Foundations of Semantic Web Technologies

Ontology Engineering

Sebastian Rudolph
Ontology Engineering

Current research

Trust

Proof

Unifying Logic

Query: SPARQL

ontology: OWL

Rules: RIF

RDF-S

Data interchange: RDF

XML

URI

Unicode

Now standardized

User Interface & applications

Crypto
Why Ontology Engineering?

- ontology languages and reasoners provide the technical infrastructure for using semantic technologies
- however: domain specific knowledge specifications (=ontologies) have to come from somewhere
- questions:
  - how to create (good, useful) ontologies?
  - how to judge ontology quality?
Why Ontology Engineering?

- many aspects of these questions are similar to problems of software engineering
- in both cases, complex artefacts are created (often in a collaborative way) that should be
  - correct
  - functional
  - comprehensible
  - reusable
  - etc.
Requirement Analysis

...always a good idea before one starts constructing an ontology:

- is a semantic representation needed at all? (alternative: database solution)
- if so, should it be a representation based on formal logic
  (alternative: textual representation, in particular if the purpose is human-to-human information exchange)
  - contra: cost of setting it up, established practice
  - pro: flexible usage/exchange
  - pro: reasoning allows for managing implicit knowledge
Requirement Analysis

- tool support
  - does the choice of a certain approach create dependencies to a specific tool?
  - under what license models are the necessary tools available?
  - how stable/mature is the software?
  - what support is offered by the vendor?
  - are the available tools sufficiently interoperable?
**Requirement Analysis**

- functional aspects
  - what domain is to be modelled/what aspects of that domain are to be represented?
  - what is the level of granularity of the information to be specified?
  - what tasks are supposed to be performed using the ontology?
    - browsing a body of knowledge
    - search for information
    - query processing
    - automated inferencing
- what inferences are wanted
Ontology Creation

- there are many possible sources of knowledge that can be utilized:
  - humans
  - texts
  - webpages
  - databases

- these sources are different in terms of (i) how explicit and (ii) how structured the knowledge is
Humans

- domain experts = humans who have the required knowledge
- ...are normally not logicians and don‘t know ontology languages
- thus, often knowledge engineers are needed as „mediators“
- these must be knowledgeable in formal logic and have good communication skills
Texts

- accessible in a very direct way: digital textual resources, still: text ≠ logic
- possibility: deployment of automated methods for extracting knowledge from text (Ontology Learning)
- different approaches:
  - pattern-based search for predefined relations (Information Extraction)
  - complete syntactic decomposition and conversion into logical expressions (deep semantic analysis)
Deep Semantic Analysis - Example

Markus does not like animal food. But he ordered a Thai dish that contains fish.

\[ \neg \exists \text{likes.}(\text{Animal} \cap \text{Food})(\text{markus}) \]
\[ \exists \text{ordered}(\text{Dish} \cap \exists \text{contains.Fish})(\text{markus}) \]
Semi-structured Resources

- as opposed to text, other digital documents such as web pages have more explicit structure (mark-up, links, etc.)
- these can be directly converted into e.g. RDF
- other examples:
  - wikis
  - file systems
Structured Resources

- databases are „fully structured“
- for coupling a database with an ontology a mapping is required that connects the database‘s schema part with classes and properties of the ontology
- then the data part of the ontology can be interpreted as ABox
Ontology Evaluation

- what makes a good ontology?
- there are different criteria, e.g.:
  - logical criteria
  - structural/formal criteria
  - „correctness“
Logical Criteria

- consistency of KB or classes

  Horse \subseteq \neg \text{Flies}
  \text{FlyingHorse} \equiv \text{Horse} \sqcap \text{Flies}

- logical completeness

  Bird \subseteq \neg \text{Mammal} \quad \text{Bird(ostrich)}
  Bird \subseteq \text{Oviparous} \quad \text{Mammal} \sqcap \text{Viviparous(lion)}
  \text{Oviparous} \subseteq \neg \text{Viviparous}

  \text{Mammal} \subseteq \text{Viviparous}

  \text{Mammal} \sqcap \text{Oviparous(platypus)}
Structural Criteria

- investigation of class hierarchy:

  Architecture ⊆ Faculty
  University ⊆ Building
  Faculty ⊆ University
  Building ⊆ Architecture

- checking correctness is rather difficult (grounding problem)
**Typical Modeling “Errors”**

- omitting disjointness
- omitting role characteristics
- domain / range too specific
- wrong interpretation of universal quantifier
- mistaking „part of“ for „subclass of“

```
Man ⊆ Human
Man(pascal)
```

```
Man ⊆ Man ∩ Woman
Woman ⊆ Human
Woman(anne)
```

```
Happy ≡ ∀hasChild.Happy
```

```
Finger ⊆ Hand
Toe ⊆ Foot
Hand ⊆ Arm
Foot ⊆ Leg
Arm ⊆ Body
Leg ⊆ Body
Arm ∩ Leg ⊆ ⊥
```
Typical Modeling “Errors”

- direction of property unclear
  
  ```
  ex:author rdfs:range ex:Publication .
  ex:author rdfs:domain ex:Person .
  ```

- subclasses vs. equivalence

- too „verbal“ translation...
Modularisation / Patterns

- facilitates reuse
- allows for faster reasoning
- usage of „best practices“
Ontology Refinement

(semi)automatic „improvement“ of ontologies

- ontology repair
- ontology update / evolution
- logical completion of ontologies