Wissenschafterinnenkolleg Internettechnologien

Technische Universität Wien

Vienna University of Technology

# Ontologien mit OWL

## 28.4.2004, 15:00 – 16:30

## AK aus Knowledge Engineering 1

Elke Michlmayr

# Übersicht

- Web Ontology Language (OWL)
  - Features
  - OWL Lite, OWL DL, OWL Full
  - Property Characteristics
  - Property Restrictions
  - Class Expressions
  - Ontology Sharing und Ontology Mapping
  - Tools

# OWL Features

- OWL Ontologie besteht aus
  - Klassen
  - Properties
  - Individuals (Instanzen von Klassen)
- Open World Assumption
  - „Absence of information is not interpreted as negative information.“
  - Beispiel: hatTier(PersonA,TierB)
    -> PersonA kann auch andere Tiere haben
- Keine Unique Name Assumption
  - PersonA und PersonB sind nicht notwendigerweise verschiedene Instanzen
  - Verschiedenheit muss explizit ausgedrückt werden

# OWL: Beispiele

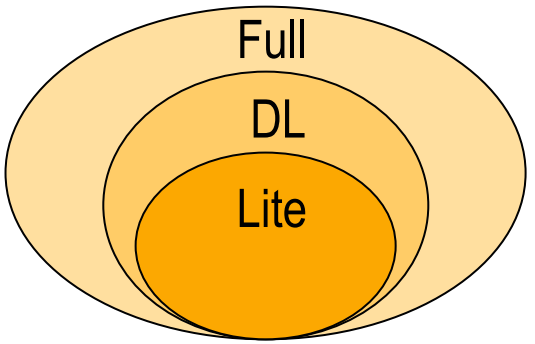
- Beziehungen zwischen Klassen bzw. Eigenschaften
  - Klasse `Tier` und Klasse `Person` sind disjunkt
  - `hatBesitzerIn` ist invers zu `hatTier`
- Beziehungen zwischen Instanzen
  - Instanz `PersonA` und Instanz `PersonB` nicht die gleiche Person
- Beschreibungen von Klassen („class expressions“)
  - Die Klasse `NamenlosePerson` besteht aus allen Instanzen der Klasse `Person`, die keinen Wert für die Property `name` haben

# Wozu Ontologien/OWL?

- Wissensrepräsentation
  - Use Case: Pizza Ontologie definiert Konzept „vegetarische Pizza“ für Agents
- Reasoning
  - Besonders wichtig bei umfangreichen Ontologien
  - Konsistenzüberprüfung
    - Falsch definierte Klassenzugehörigkeit von Instanzen
    - Überprüfung bzw. Berechnung der Taxonomie (subClassOf Hierarchie)
  - Automatische Klassifikation von Instanzen

# 3 Arten: OWL Lite, OWL DL, OWL Full

- OWL Lite: Subset von OWL DL
- OWL DL: basiert auf $\mathcal{SHIQ}$ Description Logic
  - Formal definierte Semantik
  - Entscheidbarkeit (decidability):
    Alle Berechnungen in endlicher Zeit
  - Vollständigkeit (completeness):
    Alle Schlüsse werden gezogen
  - Korrektheit (soundness):
    Alle gezogenen Schlüsse sind gültig




- OWL Full: Vereinigung von OWL Syntax und RDF
  - Erlaubt auch Metamodellierung: Konstrukte auf vordefinierte Konstrukte anwenden
  - Keine Garantie für Berechenbarkeit

# OWL Klassen

- `Class, rdfs:subclassOf`
- 2 vordefinierte Klassen
  - `Thing` (Klasse, die alle Individuals enthält)
  - `Nothing` (Leere Klasse)

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="#PotableLiquid" />
</owl:Class>
```

RDF/XML syntax

```
class(a:Wine partial a:PotableLiquid)
```

Abstract syntax

```
Wine ⊑ PotableLiquid
```

DL syntax

# OWL Properties

- `rdfs:subPropertyOf, rdfs:domain, rdfs:range`
- 2 Arten von Properties:
  - `DataTypeProperty` (rdfs:Literal, XML Schema Datentypen)
  - `ObjectProperty` (Relationen zwischen zwei Instanzen)

```
<owl:ObjectProperty rdf:ID="madeFromGrape">
  <rdfs:domain rdf:resource="#Wine" />
  <rdfs:range rdf:resource="#WineGrape" />
</owl:ObjectProperty>

<Wine rdf:ID="LindemansBin65Chardonnay">
  <madeFromGrape rdf:resource="#ChardonnayGrape" />
</Wine>
```

# OWL Properties: globale Characteristics

- `TransitiveProperty`
  A liegtIn B, B liegtIn C
  -> A liegtIn C

- `SymmetricProperty`
  A grenztAn B
  -> B grenztAn A

- `FunctionalProperty`
  A hatMutter B, A hatMutter C
  -> B = C
  (Eindeutige Abbildung:
  Eine bestimmte Instanz kann max. einen Wert haben)

- `inverseOf`
  hatMutter ist invers zu MutterVon

- `InverseFunctionalProperty`
  B MutterVon A, C MutterVon A
  -> B = C
  (Ein bestimmtes Range-Element kann nur für eine Instanz gelten)
  Darf nur in OWL Full für DatatypeProperty verwendet werden, z.B. Sozialversicherungsnummer

# OWL Properties: Restrictions (1/3)

- Restrictions sind lokal gültig
  - für eine Property in Bezug auf eine Klasse
- Kardinalität
  - `cardinality` exakte Anzahl (OWL Lite: Nur 0 oder 1 erlaubt)
  - `minCardinality, maxCardinality` untere und obere Grenze

```
<owl:Class rdf:about="#Weinlese">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hatWeinleseJahr"/>
   <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
     1
   </owl:cardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

# OWL Properties: Restrictions (2/3)

- Allquantor: $\forall P.C$
  - `allValuesFrom` legt alle Instanzen einer bestimmten Klasse als erlaubten Wertebereich für eine bestimmte Property fest
- Existenzquantor: $\exists P.C$
  - `someValuesFrom` definiert, dass eine Property existieren muss
  - und legt den Wertebereich für diese Property fest

```
<owl:Class rdf:about="Wine">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasMaker" />
   <owl:allValuesFrom rdf:resource="#Winery" />
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

# OWL Properties: Restrictions (3/3)

- Konstante (OWL DL):
  - `hasValue` definiert einen fixen Wert für eine Property
  - Alle Instanzen, die den definierten Wert für diese Property besitzen, sind Mitglieder der Klasse

```
<owl:Class rdf:ID=„DryWine">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasSugar" />
   <owl:hasValue rdf:resource="#Dry" />
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>

<WineSugar rdf:ID="Dry" />
```

Was fehlt hier?

# Konstruktoren für Class Expressions (OWL DL)

- Vereinigung: `unionOf`
  - „Die Klasse Fruit ist die Vereinigung der Klassen SweetFruit und NonSweetfruit“
- Durchschnitt: `intersectionOf`
  - „Die Klasse DryWine ist der Durchschnitt der Klasse Wine und der Klasse aller Objekte mit Wert Dry für Property hasSugar“
- Komplement: `complementOf`
  - Selektiert alle Individuals einer Domäne, die nicht Mitglied einer bestimmten Klasse sind
- Disjunktivität: `disjointWith`
  - Kein Individual kann gleichzeitig Mitglied zweier disjunkter Klassen sein
- Aufzählung: `oneOf`
  - Definition einer Klasse durch Aufzählung aller Individuals, die Mitglieder der Klasse sind

# Class Expressions: Beispiele (1/3)

```
<owl:Class rdf:ID="Fruit">
 <owl:unionOf rdf:parseType="Collection">
  <owl:Class rdf:about="#SweetFruit" />
  <owl:Class rdf:about="#NonSweetFruit" />
 </owl:unionOf>
</owl:Class>

<owl:Class rdf:ID="DryWine">
 <owl:intersectionOf rdf:parseType="Collection">
  <owl:Class rdf:about="#Wine" />
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasSugar" />
   <owl:hasValue rdf:resource="#Dry" />
  </owl:Restriction>
 </owl:intersectionOf>
</owl:Class>
```

# Class Expressions: Beispiele (2/3)

```
<owl:Class rdf:ID="WineColor">
 <rdfs:subClassOf rdf:resource="#WineDescriptor" />
 <owl:oneOf rdf:parseType="Collection">
  <owl:Thing rdf:about="#White" />
  <owl:Thing rdf:about="#Rose" />
  <owl:Thing rdf:about="#Red" />
 </owl:oneOf>
</owl:Class>

<owl:Class rdf:ID="NonConsumableThing">
 <owl:complementOf rdf:resource="#ConsumableThing" />
</owl:Class>

<owl:Class rdf:ID="Meat">
 <rdfs:subClassOf rdf:resource="#EdibleThing" />
 <owl:disjointWith rdf:resource="#Seafood" />
 <owl:disjointWith rdf:resource="#Dessert" />
 <owl:disjointWith rdf:resource="#Fruit" />
</owl:Class>
```

# Class Expressions: Beispiele (3/3)

```
<owl:Class rdf:about="#giraffe">
 <rdfs:subClassOf rdf:resource="#animal" />
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#eats" />
   <owl:allValuesFrom rdf:resource="#leaf" />
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:about="#leaf">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#part_of" />
   <owl:someValuesFrom rdf:resource="#tree" />
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

# Ontology Sharing/Ontology Mapping

- Eine OWL Ontologie kann andere OWL Ontologien importieren:

```
<owl:Ontology rdf:about="">
  <owl:imports rdf:resource="http://www.xyz.org/ont.owl"/>
  ...
</owl:Ontology>
```

- Mapping zwischen OWL Ontologien
  - Klassen `equivalentClass`
  - Eigenschaften `equivalentProperty`
  - Individuals `sameAs, differentFrom, AllDifferent`

# Zusammenfassung: Konstruktoren

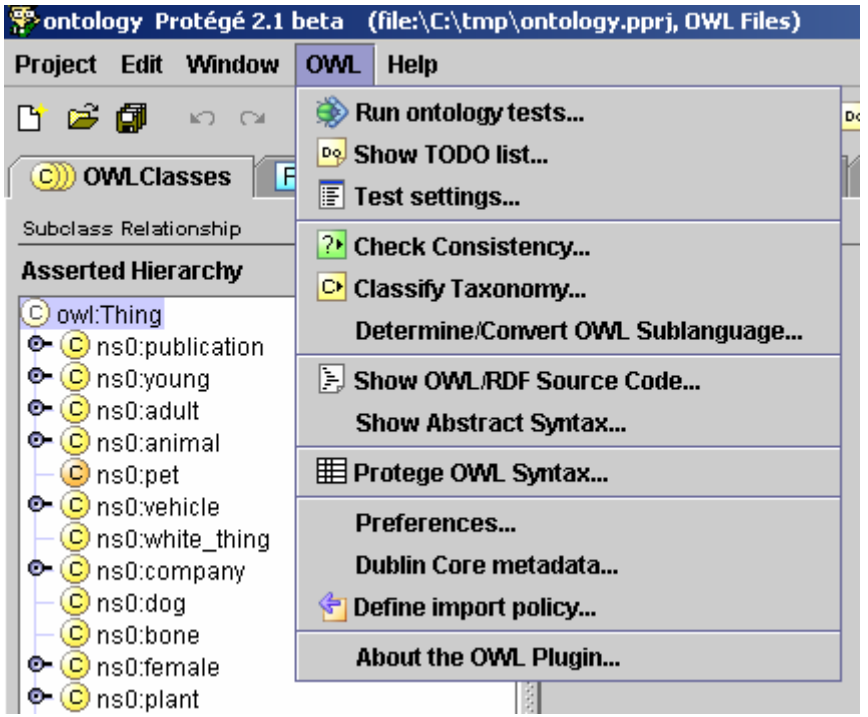
| Constructor | DL Syntax | Example |
|---|---|---|
| intersectionOf | $C_1 \sqcap \ldots \sqcap C_n$ | Human $\sqcap$ Male |
| unionOf | $C_1 \sqcup \ldots \sqcup C_n$ | Doctor $\sqcup$ Lawyer |
| complementOf | $\neg C$ | $\neg$Male |
| oneOf | $\{x_1\} \sqcup \ldots \sqcup \{x_n\}$ | {john} $\sqcup$ {mary} |
| allValuesFrom | $\forall P.C$ | $\forall$hasChild.Doctor |
| someValuesFrom | $\exists P.C$ | $\exists$hasChild.Lawyer |
| maxCardinality | $\leqslant nP$ | $\leqslant$1hasChild |
| minCardinality | $\geqslant nP$ | $\geqslant$2hasChild |

# Zusammenfassung: Axiome

| Axiom | DL Syntax | Example |
|---|---|---|
| subClassOf | $C_1 \sqsubseteq C_2$ | Human $\sqsubseteq$ Animal $\sqcap$ Biped |
| equivalentClass | $C_1 \equiv C_2$ | Man $\equiv$ Human $\sqcap$ Male |
| disjointWith | $C_1 \sqsubseteq \neg C_2$ | Male $\sqsubseteq \neg$Female |
| sameIndividualAs | $\{x_1\} \equiv \{x_2\}$ | {President_Bush} $\equiv$ {G_W_Bush} |
| differentFrom | $\{x_1\} \sqsubseteq \neg\{x_2\}$ | {john} $\sqsubseteq \neg${peter} |
| subPropertyOf | $P_1 \sqsubseteq P_2$ | hasDaughter $\sqsubseteq$ hasChild |
| equivalentProperty | $P_1 \equiv P_2$ | cost $\equiv$ price |
| inverseOf | $P_1 \equiv P_2^-$ | hasChild $\equiv$ hasParent$^-$ |
| transitiveProperty | $P^+ \sqsubseteq P$ | ancestor$^+ \sqsubseteq$ ancestor |
| functionalProperty | $\top \sqsubseteq \leqslant 1 P$ | $\top \sqsubseteq \leqslant 1$hasMother |
| inverseFunctionalProperty | $\top \sqsubseteq \leqslant 1 P^-$ | $\top \sqsubseteq \leqslant 1$hasSSN$^-$ |

# Tools

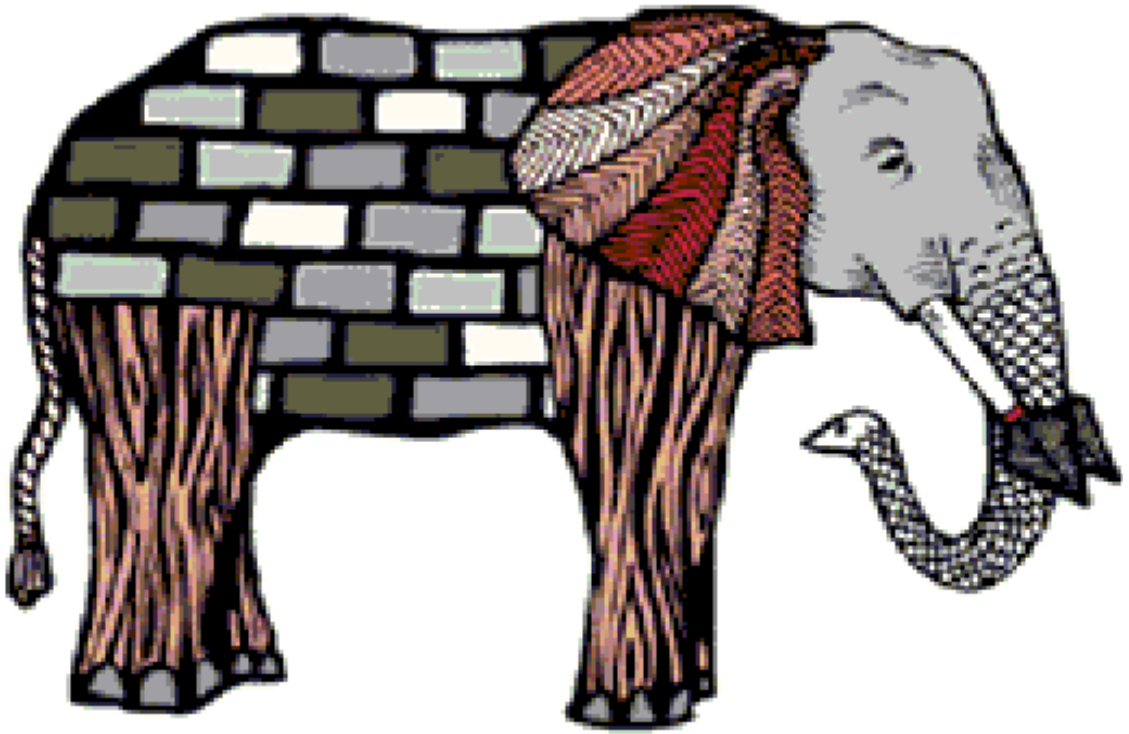
- Editoren
  - Protege ( + OWL Plugin) http://protege.stanford.edu/
  - OntoEdit http://www.ontoprise.de/customercenter/software_downloads/free
- DL Reasoner
  - Racer http://www.cs.concordia.ca/~haarslev/racer/
  - FaCT
- OWL Validator http://owl.bbn.com/validator/



Protege + OWL Plugin + Racer

# Literatur

- OWL Guide
  http://www.w3.org/TR/owl-guide/
- Ontology Development 101
  http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.pdf
- OWL Reasoning Examples
  http://owl.man.ac.uk/2003/why/latest/
- Beispielontologien
  http://www.daml.org/ontologies/
- OWL Tutorial
  http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial/



http://www.w3.org/Talks/2001/1102-semweb-fin/all.htm