Modul Semantik Web (SS2011)

Dr. Sören Auer
Dr. Jens Lehmann
Prof. Dr. Gerhard Brewka
Frank Loebe

Institut für Informatik · Universität Leipzig

OWL - Syntax und Intuition

10. Mai 2011

AKSW

Semantic Web

1. Einleitung und Ausblick
2. URIs und Einführung in RDF
3. RDF Schema
4. Logik – Grundlagen
5. Semantik von RDF(S)
6. **OWL – Syntax und Intuition**
7. OWL – Semantik und Reasoning
8. Spezifikation von Regeln in RDF - RIF
9. RDF-Datenbanken, Triple- und Knowledge-Stores, Anfragesprachen SPARQL, SPARUL
10. Integration von RDF und XHTML - RDFa, GRDDL
11. Linked Data Web, Semantische Wikis
12. Semantic Web Anwendugen, Rück- und Ausblick

AKSW

Gliederung

1. Motivation
2. OWL-Allgemein
3. Klassen, Rollen und Individuen
4. Klassenbeziehungen
5. komplexe Klassen
6. Eigenschaften von Rollen
7. OWL Varianten
8. Referenz & Übersicht

AKSW

Gliederung

AKSW

Ontologie-Philosophisch

- Begriff existiert nur in der Einzahl (es gibt also keine Ontologien)
- bezeichnet die "Lehre vom Sein"
- zu finden bei Aristoteles (Sokrates), Thomas von Aquin, Descartes, Kant, Hegel, Wittgenstein, Heidegger, Quine, ...

AKSW

Ontologie-Informatisch

Gruber (1993):
“An Ontology is a
formal specification
of a **shared**
conceptualization
of a **domain** of interest.”

Maschinell interpretierbar
Beruht auf Konsens
Beschreibt Begrifflichkeiten
Bezogen auf ein “Thema”
(Gegenstandsbereich)

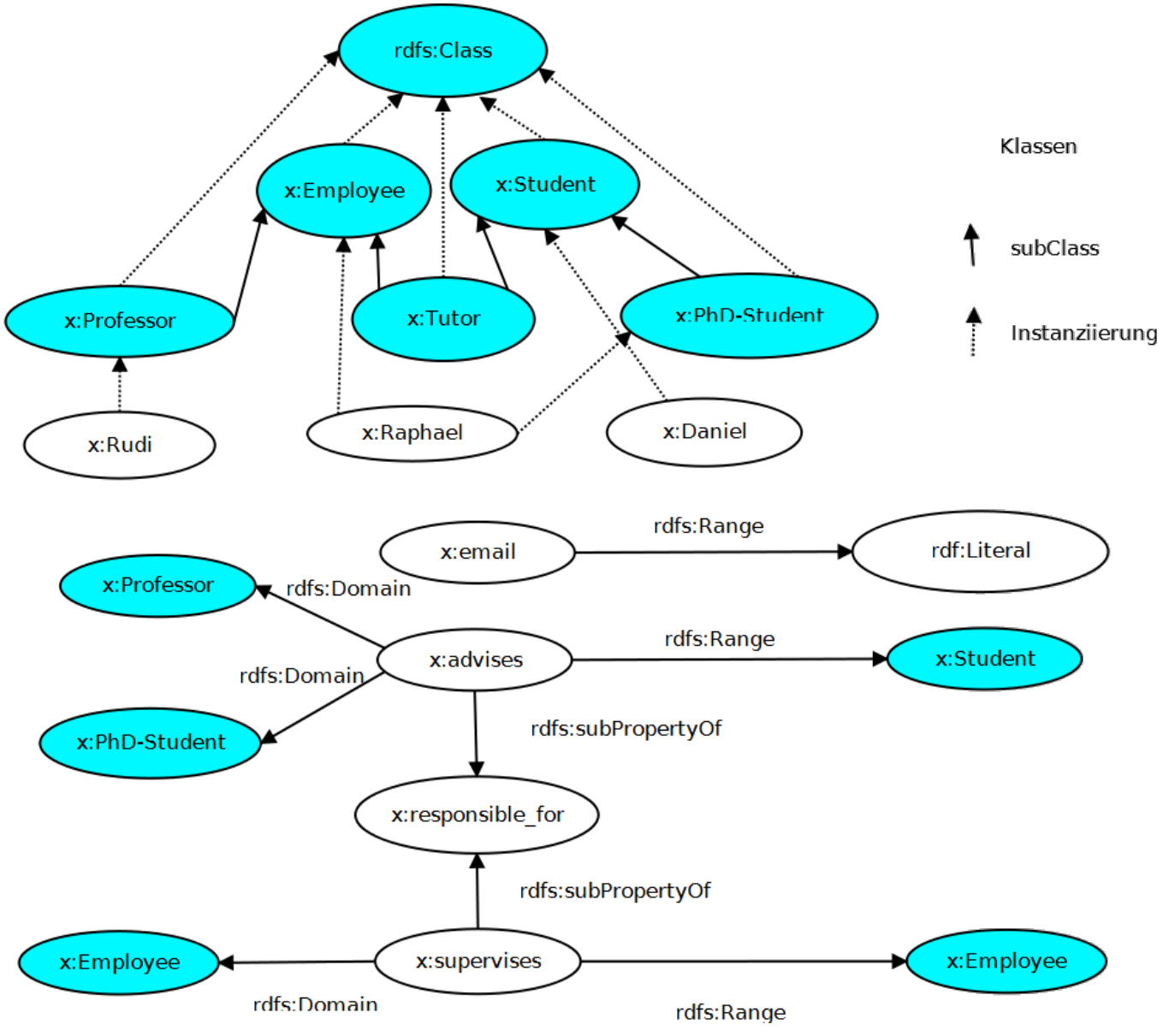
AKSW

Ontologie-Praktisch Einige Anforderungen

- Instanziierung von Klassen durch Individuen
- Begriffshierarchien (Taxonomien, Vererbung): Klassen, Begriffe
- binäre Relationen zwischen Individuen: Properties, Roles
- Eigenschaften von Relationen (z.B. range, transitive)
- Datentypen (z.B. Zahlen): concrete domains
- logische Ausdrucksmittel
- klare Semantik!

AKSW

RDFS-Einfache Ontologien



RDF Schema als Ontologiesprache?

- geeignet für einfache Ontologien
- Vorteil: automatisches Schlussfolgern ist relativ effizient
- aber: für komplexere Modellierungen ungeeignet
- Rückgriff auf mächtigere Sprachen, wie
  - OWL
  - F-Logik
  - ...

AKSW

Gliederung

AKSW

OWL-Allgemeines

- **W3C Recommendation** seit 2004
- Semantisches Fragment von FOL
- Drei Varianten: OWL Lite $\subseteq$ OWL DL $\subseteq$ OWL Full
- Keine Reifikation in OWL DL $\Rightarrow$ RDFS ist Fragment von OWL Full
- OWL DL ist entscheidbar und entspricht der Beschreibungslogik $\mathcal{SHOIN}$
- W3C-Dokumente (Vorlesungswebseite) enthalten Details, die hier nicht alle angesprochen werden können.

OWL Dokumente

- sind RDF/XML Dokumente in der Standard-Syntax
- es gibt auch andere Formate die häufig sogar leichter zu lesen und zu verarbeiten sind
- Einfaches Beispiel: Ontology `http://my-ontology.org` mit zwei Klassen Person und Student und einer Subklassenbeziehung dazwischen

OWL Syntax

- RDF/XML:

```
<?xml version="1.0"?><!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>
<rdf:RDF xmlns="http://my-ontology.org/"
    xml:base="http://my-ontology.org/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about=""/>
    <owl:Class rdf:about="Person">
        <rdfs:subClassOf rdf:resource="&owl;Thing"/>
    </owl:Class>
    <owl:Class rdf:about="Student">
        <rdfs:subClassOf rdf:resource="Person"/>
    </owl:Class>
</rdf:RDF>
```

OWL Syntax

- OWL/XML:

```
<?xml version="1.0"?>
    <!DOCTYPE Ontology [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    [...]
]>
<Ontology [...]
    URI="http://my-ontology.org/">
    <SubClassOf>
        <Class URI="&my-ontology;Person"/>
        <Class URI="&owl;Thing"/>
    </SubClassOf>
    <SubClassOf>
        <Class URI="&my-ontology;Student"/>
        <Class URI="&my-ontology;Person"/>
    </SubClassOf>
</Ontology>
```

OWL Syntax

- Turtle (siehe auch N3 und N-Triples):

```
@prefix       : <http://my-ontology.org/> .
@prefix rdfs  : <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl   : <http://www.w3.org/2002/07/owl#> .
@prefix rdf   : <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@base <http://my-ontology.org/> .
<http://my-ontology.org/>   rdf:type          owl:Ontology  .
:Person                     rdf:type          owl:Class  .
:Student                    rdf:type          owl:Class  ;
                            rdfs:subClassOf   :Person  .
```

OWL Syntax

- Manchester OWL Syntax:

```
Prefix: rdfs = <http://www.w3.org/2000/01/rdf-schema#>
Prefix: owl = <http://www.w3.org/2002/07/owl#>
Prefix: rdf = <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Ontology: <http://my-ontology.org/>
Class: Person
  SubClassOf: owl:Thing
Class: Student
  SubClassOf: Person
```

- Beschreibungslogik: $\mathit{Student} \sqsubseteq \mathit{Person}$
- funktionale Syntax
- KRSS2
- OBO

AKSW

OWL Dokumente

- **Mehrere Syntaxvarianten** für OWL **je nach Anwendungsfall**:
  - RDF/XML: Standardsyntax/Datenaustausch
  - OWL/XML: einfacher für XML-Werkzeuge
  - Turtle: einfacher RDF Triple zu lesen/schreiben
  - MOS: einfacher DL-Ontologien zu lesen/schreiben
  - Funktional: einfacher formale Struktur zu sehen
- Bereitstellung von **RDF/XML "Pflicht"** bei Veröffentlichung von Ontologien, andere Varianten optional
- OWL-Datei besteht aus:
  - Kopf mit allgemeinen Angaben
  - Rest mit eigentlicher Ontologie

AKSW

Der Kopf eines OWL Dokumentes

- Definition von Namespaces in der Wurzel

```
<rdf:RDF
xmlns="http://www.semanticweb-grundlagen.de/beispielontologie#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#">
...
</rdf:RDF>
```

Der Kopf eines OWL Dokumentes

- Allgemine Informationen

```
<owl:Ontology rdf:about="">
  <rdfs:comment rdf:datatype="&xsd;string">
      SWRC Ontologie in der Version vom Dezember 2005
  </rdfs:comment>
  <owl:versionInfo>v0.5</owl:versionInfo>
  <owl:imports rdf:resource="http://www.semanticwebgrundlagen.de/foo"/>
  <owl:priorVersion rdf:resource="http://ontoware.org/projects/swrc"/>
</owl:Ontology>
```

AKSW

Der Kopf eines OWL Dokumentes

von RDFS geerbt
rdfs:comment
rdfs:label
rdfs:seeAlso
rdfs:isDefinedBy

außerdem
owl:imports

für Versionierung
owl:versionInfo
owl:priorVersion
owl:backwardCompatibleWith
owl:incompatibleWith
owl:DeprecatedClass
owl:DeprecatedProperty

AKSW

Gliederung

AKSW

Klassen, Rollen und Individuen

Die drei Bausteine von Ontologieaxiomen (analog zu RDFS):

- **Individuen/Objekte**
  - konkrete Elemente in modellierter Welt
- **Klassen/Konzepte**
  - Mengen von Objekten
- **Rollen/Properties**
  - verknüpfen zwei Individuen

Klassen

Definition

RDF/XML Syntax

```
<owl:Class rdf:about="Professor"/>
```

Manchester Syntax

```
Class: Professor
```

Turtle Syntax

```
:Professor rdf:type owl:Class .
```

vordefiniert:
owl:Thing
owl:Nothing

Individuen

Definition durch Klassenzugehörigkeit

RDF/XML Syntax

```
<rdf:Description rdf:ID="Faehnrich">
    <rdf:type rdf:resource="#Professor"/>
</rdf:Description>
```

gleichbedeutend:

RDF/XML Syntax

```
<Professor rdf:ID="Faehnrich"/>
```

Manchester Syntax

```
Individual: Faehnrich
  Types: Professor
```

Turtle Syntax

```
:Faehnrich      rdf:type      :Professor .
```

abstrakte Rollen

abstrakte Rollen werden definiert wie Klassen

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="zugehoerigkeit"/>
```

Domain und Range abstrakte Rollen

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="zugehoerigkeit">
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range rdf:resource="#Organisation"/>
</owl:ObjectProperty>
```

abstrakte Rollen

abstrakte Rollen werden definiert wie Klassen

Manchester Syntax

```
ObjectProperty: zugehoerigkeit
```

Domain und Range abstrakte Rollen

Manchester Syntax

```
ObjectProperty: zugehoerigkeit
    Domain: Person
    Range: Organisation
```

abstrakte Rollen

abstrakte Rollen werden definiert wie Klassen

Turtle Syntax

```
:zugehoerigkeit    rdf:type        owl:ObjectProperty  .
```

Domain und Range abstrakte Rollen

Turtle Syntax

```
:zugehoerigkeit  rdf:type       owl:ObjectProperty  ;
                 rdfs:range     :Organisation  ;
                 rdfs:domain    :Person  .
```

konkrete Rollen

konkrete Rollen haben Datentyp im Range

RDF/XML Syntax

```
<owl:DatatypeProperty rdf:ID="vorname"/>
```

Domain und Range konkreter Rollen

RDF/XML Syntax

```
<owl:DatatypeProperty rdf:ID="vorname">
    <rdfs:domain rdf:resource="#Person" />
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

Viele XML Datentypen können verwendet werden. Im Standard vorgeschrieben sind *integer* und *string*.

AKSW

konkrete Rollen

konkrete Rollen haben Datentyp im Range

Manchester Syntax

```
DatatypeProperty: vorname
```

Domain und Range konkreter Rollen

Manchester Syntax

```
DatatypeProperty: vorname
  Domain: Person
  Range: string
```

Viele XML Datentypen können verwendet werden. Im Standard vorgeschrieben sind *integer* und *string*.

konkrete Rollen

konkrete Rollen haben Datentyp im Range

Turtle Syntax

```
:vorname      rdf:type      owl:DatatypeProperty  .
```

Domain und Range konkreter Rollen

Turtle Syntax

```
:vorname                    rdfs:domain      :Person  ;
                            rdfs:range       xsd:string  .
```

Viele XML Datentypen können verwendet werden. Im Standard vorgeschrieben sind *integer* und *string*.

Individuen und Rollen

RDF/XML Syntax

```
<Person rdf:ID="Faehnrich">
    <zugehoerigkeit rdf:resource="#InfAI"/>
    <zugehoerigkeit rdf:resource="#BIS"/>
    <vorname rdf:datatype="&xsd;string">Klaus-Peter</vorname>
</Person>
```

Rollen sind im allgemeinen nicht funktional.

Individuen und Rollen

Manchester Syntax

```
Individual: Faehnrich
  Types: Person
  Facts: zugehoerigkeit InfAI, zugehoerigkeit BIS, vorname Klaus-Peter
```

Rollen sind im allgemeinen nicht funktional.

Individuen und Rollen

Turtle Syntax

```
:Faehnrich      rdf:type            :Person ;
                :zugehoerigkeit     :InfAI ,
                                    :BIS ;
                :vorname            "Klaus-Peter"^^xsd:string .
```

Rollen sind im allgemeinen nicht funktional.

Modellierungsbeispiel

Modellierungsaufgabe: Es gibt männliche und weibliche Personen.
Personen haben einen Namen.

Modellierungsbeispiel

Modellierungsaufgabe: Es gibt männliche und weibliche Personen. Personen haben einen Namen.

Lösung (Turtle Syntax)

```
:Male       rdfs:subClassOf   :Person .
:Female     rdfs:subClassOf   :Person .
:hasName    rdf:type          owl:DatatypeProperty ;
            rdfs:domain       :Person ;
            rdfs:range        xsd:string .
```

Modellierungsbeispiel

Modellierungsaufgabe: Es gibt männliche und weibliche Personen. Personen haben einen Namen.

Lösung (Turtle Syntax)

```
:Male       rdfs:subClassOf    :Person .
:Female     rdfs:subClassOf    :Person .
:hasName    rdf:type           owl:DatatypeProperty ;
            rdfs:domain        :Person ;
            rdfs:range         xsd:string .
:John       rdf:type           :Male ;
            :hasName           "John"^^xsd:string .
:Jill       rdf:type           :Female .
            :hasName           "Jill"^^xsd:string .
```

AKSW

Gliederung

AKSW

Einfache Klassenbeziehungen – rdfs:subClassOf

RDF/XML Syntax

```
<owl:Class rdf:ID="Professor">
    <rdfs:subClassOf rdf:resource="#Fakultaetsmitglied"/>
</owl:Class>
<owl:Class rdf:ID="Fakultaetsmitglied">
    <rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>
```

Es folgt durch Inferenz, dass *Professor* eine Subklasse von *Person* ist.

Einfache Klassenbeziehungen – rdfs:subClassOf

Manchester Syntax

```
Class: Professor
  SubClassOf: Fakultaetsmitglied
Class: Fakultaetsmitglied
  SubClassOf: Person
```

Es folgt durch Inferenz, dass *Professor* eine Subklasse von *Person* ist.

Einfache Klassenbeziehungen – rdfs:subClassOf

Turtle Syntax

```
:Professor              rdf:type            owl:Class ;
                        rdfs:subClassOf     :Fakultaetsmitglied .
:Fakultaetsmitglied     rdf:type            owl:Class ;
                        rdfs:subClassOf     :Person .
```

Es folgt durch Inferenz, dass *Professor* eine Subklasse von *Person* ist.

AKSW

Einfache Klassenbeziehungen – owl:disjointWith

RDF/XML Syntax

```
<owl:Class rdf:ID="Professor">
    <rdfs:subClassOf rdf:resource="#Fakultaetsmitglied"/>
</owl:Class>
<owl:Class rdf:ID="Buch">
    <rdfs:subClassOf rdf:resource="#Publikation"/>
</owl:Class>
<owl:Class rdf:ID="Fakultaetsmitglied">
    <owl:disjointWith rdf:resource="#Publikation"/>
</owl:Class>
```

Es folgt durch Inferenz, dass *Professor* und *Buch* ebenfalls disjunkte Klassen sind.

AKSW

Einfache Klassenbeziehungen – owl:disjointWith

Manchester Syntax

```
Class: Professor
  SubClassOf: Fakultaetsmitglied
Class: Buch
  SubClassOf: Publikation
Class: Fakultaetsmitglied
  DisjointWith: Publikation
```

Es folgt durch Inferenz, dass *Professor* und *Buch* ebenfalls disjunkte Klassen sind.

AKSW

Einfache Klassenbeziehungen – owl:disjointWith

Turtle Syntax

```
:Buch                   rdf:type            owl:Class ;
                        rdfs:subClassOf     :Publikation .
:Fakultaetsmitglied     rdf:type            owl:Class ;
                        owl:disjointWith    :Publikation .
:Professor              rdf:type            owl:Class ;
                        rdfs:subClassOf     :Fakultaetsmitglied .
:Publikation            rdf:type            owl:Class .
```

Es folgt durch Inferenz, dass *Professor* und *Buch* ebenfalls disjunkte Klassen sind.

Einfache Klassenbeziehungen – owl:equivalentClass

RDF/XML Syntax

```
<owl:Class rdf:ID="Buch">
    <rdfs:subClassOf rdf:resource="#Publikation"/>
</owl:Class>
<owl:Class rdf:about="#Publikation">
    <owl:equivalentClass rdf:resource="#Publikationen"/>
</owl:Class>
```

Es folgt durch Inferenz, dass *Buch* eine Subklasse von *Publikationen* ist.

Einfache Klassenbeziehungen – owl:equivalentClass

Manchester Syntax

```
Class: Buch
  SubClassOf: Publikation
Class: Publikation
  EquivalentTo: Publikationen
```

Es folgt durch Inferenz, dass *Buch* eine Subklasse von *Publikationen* ist.

Einfache Klassenbeziehungen – owl:equivalentClass

Turtle Syntax

```
:Buch                rdf:type                owl:Class ;
                     rdfs:subClassOf         :Publikation .
:Publikation         rdf:type                owl:Class ;
                     owl:equivalentClass     :Publikationen.
:Publikationen       rdf:type                owl:Class .
```

Es folgt durch Inferenz, dass *Buch* eine Subklasse von *Publikationen* ist.

AKSW

Individuen und Klassenbeziehungen

RDF/XML Syntax

```
<Buch rdf:ID="SemanticWebGrundlagen">
    <Autor rdf:resource="#PascalHitzler"/>
    <Autor rdf:resource="#MarkusKroetzsch"/>
    <Autor rdf:resource="#SebastianRudolph"/>
    <Autor rdf:resource="#YorkSure"/>
</Buch>

<owl:Class rdf:ID="Buch">
    <rdfs:subClassOf rdf:resource="#Publikation"/>
</owl:Class>
```

Es folgt durch Inferenz, dass *SemanticWebGrundlagen* eine *Publikation* ist.

Individuen und Klassenbeziehungen

Manchester Syntax

```
Class: Buch
  SubClassOf: Publikation
Individual: SemanticWebGrundlagen
  Types: Buch
  Facts: Autor PascalHitzler, Autor MarkusKroetzsch,
     Autor SebastianRudolph, Autor YorkSure
```

Es folgt durch Inferenz, dass *SemanticWebGrundlagen* eine *Publikation* ist.

Individuen und Klassenbeziehungen

Turtle Syntax

```
:Buch                    rdf:type          owl:Class ;
                         rdfs:subClassOf   :Publikation .
:SemanticWebGrundlagen   rdf:type          :Buch ;
                         :Autor            :MarkusKroetzsch ,
                                           :PascalHitzler ,
                                           :SebastianRudolph ,
                                           :YorkSure .
```

Es folgt durch Inferenz, dass *SemanticWebGrundlagen* eine *Publikation* ist.

AKSW

Beziehungen zwischen Individuen

RDF/XML Syntax

```
<Professor rdf:ID="Faehnrich"/>
<rdf:Description rdf:about="#Faehnrich">
    <owl:sameAs rdf:resource="#ProfessorFaehnrich"/>
</rdf:Description>
```

Es folgt durch Inferenz, dass *ProfessorFaehnrich* ein *Professor* ist.

Verschiedenheit von Individuen wird spezifiziert mittels **owl:differentFrom**.

Beziehungen zwischen Individuen

Manchester Syntax

```
Individual: Faehnrich
  SameAs: ProfessorFaehnrich
```

Es folgt durch Inferenz, dass *ProfessorFaehnrich* ein *Professor* ist.

Verschiedenheit von Individuen wird spezifiziert mittels **owl:differentFrom**.

Beziehungen zwischen Individuen

Turtle Syntax

```
:Faehnrich      rdf:type      :Professor ;
                owl:sameAs    :ProfessorFaehnrich .
```

Es folgt durch Inferenz, dass *ProfessorFaehnrich* ein *Professor* ist.

Verschiedenheit von Individuen wird spezifiziert mittels **owl:differentFrom**.

Beziehungen zwischen Individuen

RDF/XML Syntax

```
<owl:AllDifferent>
    <owl:distinctMembers rdf:parseType="Collection">
        <Person rdf:about="#RudiStuder"/>
        <Person rdf:about="#YorkSure"/>
        <Person rdf:about="#PascalHitzler"/>
    </owl:distinctMembers>
</owl:AllDifferent>
```

Abgekürzte Schreibweise anstelle der Verwendung von mehreren *owl:differentFrom*. Der Einsatz von *owl:AllDifferent* und *owl:distinctMembers* ist nur dafür vorgesehen.

AKSW

Beziehungen zwischen Individuen

Manchester Syntax

```
DifferentIndividuals: YorkSure, PascalHitzler, RudiStuder
```

Turtle Syntax

```
[ rdf:type owl:AllDifferent ;
  owl:distinctMembers ( :PascalHitzler
                        :RudiStuder
                        :YorkSure
                      )
] .
```

AKSW

Abgeschlossene Klassen

RDF/XML Syntax

```
<owl:Class rdf:about="#SekretaerinnenVonFaehnrich">
    <owl:oneOf rdf:parseType="Collection">
        <Person rdf:about="#HanneloreSchoebel"/>
        <Person rdf:about="#KerstinBeier"/>
    </owl:oneOf>
</owl:Class>
```

Dies besagt, dass es nur genau diese beiden *SekretaerinnenVonFaehnrich* gibt.

AKSW

Abgeschlossene Klassen

Manchester Syntax

```
Class: SekretaerinnenVonFaehnrich
  EquivalentTo: {HanneloreSchoebel, KerstinBeier}
```

Dies besagt, dass es nur genau diese beiden *SekretaerinnenVonFaehnrich* gibt.

Abgeschlossene Klassen

Turtle Syntax

```
:SekretaerinnenVonFaehnrich  rdf:type          owl:Class ;
        owl:equivalentClass [ rdf:type owl:Class ;
                              owl:oneOf ( :HanneloreSchoebel
                                          :KerstinBeier
                                        )
                            ] .
```

Dies besagt, dass es nur genau diese beiden *SekretaerinnenVonFaehnrich* gibt.

Gliederung

AKSW

Logische Klassenkonstruktoren

- logisches Und (Konjunktion):
  owl:intersectionOf
- logisches Oder (Disjunktion):
  owl:unionOf
- logisches Nicht (Negation):
  owl:complementOf
- Werden verwendet, um komplexe Klassen aus einfachen Klassen zu konstruieren.

AKSW

Konjunktion

RDF/XML Syntax

```
<owl:Class rdf:about="#SekretaerinnenVonFaehnrich">
    <owl:equivalentClass>
        <owl:intersectionOf rdf:parseType="Collection">
            <owl:Class rdf:about="#Sekretaerinnen"/>
            <owl:Class rdf:about="#AngehoerigeAbteilungFaehnrich"/>
        </owl:intersectionOf>
    </owl:equivalentClass>
</owl:Class>
```

Es folgt z.B. durch Inferenz, dass alle *SekretaerinnenVonFaehnrich* auch *Sekretaerinnen* sind.

AKSW

Konjunktion

Manchester Syntax

```
Class: SekretaerinnenVonFaehnrich
  EquivalentTo: Sekretaerinnen and AngehoerigeAbteilungFaehnrich
```

Es folgt z.B. durch Inferenz, dass alle *SekretaerinnenVonFaehnrich* auch *Sekretaerinnen* sind.

Konjunktion

Turtle Syntax

```
:SekretaerinnenVonFaehnrich  owl:equivalentClass  [
    rdf:type                 owl:Class ;
    owl:intersectionOf       (:Sekretaerinnen :AngehoerigeAbteilungFaehnrich)
 ] .
```

Es folgt z.B. durch Inferenz, dass alle *SekretaerinnenVonFaehnrich* auch *Sekretaerinnen* sind.

Disjunktion

RDF/XML Syntax

```
<owl:Class rdf:about="#Professor">
    <rdfs:subClassOf>
        <owl:unionOf rdf:parseType="Collection">
            <owl:Class rdf:about="#aktivLehrend"/>
            <owl:Class rdf:about="#imRuhestand"/>
        </owl:unionOf>
    </rdfs:subClassOf>
</owl:Class>
```

Disjunktion

Manchester Syntax

```
Class: Professor
  SubClassOf: aktivLehrend or imRuhestand
```

Disjunktion

Turtle Syntax

```
:Professor  rdfs:subClassOf  [
    rdf:type                 owl:Class ;
    owl:UnionOf  ( :aktivLehrend :imRuhestand )
 ] .
```

Negation

RDF/XML Syntax

```
<owl:Class rdf:ID="Fakultaetsmitglied">
    <rdfs:subClassOf>
        <owl:complementOf rdf:resource="#Publikation"/>
    </rdfs:subClassOf>
</owl:Class>
```

semantisch Äquivalente Aussage:

RDF/XML Syntax

```
<owl:Class rdf:ID="Fakultaetsmitglied">
    <owl:disjointWith rdf:resource="#Publikation"/>
</owl:Class>
```

AKSW

Negation

Manchester Syntax

```
Class: Fakultaetsmitglied
  SubClassOf: not Publikation
```

Negation

Turtle Syntax

```
:Fakultaetsmitglied rdf:type owl:Class ;
                    rdfs:subClassOf [ rdf:type owl:Class ;
                                      owl:complementOf :Publikation
                                    ] .
```

Modellierungsbeispiel

Modellierungsaufgabe: Alle Personen sind männlich oder weiblich, aber nie beides gleichzeitig.

Modellierungsbeispiel

Modellierungsaufgabe: Alle Personen sind männlich oder weiblich, aber nie beides gleichzeitig.

Lösung (Turtle Syntax)

```
:Male       rdfs:subClassOf   :Person .
:Female     rdfs:subClassOf   :Person ;
            owl:disjointWith   :Male .
:Person     owl:equivalentClass [ rdf:type owl:Class ;
                                  owl:unionOf (:Male, :Female ) ] .
```

Rolleneinschränkungen (allValuesFrom)

dienen der Definiton komplexer Klassen durch Rollen.

RDF/XML Syntax

```
<owl:Class rdf:ID="Pruefung">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hatPruefer"/>
            <owl:allValuesFrom rdf:resource="#Professor"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

D.h. alle Prüfer einer Prüfung müssen Professoren sein.

Rolleneinschränkungen (allValuesFrom)

dienen der Definiton komplexer Klassen durch Rollen.

Manchester Syntax

```
Class: Pruefung
  SubClassOf: hatPruefer only Professor
```

D.h. alle Prüfer einer Prüfung müssen Professoren sein.

Rolleneinschränkungen (allValuesFrom)

dienen der Definiton komplexer Klassen durch Rollen.

Turtle Syntax

```
:Pruefung                rdfs:subClassOf              [
rdf:type                 owl:Restriction  ;
owl:onProperty           :hatPruefer  ;
owl:allVauesFrom         :Professor
] .
```

D.h. alle Prüfer einer Prüfung müssen Professoren sein.

Rolleneinschränkungen (someValuesFrom)

RDF/XML Syntax

```
<owl:Class rdf:ID="Pruefung">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hatPruefer"/>
            <owl:someValuesFrom rdf:resource="#Person"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

D.h. jede Prüfung muss mindestens einen Prüfer haben.

AKSW

Rolleneinschränkungen (someValuesFrom)

Manchester Syntax

```
Class: Pruefung
  SubClassOf: hatPruefer some Person
```

D.h. jede Prüfung muss mindestens einen Prüfer haben.

Rolleneinschränkungen (someValuesFrom)

Turtle Syntax

```
:Pruefung                rdfs:subClassOf              [
rdf:type                 owl:Restriction  ;
owl:onProperty           :hatPruefer  ;
owl:someVauesFrom        :Person
] .
```

D.h. jede Prüfung muss mindestens einen Prüfer haben.

Rolleneinschränkungen (Kardinalitäten)

RDF/XML Syntax

```
<owl:Class rdf:ID="Pruefung">
    <rdfs:subClassOf>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#hatPruefer"/>
          <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">
          2
          </owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Eine Prüfung kann *höchstens zwei* Prüfer haben.

AKSW

Rolleneinschränkungen (Kardinalitäten)

Manchester Syntax

```
Class: Pruefung
  SubClassOf: hatPruefer max 2
```

Eine Prüfung kann *höchstens zwei* Prüfer haben.

Rolleneinschränkungen (Kardinalitäten)

Turtle Syntax

```
:Pruefung       rdf:type       owl:Class ;
                rdfs:subClassOf [ rdf:type owl:Restriction ;
                                  owl:onProperty :hatPruefer ;
                                  owl:maxCardinality "2"^^xsd:nonNeg...
                                ] .
```

Eine Prüfung kann *höchstens zwei* Prüfer haben.

Modellierungsbeispiel

Modellierungsaufgabe: Eine Performanceanforderung (an ein Softwareprodukt) wird von einem Kunden erstellt und führt zu einer Systemanforderung.

Modellierungsbeispiel

Modellierungsaufgabe: Eine Performanceanforderung (an ein Softwareprodukt) wird von einem Kunden erstellt und führt zu einer Systemanforderung.

Lösung (Manchester Syntax)

```
Class: PerformanceRequirement
  SubClassOf: Requirement
              and (createdBy only Customer)
              and (leadsTo some SystemRequirement)
```

Rolleneinschränkungen (Kardinalitäten)

RDF/XML Syntax

```
<owl:Class rdf:ID="Pruefung">
    <rdfs:subClassOf>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#hatThema"/>
          <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">
           3
          </owl:minCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Eine Prüfung muss sich über *mindestens drei* Themengebiete erstrecken.

Rolleneinschränkungen (Kardinalitäten)

Manchester Syntax

```
Class: Pruefung
  SubClassOf: hatThema min 3
```

Eine Prüfung muss sich über *mindestens drei* Themengebiete erstrecken.

Rolleneinschränkungen (Kardinalitäten)

Turtle Syntax

```
:Pruefung   rdf:type            owl:Class ;
            rdfs:subClassOf     [ rdf:type owl:Restriction ;
                                  owl:onProperty :hatThema ;
                                  owl:minCardinality "3"^^xsd:nonNegative...
                                ] .
```

Eine Prüfung muss sich über *mindestens drei* Themengebiete erstrecken.

Rolleneinschränkungen (Kardinalitäten)

RDF/XML Syntax

```
<owl:Class rdf:ID="Pruefung">
    <rdfs:subClassOf>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#hatThema"/>
          <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
          3
          </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Eine Prüfung muss sich über *genau drei* Themengebiete erstrecken.

Rolleneinschränkungen (Kardinalitäten)

Manchester Syntax

```
Class: Pruefung
  SubClassOf: hatThema exactly 3
```

Eine Prüfung muss sich über *genau drei* Themengebiete erstrecken.

Rolleneinschränkungen (Kardinalitäten)

Turtle Syntax

```
:Pruefung   rdf:type            owl:Class ;
            rdfs:subClassOf     [ rdf:type owl:Restriction ;
                                  owl:onProperty :hatThema ;
                                  owl:cardinality "3"^^xsd:nonNegative...
                                ] .
```

Eine Prüfung muss sich über *genau drei* Themengebiete erstrecken.

Rolleneinschränkungen (hasValue)

RDF/XML Syntax

```
<owl:Class rdf:ID="PruefungBeiFaehnrich">
    <rdfs:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hatPruefer"/>
            <owl:hasValue rdf:resource="#Faehnrich"/>
        </owl:Restriction>
    </rdfs:equivalentClass>
</owl:Class>
```

Rolleneinschränkungen (hasValue)

Manchester Syntax

```
Class: PruefungBeiFaehnrich
  EquivalentTo: hatPruefer value { Faehnrich }
```

Rolleneinschränkungen (hasValue)

Turtle Syntax

```
:Pruefung          rdfs:equivalentClass      [
    rdf:type           owl:Restriction ;
    owl:onProperty     :hatPruefer ;
    owl:hasValue       :Faehnrich
] .
```

owl:hasValue verweist immer auf ein konkretes Objekt. Dies ist Äquivalent zum Beispiel auf der nächsten Folie.

Rolleneinschränkungen (hasValue)

RDF/XML Syntax

```
<owl:Class rdf:ID="PruefungBeiFaehnrich">
    <rdfs:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hatPruefer"/>
            <owl:someValuesFrom>
              <owl:oneOf rdf:parseType="Collection">
                <owl:Thing rdf:about="#Faehnrich"/>
              </owl:oneOf>
            </owl:someValuesFrom>
        </owl:Restriction>
    </rdfs:equivalentClass>
</owl:Class>
```

Rolleneinschränkungen (hasValue)

Manchester Syntax

```
Class: PruefungBeiFaehnrich
  EquivalentTo: hatPruefer some {Faehnrich}
```

Rolleneinschränkungen (hasValue)

Turtle Syntax

```
:PruefungBeiFaehnrich        rdfs:equivalentClass     [
    rdf:type    owl:Class ;
    owl:oneOf   ( :Faehnrich )
 ] .
```

Gliederung

AKSW

Rollenbeziehungen

owl:subPropertyOf:

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="hatPruefer">
    <rdfs:subPropertyOf rdf:resource="#hatAnwesenden"/>
</owl:ObjectProperty>
```

Ebenso: *owl:equivalentProperty*

Rollen können auch invers (*owl:inverseOf*) zueinander sein:

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="hatPruefer">
    <owl:inverseOf rdf:resource="#prueferVon"/>
</owl:ObjectProperty>
```

Rollenbeziehungen

Manchester Syntax

```
ObjectProperty: hatPruefer
  SubPropertyOf: hatAnwesenden
```

Manchester Syntax

```
ObjectProperty: hatPruefer
  InverseOf: prueferVon
```

Turtle Syntax

```
:hatPruefer    rdf:type            owl:ObjectProperty ;
               rdfs:subPropertyOf  :hatAnwesenden .
```

Turtle Syntax

```
:hatPruefer    rdf:type            owl:ObjectProperty ;
               owl:inverseOf       :prueferVon .
```

Rolleneigenschaften

- Domain
- Range
- Transitivität, d.h. r(a,b) und r(b,c) impliziert r(a,c)
- Symmetrie, d.h. r(a,b) impliziert r(b,a)
- Funktionalität r(a,b) und r(a,c) impliziert b=c
- Inverse Funktionalität r(a,b) und r(c,b) impliziert a=c

AKSW

Domain und Range

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="zugehoerigkeit">
    <rdfs:range rdf:resource="#Organisation"/>
</owl:ObjectProperty>
```

ist gleichbedeutend mit dem Folgenden:

RDF/XML Syntax

```
<owl:Class rdf:about="&owl;Thing">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#zugehoerigkeit"/>
            <owl:allValuesFrom rdf:resource="#Organisation"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Domain und Range

Manchester Syntax

```
ObjectProperty: zugehoerigheit
  Range: Organisation
```

ist gleichbedeutend mit dem Folgenden:

Manchester Syntax

```
Class: owl:Thing
  SubClassOf: zugehoerigkeit only Organisation
```

Domain und Range

Turtle Syntax

```
:zugehoerigkeit    rdf:type        owl:ObjectProperty  ;
                   rdfs:range      :Organisation  .
```

ist gleichbedeutend mit dem Folgenden:

Turtle Syntax

```
owl:Thing                  rdfs:subClassOf            [
     rdf:type                owl:Restriction  ;
     owl:onProperty          :zugehoerigkeit  ;
     owl:allValuesFrom       :Organisation
] .
```

Domain und Range: Vorsicht!

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="zugehoerigkeit">
    <rdfs:range rdf:resource="#Organisation"/>
</owl:ObjectProperty>

<Zahl rdf:ID="Fuenf">
    <zugehoerigkeit rdf:resource="#Primzahlen"/>
</Zahl>
```

Es folgt nun, dass *Primzahlen* eine Organisation ist!

Domain und Range: Vorsicht!

Manchester Syntax

```
ObjectProperty: zugehoerigkeit
  Range: Organisation
Individual: Fuenf
  Types: Zahl
  Facts: zugehoerigkeit Primzahlen
```

Es folgt nun, dass *Primzahlen* eine Organisation ist!

AKSW

Domain und Range: Vorsicht!

Turtle Syntax

```
:zugehoerigkeit     rdf:type            owl:ObjectProperty  ;
                    rdfs:range          :Organisation  .
:Fuenf              rdf:type            :Zahl  ;
                    :zugehoerigkeit     :Primzahlen  .
```

Es folgt nun, dass *Primzahlen* eine Organisation ist!

Rolleneigenschaften

RDF/XML Syntax

```
<owl:ObjectProperty rdf:ID="hatKollegen">
    <rdf:type rdf:resource="&owl;TransitiveProperty"/>
    <rdf:type rdf:resource="&owl;SymmetricProperty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hatProjektleiter">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="istProjektleiterFuer">
    <rdf:type rdf:resource="&owl;InverseFunctionalProperty"/>
</owl:ObjectProperty>
<Person rdf:ID="SoerenAuer">
    <hatKollegen rdf:resource="#SebastianTramp"/>
    <hatKollegen rdf:resource="#JensLehmann"/>
    <istProjektleiterFuer rdf:resource="#Triplify"/>
</Person>
<Projekt rdf:ID="OntoWiki">
    <hatProjektleiter rdf:resource="#SoerenAuer"/>
    <hatProjektleiter rdf:resource="#AuerSoeren"/>
</Projekt>
```

Folgerungen aus dem Beispiel

- SebastianTramp hatKollegen SoerenAuer
- SebastianTramp hatKollegen JensLehmann
- SoerenAuer owl:sameAs AuerSoeren

AKSW

Gliederung

AKSW

OWL Varianten

- OWL Full
  - Enthält OWL DL und OWL Lite
  - Enthält als einzige OWL-Teilsprache ganz RDFS
  - Semantik enthält einige Aspekte, die aus logischem Blickwinkel problematisch sind.
  - Unentscheidbar.
  - Wird durch aktuelle Softwarewerkzeuge nur bedingt unterstützt.
- OWL DL
  - Enthält OWL Lite und ist Teilsprache von OWL Full.
  - Entscheidbar.
  - Wird von aktuellen Softwarewerkzeugen fast vollständig unterstützt.
  - Komplexität NExpTime (worst-case).
- OWL Lite
  - Ist Teilsprache von OWL DL und OWL Full.
  - Entscheidbar.
  - Wenig ausdrucksstark.
  - Komplexität ExpTime (worst-case).

AKSW

OWL Full

- Uneingeschränkte Nutzung aller OWL und RDFS-Sprachelemente (muss gültiges RDFS sein):
- Schwierig z.B.: nicht vorhandene Typentrennung (Klassen,Rollen, Individuen), dadurch:
  - *owl:Thing* dasselbe wie *rdfs:resource*
  - *owl:Class* dasselbe wie *rdfs:Class*
  - *owl:DatatypeProperty* Subklasse von *owl:ObjectProperty*
  - *owl:ObjectProperty* dasselbe wie *rdf:Property*

AKSW

Beispiel für Typendurchmischung in OWL Full

Verknüprfung von Klasse und Objekt über eine Property:

```
<owl:Class rdf:about="\#Buch">
    <englischerName rdf:datatype="\&xsd;string">
        book
    </englischerName>
    <franzoesischerName rdf:datatype="\&xsd;string">
        livre
    </franzoesischerName>
</owl:Class>
```

Inferenzen über solche Konstrukte werden oft nicht wirklich benötigt.

OWL DL

- Nur Verwendung von explizit erlaubten RDFS Sprachelementen (z.B. die in unseren Beispielen). Nicht erlaubt: rdfs:Class, rdfs:Property
- Typentrennung. Klassen und Rollen müssen explizit deklariert werden.
- Konkrete Rollen dürfen nicht als transitiv, symetrisch, invers oder invers funktional deklariert werden.
- Zahlenrestriktionen dürfen nicht mit transitiven Rollen, deren Subrollen, oder Inversen davon verwendet werden.

AKSW

OWL Lite

- alle Einschränkungen für OWL DL
- außerdem:
  - nicht erlaubt: *oneOf, unionOf, complementOf, hasValue, disjointWith*
  - Zahlenrestriktionen nur mit 0 und 1 erlaubt.
  - Einige Einschränkungen zum Auftreten von anonymen (komplexen) Klassen, z.B. nur im Subjekt von *rdfs:subClassOf*.

AKSW

Gliederung

OWL Werkzeuge

- Editoren
  - Protegé, `http:protege.stanford.edu`
  - OntoWiki, `http://ontowiki.net/`
  - SWOOP, `http://www.mindswap.org/2004/SWOOP/`
  - OWL Tools, `http://owltools.ontoware.org/`
- Inferenzmaschinen
  - Pellet, `http://clarkparsia.com/pellet/`
  - KAON2, `http://kaon2.semanticweb.org`
  - FACT++, `http://owl.man.ac.uk/factplusplus/`
  - Racer, `http://www.racer-systems.com/`
  - HermIT, `http://www.hermit-reasoner.com/`

AKSW

OWL Sprachelemente

Kopf

rdfs:comment
rdfs:label
rdfs:seeAlso
rdfs:isDefinedBy
owl:versionInfo
owl:priorVersion
owl:backwardCompatibleWith
owl:incompatibleWith
owl:DeprecatedClass
owl:DeprecatedProperty
owl:imports

Beziehungen zwischen Individuen

owl:sameAs
owl:differentFrom
owl:AllDifferent
(zusammen mit
owl:distinctMembers)

Vorgeschriebene Datentypen

xsd:string
xsd:integer

AKSW

OWL Sprachelemente

Klassenkonstruktoren und -beziehungen

owl:Class
owl:Thing
owl:Nothing
rdfs:subClassOf
owl:disjointWith
owl:equivalentClass
owl:intersectionOf
owl:unionOf
owl:complementOf

Rollenrestriktionen

owl:allValuesFrom
owl:someValuesFrom
owl:hasValue
owl:cardinality
owl:minCardinality
owl:maxCardinality
owl:oneOf

OWL Sprachelemente

Rollenkonstruktoren, -beziehungen und -eigenschaften:

owl:ObjectProperty
owl:DatatypeProperty
rdfs:subPropertyOf
owl:equivalentProperty
owl:inverseOf
rdfs:domain
rdfs:range
owl:TransitiveProperty
owl:SymmetricProperty
owl:FunctionalProperty
owl:InverseFunctionalProperty

Weiterführende Literatur

- `http://www.w3.org/2004/OWL/` zentrale W3C Webseite für OWL.
- `http://www.w3.org/TR/owl-features/` Überblick über OWL.
- `http://www.w3.org/TR/owl-ref/` vollständige Beschreibung der OWL-Sprachkomponenten.
- `http://www.w3.org/TR/owl-guide/` zeigt, wie OWL zur Wissensmodellierung verwendet werden kann.
- `http://www.w3.org/TR/owl-semantics/` beschreibt die Semantik von OWL, die wir auf andere Weise später behandeln werden.
- Deutsche Übersetzungen mancher W3C Dokumente findet man unter `http://www.w3.org/2005/11/Translations/Lists/ListLang-de.html`

Danke für die Aufmerksamkeit!

AKSW