The Heterogeneous Tool Set — some recent developments and highlights

Till Mossakowski¹ and Mihai Codescu²

¹ Institute of Intelligent Cooperating Systems, Otto-von-Guericke-University Magdeburg, Germany
² Institute of Mathematics "Simion Stoilow" of the Romanian Academy, Research Group of the project PED-0494, Bucharest, Romania

The is a large body of literature on modular structured and architectural specification, refinement, colimits, proof calculi etc. [3,13,4,1,2,12,14], which develops these notions (at the level of definitions, theorems and also language design) independently of the underlying logic. The logic at hand needs to be formalised as an institution. The notion of institution is a powerful abstraction, introduced by Goguen and Burstall [7]. The satisfaction condition imposed by the notion of institution is a rather mild requirement met by most logical systems.

The Heterogeneous Tool Set (Hets [11]) is a parsing, analysis and proof management tool for heterogeneous specifications, available at hets.eu. It provides a software interface for plugging-in institutions, and provides tool support for modular specification and refinement in an institution-independent way, thus complementing the theoretical institution-independent approach in the literature with a practical tool. Moreover, based on a graph of institutions and institution translations (formalised as comorphisms [8]), Hets supports heterogeneous multi-logic specifications as developed in [15,6,9].

In this work, we update the ten-year old description of Hets in [11] with recent highlights and developments.

Distributed Ontology, Model and Specification Language (DOL) Originally, Hets has used the Common Algebraic Specification Language CASL [2,12] as input language. CASL provides institution-independent structuring constructs, and can be used with any institution as base logic. Based on CASL, we have developed the Distributed Ontology, Model and Specification Language (DOL) [10], see also omg.org/spec/DOL and dol.omg.org. DOL extends CASL several ways: several notions of reduction (including ontology module extraction and uniform interpolation), minimization (following McCarthy’s circumscription), alignments (a kind of relational theory morphisms), as well as networks of specifications and their colimits (the latter is called combination in DOL).

Internet and Linked Data Compatibility While in CASL, identifiers are essentially strings (and internet compatibility is only given indirectly via downloads of libraries), Hets now supports arbitrary URLs/IRIs. Specifications and specification libraries can directly accessed via their URL.

Architectural Refinement Simple refinements are specification morphisms [14] (logically: interpretations of theories). They are available in CASL [2,12] as so-called views. In [5], we have developed a more complex refinement language, which allows the user to formalize complete developments as refinement trees. The newly-introduced language supports branching (using CASL architectural specifications), going thus beyond what can be expressed using views, and further
refinement of leaves of a refinement tree, as well as composition of refinements. In Hets, refinements written in this language can be parsed and statically analyzed, and their correctness and consistency can be checked using the calculi introduced in [5]. Moreover, the refinement trees can be visualised and inspected from within Hets.

**Database Support** Traditionally, Hets analysis specification libraries in a local session that cannot be made permanent. Recently, database (sqlite or Postgresql) has been added. Thus, it is possible to write all information of specification libraries, as well as proofs that have been carried out, into a database, and use this in later sessions, as well as for other tools like Ontohub.org.

**References**