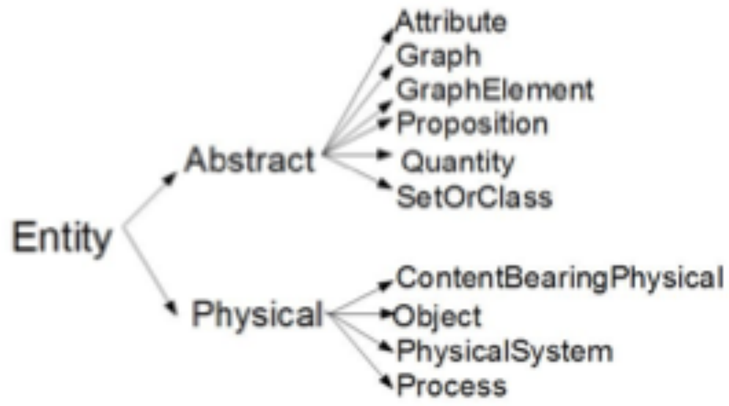
Typeful Ontologies with Direct Multilingual Verbalization

The Motivation:
+ **type system -** more robust and consistent representation
+ **direct natural language generation** - better readability
+ **parallel multilingual verbalization** - accessibility



The Use Case:
+ Suggested Upper-Merged Ontology(SUMO)
- largest open-source ontology
- over 10,000 concepts and relations
- over 5,000 axioms

The Experiment - Translation:
+ dependently typed framework - enforces semantic consistency
+ clear delimitation between classes and instances
+ examples:
- type declarations:

```
(instance Pi PositiveRealNumber) →
                    fun Pi  : Ind PositiveRealNumber
```

- axioms:

```
(=>
(instance ?P Wading)
 (exists (?W)
  (and
   (instance ?W BodyOfWater)
   (located ?P ?W))))     →
 forall Wading(\P → exists BodyOfWater (\W → located
(var Wading Physical ? P) (var BodyOfWater Object ? W)))
```

The Tools:
+Grammatical Framework(GF)
- a grammar formalism based on type theory
- a functional programming language with strong static typing and support for dependent types
- mainly used for multilingual grammar applications, natural language generation and dialogue systems
- based on the idea of having an abstract syntax(semantic interlingua) and concrete syntaxes(natural language front-ends)
- ideal for writing controlled languages - built-in parser
- multilingual translation between any language pair

The Experiment:
+ representing SUMO in GF => **SUMO-GF**
+ verbalizing SUMO(declarations and axioms) in 3 languages: English, French and Romanian
+ further translation of SUMO-GF into TPTP and performing automated reasoning tasks on it
+ comparing the trade-off between type-safety and expressivity of SUMO-GF/SUMO
+ discovered ca 8% type errors in the original SUMO axioms

The Experiment - Natural Language Generation
+ automatic for concept declarations, semi-automatic for relations
+ compositional - free verbalization for axioms
+ syntactically correct even for morphologically rich languages
+ examples:
-for every unique list LIST, every positive integer NUMBER2 and every positive integer NUMBER1 we have that if the element with number NUMBER1 in LIST is equal to the element with number NUMBER2 in LIST, then NUMBER1 is equal to NUMBER2

-pour chaque maison A il existe une maison B telle que B est plus petite que A

Ramona Enache
ramona.enache@chalmers.se
Department of Computer Science and Engineering
Gothenburg University and Chalmers University of Technology

Funded by the European Project MOLTO(Multilingual Online Translation) 2010-2013, FP7-ICT-247914
Based on joint work with Krasimir Angelov