Application of OWL 1.1 to Systems Engineering

01 April 2008

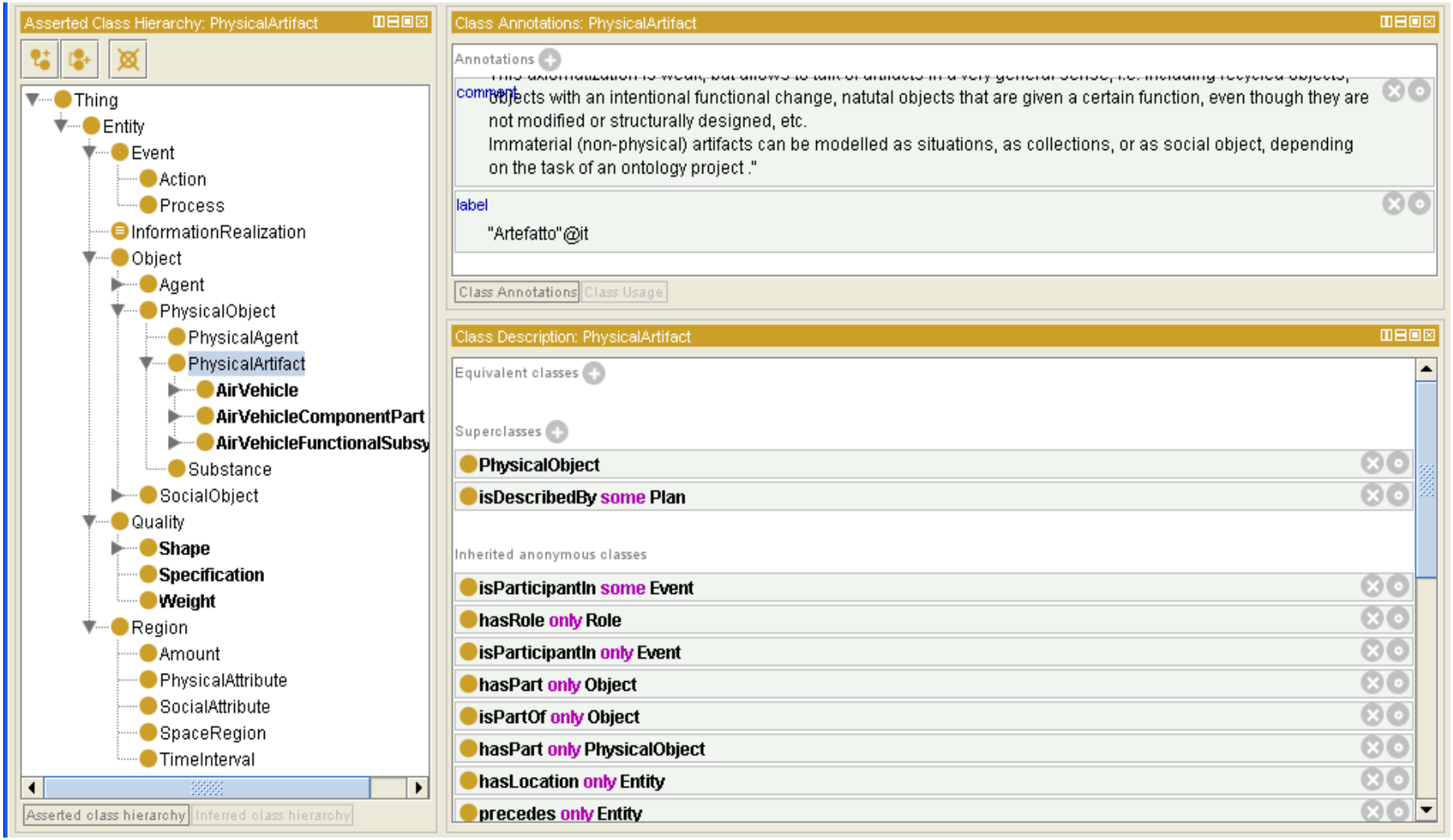
Henson Graves

Ian Horrocks

Use of Knowledge Representation and Reasoning in Product Development

- **Current Systems Engineering languages, standards, and tools are**
  - Restricted in (certain aspects of) their expressiveness and do not provide formal semantics
  - Yet many activities involve (some form of) reasoning, e.g., requirements verification
- **Long history of attempts to use formal methods for engineering**
  - Too hard to use, don't scale
- **Can OWL 1.1 provide a semantic integration framework?**
  - For engineering domain, with ontology capturing meaning independent of interpretation by subject matter experts?
  - So automated reasoning can be used to check design properties such as consistency and conformance with specification?
  - Not to replace SysML, UML, and engineering tools, but incorporate them into an integrated framework

We Are Developing an Air System Ontology in Protégé 4.0 Using the Fact++ Reasoner and DOLCE Ultra Lite (DUL)



…use of a foundation ontology saves a lot of time

Achievements

- **Used class constructors to define (requirements) classes of physical objects with structural and measurable properties**
- **Resulting ontology has about 300 classes**
- **Verified that requirements class is consistent**
- **Defined (design) classes characterized by generic instances**
- **Gone outside the logic to database that represents results of measurement analysis & simulation to conclude additional properties of design instance**
- **Verified that resulting extended generic design instance still satisfies requirements**

Benefits and Shortcomings of OWL 1.1

- **OWL 1.1 with DUL classes works well for representation of product structure and (static) properties**
  - Used *partOf* relations, *Quality*, with values in *Region*
- **New OWL 1.1 features crucial**
  - Complex role inclusions
    - Transfer of properties across part-whole relations
  - Extended support for datatypes
    - Numerous design constraints relate to concrete values such as weight, speed, temperature, distance, …
    - Complex datatypes, representing, e.g., shapes or performance measures
  - Extended annotation
    - E.g., for provenance of information

Benefits and Shortcomings of OWL 1.1

- **Some requirements not (easily) expressible OWL DL**
  - Extended reasoning with datatypes and values, e.g., agregation
    - Weight of product is sum of weights of components
  - Behavioral and other dynamic requirements
    - I.e., statements involving state change
  - Interfacing and integrating with other systems
    - Storage and representation systems such as DBs
    - Testing and measurement systems such as simulators

Lessons Learned

- **OWL 1.1 is not a replacement for systems engineering language and tools**
  - But is a good candidate for semantic integration
- **Potential for reasoning in OWL for systems engineering is great**
  - Achieved reasoning experiments for requirements consistency, and design satisfaction of requirements
- **Further language and tool development is needed**
  - Ontology management
  - Explanation and annotation
  - Modularization
  - Architecture to integrate with other computational and reasoning systems