Ontology development with CASL and HETS

Klaus Lüttich

LOA, Trento, 05.04.2005

SFB/TR 8 SPATIAL COGNITION

Overview

- CASL– background and introduction
- Ontology examples in CASL
- Type checking and static analysis

CASL– the Common Algebraic Specification Language

- de facto standard for specification of functional requirements
- developed by the "Common Framework Initiative" (an open international collaboration)
- approved by IFIP WG 1.3 "Foundations of Systems Specifications"
- CASL User Manual (LNCS 2900) and Reference Manual (LNCS 2960) just have appeared

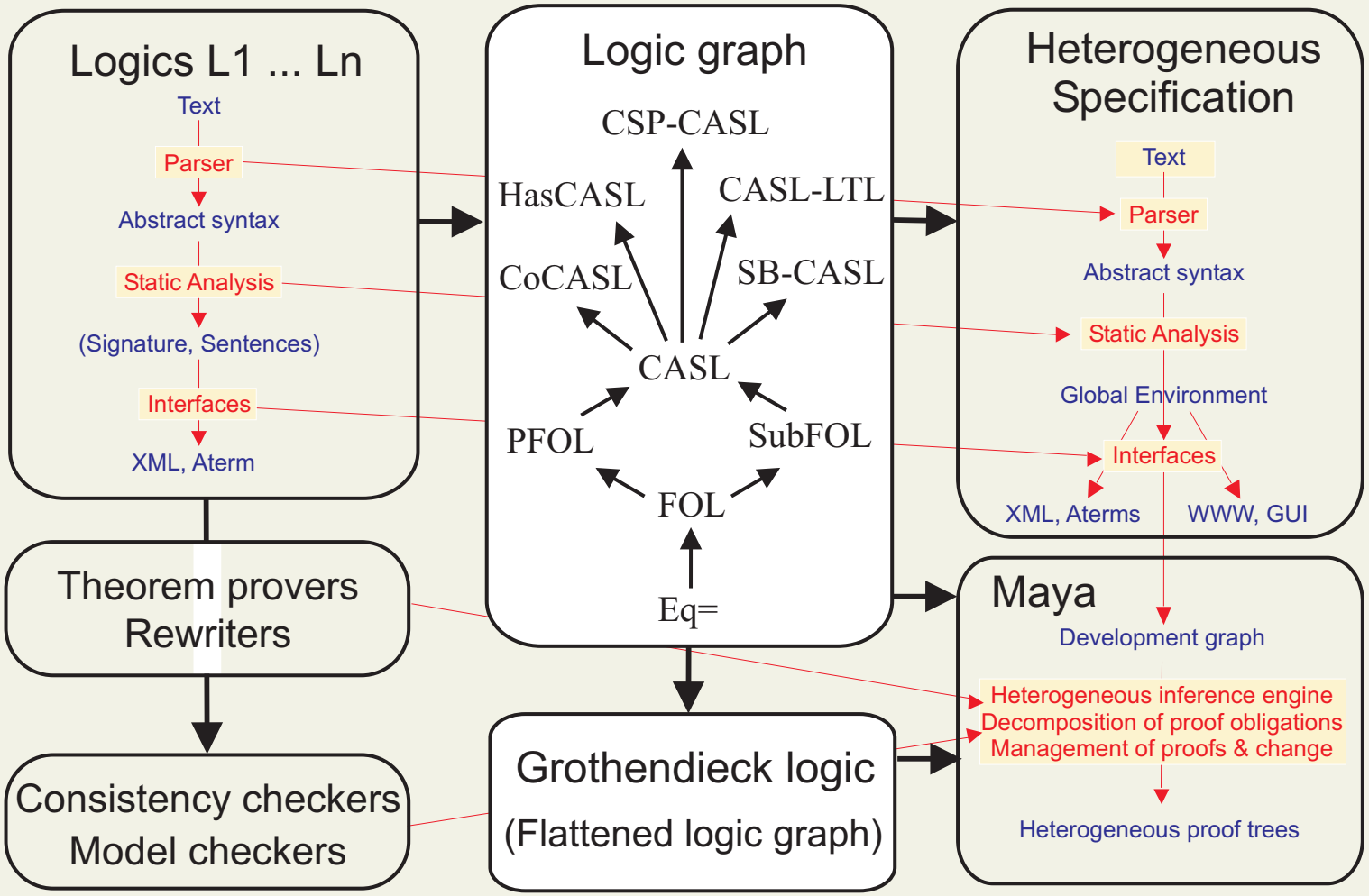
Central features of CASL

- supports both specification of abstract requirements
  (loose specification)
  as well as description of specific models
  (design specification)
- In-the-small:
  first-order logic, subsorting, partiality, induction, datatypes
- In-the-large:
  structuring of specifications and models,
  libraries distributed over the Internet

CASL in contrast to untyped FOL

CASL	**untyped FOL**
sorts	unary predicates
sub sorting	implication of unary predicates
just different sorts	disjointness of unary predicates
typed predicates	predicate implies unary predicates
axioms quantified over sorts	disjunction/conjunction of unary predicates implies axiom
typed functions	predicates with unique existential quantification

Architecture of the heterogeneous tool set Hets



Taxonomy with subsorting

%% subsumptions

sorts $A, B < Top$;

$AA, AB < A$;

$BA, BB < B$

Taxonomy with subsorting

%% subsumptions

sorts $A, B < Top$;

$AA, AB < A$;

$BA, BB < B$

%% disjointness of concepts

free type $Top ::= sorts\ A, B$

Taxonomy with subsorting

%% subsumptions

sorts $A, B < Top;$

$AA, AB < A;$

$BA, BB < B$

%% disjointness of concepts

free type $Top ::= sorts\ A, B$

%% unary predicates

preds $Un1 : Top;$

$Un2 : BA$

Show ASCII Analyse with Hets

(Typed) Roles and Predicates

preds $P1 : AB \times BA;$

$P2 : Top \times B \times BB$

$P3(x, y: B; z: A) \Leftrightarrow P1(z, y) \wedge P2(z, y, x)$

(Typed) Roles and Predicates

preds $P1 : AB \times BA;$

$P2 : Top \times B \times BB$

$P3(x, y: B; z: A) \Leftrightarrow \sout{P1(z, y) \wedge P2(z, y, x)}$

$P1(z \text{ as } AB, y \text{ as } BA) \wedge P2(z, y, x \text{ as } BB)$

Axioms on sorts (concepts / classes)

Sorts can be defined with a Formula:

sort $C = x\colon\ Top\ \bullet\ \phi(x)$

Axioms on sorts (concepts / classes)

Sorts can be defined with a Formula:

sort $C = x\colon\ Top\ \bullet\ \phi(x)$

this equivalent to:

$C < Top$ and

Axioms on sorts (concepts / classes)

Sorts can be defined with a Formula:

sort $C = x\colon\ Top\ \bullet\ \phi(x)$

this equivalent to:

$C < Top$ and

$\forall x\colon\ Top\ \bullet\ \mathsf{x} \in C \Leftrightarrow \phi(x)$

Overloading and Subsorting

- embeddings into supersorts work implicitly
- a term that is well typed for a supersort must be projected to the subsort with as
- you can have formulas depending on the membership in a sort like this:
  $\forall x\colon\ Top\ \bullet\ \mathsf{x} \in C \Rightarrow \phi(x)$

If you want a predicate defined on subsorts of a sort you better introduce the predicate like this:

spec $\textsc{Taxo} =$

sorts A, B

free type $Top ::= sorts\ A, B$

end

spec PARTHOOD =

TAXO

then

preds $P : A \times A$;

$P : B \times B$

$\forall\ x{:}\ Top$

• $(P\ (x, x)) \Rightarrow (\phi\ (x))$ **%%** is ill typed

$\forall\ x{:}\ A$

• $(P\ (x, x)) \Rightarrow (\phi\ (x))$ **%%** is correct

end

Parametrised / Generic Specifications

Rule of Thumb: Keep parameters as small as possible

- a parameter is a basic specification or a named specification
- instantiation of such a specification works by providing all the symbols (and axioms) needed to fit the specified parameter
- an instantiation can add symbols to the signature

Examples of generic specification

library GenericExample2

spec Primitives =

sorts PD, PED, S, SL, T, TL

free type $PT ::= sorts\ PD, PED, S, SL, T, TL$

Examples of generic specification

library GenericExample2

spec Primitives =

sorts PD, PED, S, SL, T, TL

free type $PT ::= sorts\ PD, PED, S, SL, T, TL$

end

spec GenParthood [**sort** s] =

pred $P : s \times s$

$\forall\ x, y, z : s$

- $P(x, x)$ %(Ad11)%
- $P(x, y) \wedge P(y, x) \Rightarrow x = y$ %(Ad12)%
- $P(x, y) \wedge P(y, z) \Rightarrow P(x, z)$ %(Ad13)%

end

spec GenMereology [**sort** s] =
GenParthood [**sort** s]
then
%% some definitions and axioms
end

spec GenMereology [**sort** s] =
GenParthood [**sort** s]
then
%% some definitions and axioms
end
spec Mereology =
Primitives
then GenMereology [**sort** T]
then GenMereology [**sort** S]
then GenMereology [**sort** PD]
end

spec MEREOLOGY_ALTERNATIVE =
GENMEREOLOGY [PRIMITIVES **fit** $s \mapsto T$]
then GENMEREOLOGY [PRIMITIVES **fit** $s \mapsto S$]
then GENMEREOLOGY [PRIMITIVES **fit** $s \mapsto PD$]
end