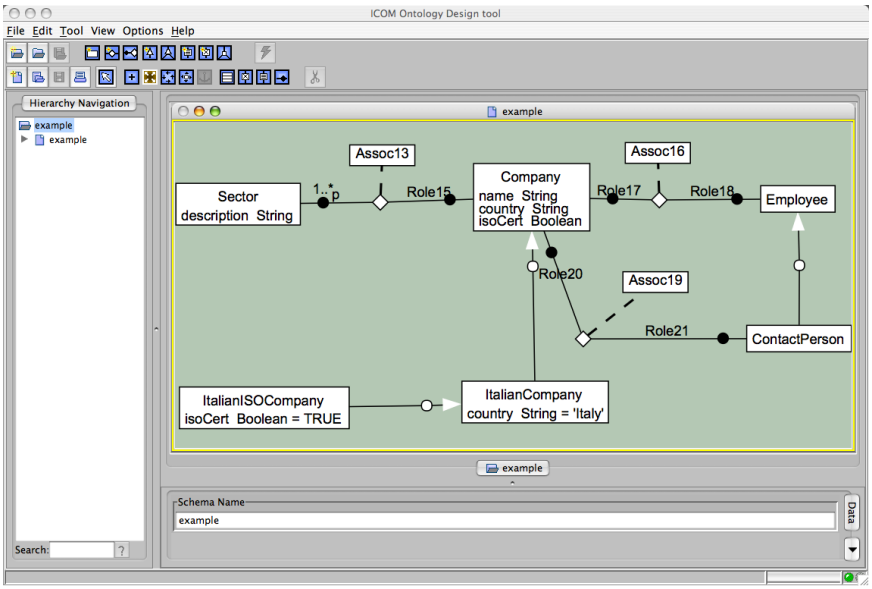

Figure 1: Ontology Editor main window

Although the Ontology Editor can be used as a standalone modelling tool, exploiting its full capabilities require the coupling of the system with any DIG enabled Description Logic reasoner [3, 1]. The leverage of automated reasoning to support the domain modelling is enabled by a precise semantic definition of all the elements of the class diagrams. The diagrams and inter-model constraints are internally translated into a class-based logic formalism. The same underlying logic enables the use of a *view definition language* to specify additional contraints, not captured at the diagram level. The next section introduces this logic-based modelling language.

After the verification process, the system provides the user with a visual account of the deductions by modifying the appearance of the model diagrams in the project. The user can discard the deductions and the entire project will be returned to its original state (and any information about unsatisfiability will be discarded). Alternatively, the equivalence, subsumption association, and role cardinality deductions can be added permanently to the project by committing them.

ICom is a fairly mature project, its first release has been published in 2001 (see [4, 2]). This paper presents the new improved version of the tool. Although the theoretical underpinning is the same, ICom in its current version underwent major changes in several crucial aspects. First of all, the diagrammatic representation is now based on UML rather than Entity Relationship diagrams. The graphical interface has been completely rewritten to improve the usability and intuitiveness of the tool. Interoperability with other tools is a crucial aspect; so, import and export modules have been developed for XMI 2.x [5] and Description Logics based ontology languages.

References

[1] Sean Bechhofer. The dig description logic interface: Dig/1.1. Technical report, University of Manchester, 2003.

[2] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. Description logic framework for information integration. In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'98)*, pages 2–13, 1998.

[3] Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. Description logics for conceptual data modeling. In Jan Chomicki and Günter Saake, editors, *Logics for Databases and Information Systems*. Kluwer, 1998.

[4] Enrico Franconi and Gary Ng. The icom tool for intelligent conceptual modelling. In *7th Intl. Workshop on Knowledge Representation meets Databases (KRDB'00)*, 2000.

[5] OMG, 2005. `http://www.omg.org/technology/documents/formal/xmi.htm`.