

Museum documentation in transdisciplinary perspective

Introduction to OWL-DL, Tools, Semantic Web and the CRM
OWL-DL implementation

B. Schiemann

Chair for Artificial Intelligence
University of Erlangen-Nürnberg

14. 09. 2008

OWL as recommendation

- OWL stands for Web Ontology Language
- Languages created for describing web contents
- Web today: contents and tags managing layout
- Semantic Web: additional tags to model the contents
- Future page at the internet → „Content“, XHTML (incl. CSS), OWL
- Reason: processable by machines/algorithms
- World wide accepted W3C standard
- Three sublanguages: OWL lite (test), OWL Full (RDF/S), *OWL-DL* (Description Logic)

Advantages of ontologies in *OWL-DL*

- 1 Syntax is readable for computers and humans
- 2 Formally defined semantics give expressions a clear meaning
- 3 Decidable inferences, but high expressivity
- 4 Computation of semantics supports modeling
- 5 Open World semantics
- 6 Tool support (editor, reasoner, semantic web software)

OWL-DL

T-Box (Schema):

- Class/Concept and property/role constructors and constraints
- *subClassOf* builds concept hierarchy
- Object properties of superclasses are inherited; possibly restricted
- Object properties and datatype properties
- Properties: functional, inverse, symmetric, inverse-functional
- Restrictions: some, min, max, exactly

A-Box (Data):

- Individuals representing concrete data

Syntax — *T-Box* (Example)

```
<owl:Class rdf:ID="Curator">
  <rdfs:subClassOf rdf:resource="#E21.Person" />
  <rdfs:subClassOf><owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty
        rdf:ID="is_current_keeper_of" />
      </owl:onProperty>
      <owl:someValuesFrom>
        <owl:Class
          rdf:about="#Biological_Collection_Object" />
        </owl:someValuesFrom>
      </owl:Restriction></rdfs:subClassOf>
</owl:Class>
```

Syntax — *A-Box* (Example)

```
<Curator rdf:ID="K_H_Lampe">  
  <is_current_keeper_of  
    rdf:resource="#MesopolobusfagiCollObj01 "  
  />  
</Curator>
```

Semantics of *OWL-DL*

- Processible syntax, but also a processible semantics
- *OWL-DL* (V 1.0): syntactic variant of *SHOIN(D)*
- Each above mentioned language element has corresponding semantics in this special description logic
- Why description logic? → decidable, expressive subset of (classical) First Order Logic
- Toolsupport

Reasoner implementations

- FaCT++ = Fast Classification of Terminologies
 - Tableaux algorithm (*SROIQ*) (*OWL-DL V 1.1*)
 - DIG Interface and direct integration
- RACER = Renamed ABox & Concept Expression Reasoner
 - Tableaux algorithm (*SHOIN(D)*)
 - DIG Interface (HTTP-based, nRQL)
- Pellet
 - Support for *SROIQ*, without n-ary datatypes
 - Direct integration

Editor implementations

Open Source:

- Protege
- SWOOP
- OilEd (currently not available)

Commercial (with support):

- TopBraid Composer
- Semantic Guide

Implementation in Protégé

The screenshot shows the Protégé 3.3 beta interface with the following components:

- Subclass Explorer:** A tree view on the left showing the hierarchy of classes. The selected class is **E42.Object_Identifier**, which is a subclass of **E41.Appellation**. Other visible classes include **owl.Thing**, **Axiom_1**, **Axiom_2**, **E1.CRM_Entity**, **E2.Temporal_Entity**, **E3.Condition_State**, **E4.Period**, **E52.Time-Span**, **E53.Place**, **E54.Dimension**, **E77.Persistent_Item**, **E39.Actor**, **E21.Person**, **E74.Group**, **E35.Title**, **E44.Place_Appellation**, **E49.Time_Appellation**, **E75.Conceptual_Object_Appellation**, **E82.Actor_Appellation**, **E51.Contact_Point**, and **E45.Address**.
- Class Editor:** The main workspace shows the class **E42.Object_Identifier**. It contains a table of properties:

Property	Value	Lang
<input checked="" type="checkbox"/> rdfs:comment	Unique codes assigned to objects in order to identify them uniquely within the context of one or more organizations. Typically alphanumeric sequences. Examples: MM.GE.195, 13.45.1976, etc.	
- Asserted Conditions:** A panel at the bottom right showing logical constraints. The selected condition is:
 - E41.Appellation**
 - P1241.was_transformed_by max 1** [from E77.Persistent_Item]
 - P3.has_note some E62.String** [from E1.CRM_Entity]
 - P921.was_brought_into_existence_by exactly 1** [from E77.Persistent_Item]
 - P931.was_taken_out_of_existence_by max 1** [from E77.Persistent_Item]

ISO Specification of *E77.Persistent_Item*

- Subclass of E1.CRM_Entity
- Superclass of E39.Actor, E41.Appellation, E51.Contact_Point, E70.Thing
- Scope Note: This class comprises items that have a persistent identity, sometimes known as endurants in philosophy. They can be repeatedly recognized within the duration of their existence by identity criteria rather than by continuity or observation. Persistent Items can be either physical entities, such as people, animals or things, or conceptual entities such as ideas, concepts, products of the imagination or common names.
- Examples: Leonardo da Vinci, Stonehenge, the hole in the ozone layer

E77.Persistent_Item

The screenshot displays a software interface for defining classes and properties. On the left, a tree view shows a hierarchy of classes: E1.CRM_Entity, E2.Temporal_Entity, E52.Time-Span, E53.Place, E54.Dimension, E77.Persistent_Item (selected), E39.Actor, E41.Appellation, E51.Contact_Point, E70.Thing, and E59.PrimitiveValue. On the right, a list of properties is shown, including E1.CRM_Entity, P124I.was_transformed_by max 1, P92I.was_brought_into_existence_by exactly 1, P93I.was_taken_out_of_existence_by max 1, and P3.has_note some E62.String. A toolbar with icons for adding, removing, and other actions is visible between the two panels.

E77.Persistent_Item OWL-DL I

```
<owl:Class rdf:about="#E77.Persistent_Item">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:about=
          "#P92I.was_brought_into_existence_by"/>
      </owl:onProperty>
      <owl:someValuesFrom>
        <owl:Class rdf:about=
          "#E63.Beginning_of_Existence"/>
      </owl:someValuesFrom>
    </owl:Restriction>
```

E77.Persistent_Item OWL-DL II

```
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:cardinality rdf:datatype="#int"
    >1</owl:cardinality>
    <owl:onProperty>
      <owl:ObjectProperty rdf:about
      ="#P92I.was_brought_into_existence_by"/>
    </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf rdf:resource="#E1.CRM_Entity"/>
```

...

E77.Persistent_Item OWL-DL III

```
</owl:Class>
```

Properties of *E77.Persistent_Item*

The screenshot displays a software interface for defining properties. On the left, a tree view lists various properties, each with a bidirectional arrow indicating its inverse. The selected property is `P121.was_present_at ↔ P12.occurred_in_the_presence_of`. Below the tree, a section titled "Super Properties" also lists `P121.was_present_at ↔ P12.occurred_in_the_presence_of`.

On the right, a table defines the domain and range for the selected property:

Domain	Range
<code>E77.Persistent_Item</code>	<code>E63.Beginning_of_Existence</code>

P92I.was_brought_into_existence_by OWL-DL

```
<owl:ObjectProperty
  rdf:about="#P92I.was_brought_into_existence_by">
  <rdfs:range
    rdf:resource="#E63.Beginning_of_Existence"/>
  <rdfs:subPropertyOf
    rdf:resource="#P12I.was_present_at"/>
  <owl:inverseOf>
    <owl:ObjectProperty
      rdf:about="#P92.brought_into_existence"/>
  </owl:inverseOf>
  <rdfs:domain
    rdf:resource="#E77.Persistent_Item"/>
</owl:ObjectProperty>
```

Datatypes in CRM

- Datatypes are used as representation of strings, etc.
- Datatype properties as inverse-functional properties
- Not permitted in OWL-DL, only in OWL Full
- Required to stay within OWL-DL for decidable reasoning:
 - Entities point to `E59.Primitive_Value` or subclasses via inverse-functional object properties
 - Subclasses of `E59.Primitive_Value` have datatype properties
 - Datatype properties point to xsd datatypes of XML Schema
- Example on following slides

E52.Time_Span

define the tem
valid for a cert
E49.Time_Appe
be best consid
properties of t
precise way. A

- E1.CRM_Entity
 - ▶ ● E2.Temporal_Entity
 - E52.Time-Span
 - E53.Place
 - E54.Dimension
 - ▶ ● E77.Persistent_Item
 - ▶ ● E59.PrimitiveValue

U R +

- E1.CRM_Entity
- P41.is_time-span_of **some** E2.Temporal_Entity
- P79.beginning_is_qualified_by **max** 1
- P81.ongoing_throughout **exactly** 1
- P83.had_at_least_duration **exactly** 1
- P84.had_at_most_duration **exactly** 1
- P3.has_note **some** E62.String

Properties of E52.Time_Span

The screenshot shows a software interface with two main panels. The left panel is a list of properties, and the right panel is a table with two columns: 'Domain' and 'Range'.

Property List:

- P119.meets_in_time_with ↔ P119.meets_in_time_with
- P115I.is_finished_by ↔ P115I.is_finished_by
- P86.falls_within ↔ P86.falls_within
- P117I.includes ↔ P117I.includes
- P117.occurs_during ↔ P117.occurs_during
- P90.has_value
- P57.has_number_of_parts
- P3.has_note
 - P80.end_is_qualified_by
 - P79.beginning_is_qualified_by
- P81.ongoing_throughout
- P82.at_some_time_within

Table:

Domain	Range
E52.Time-Span	E62.String

E62.String

The screenshot shows a software interface with a class hierarchy on the left and a property definition on the right. The class hierarchy includes:

- E52.Time-Span
- E53.Place
- E54.Dimension
- ▶ ● E77.Persistent_Item
- ▼ ● E59.PrimitiveValue
 - E60.Number
 - E61.Time_Primitive
 - E62.String

The right pane shows a property definition:

▶ has_PrimitiveString (multiple string) (cardinality 1)

Implementation scheme for museums

- Use CRM OWL-DL as the reference ontology
- Model the application ontology (AO) with subconcepts of CRM
- Concept hierarchy corresponds to DB schema
- Individuals correspond to database entries
- Consistency checks: DL reasoners, e.g. RACER
- Use AO as basis for new applications
- Integration of data from other museums with specialised CRM OWL-DL on CRM level
- CRM should become Reference Ontology

Literature & Links

- Protege: <http://protege.stanford.edu/>
- TopBraid:
<http://www.topquadrant.com/topbraid/composer/index.html>
- OilEd: <http://oiled.man.ac.uk/>
- Semantic Guide: <http://www.ontoprise.de/index.php?id=34>
- SWOOP: <http://code.google.com/p/swoop/>
- RACER: <http://www.racer-systems.com/>
- FaCT++: <http://owl.man.ac.uk/factplusplus/>
- Pellet: <http://pellet.owldl.com/>

Literature & Links

- Web Ontology Language (OWL) Guide Version 1.0 W3C Working Draft 4 November 2002
- W3C OWL Abstract Syntax and Semantics
<http://www.w3.org/TR/owl-semantics/>
- I. Horrocks: „Reasoning with DAML+OIL: What can it do for YOU?“
- SWRL: I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, and M. Dean. Swrl: A semantic web rule language combining owl and ruleml, 2003. Available at <http://www.daml.org/2003/11/swrl/>

Summary

- 1 OWL-DL
 - Definition
 - Language elements
 - Syntax
 - Semantics of OWL-DL
- 2 Tools
 - Reasoners
 - Editors
- 3 CRM 4.2 in OWL-DL
 - Overview
 - Example E77.Persistent_Item
 - Example concept definition
 - Example property definition
 - Datatypes in CRM

Thank you!

- For your Attention!
- Questions?