

Vorlesung Netzbasierte Informationssysteme

Semantic Web II

Prof. Dr. Adrian Paschke

Arbeitsgruppe Corporate Semantic Web (AG-CSW)

Institut für Informatik, Freie Universität Berlin

paschke@inf.fu-berlin.de

<http://www.inf.fu-berlin.de/groups/ag-csw/>



Semantic Web

- Semantic Web – Ein Einführung (Teil 1)
 - Semantic Web Vision
 - RDF
 - RDFS
 - RDF Anfragesprachen / SPARQL
- Semantic Web – Ontologie (Teil 2)
 - Was ist eine Ontologie
 - W3C Web Ontology Language (OWL)
 - OWL 1.1 und OWL 2
- Semantic Web – Regeln und Erweiterte Konzepte (Teil 3)

Was ist eine Ontologie?

Aristoteles - Ontologie

- Davor: **Studie der Natur des Seins**
- Seit Aristoteles: **Studie der Wissensrepräsentation und Schlussfolgerung**
- Begrifflichkeit:
 - Genus: (Klassen)
 - Spezies: (Subklassen)
 - Differentiae: (Charakteristiken welches die Gruppierung oder Unterscheidung von Objekten erlaubt)
- Syllogismus (Inferenzregeln)



Was ist eine Ontologie?

- People can't share knowledge if they do not speak a common language (Davenport, Prusak, 98)
- An ontology is a formal specification of a shared conceptualization of a domain of interest (Gruber, 93)
- An ontology is a document or file that formally defines the relations among terms (Berners-Lee, 01)
- Ontologien standardisieren und formalisieren die Bedeutung von Wörtern in Form von Konzepten.
- Alles (Personen, Projekte ...) kann als Ressource angesehen werden, die über Beziehungen mit anderen Ressourcen verbunden ist.

Was ist eine Ontologie?

An Ontology is a

formal specification

→ Ausführbar, Diskutierbar

of a shared

→ Gruppe von Personen

conceptualization

→ Über Konzepte; abstrakte Klassen

of a domain of interest

→ z.b. eine Anwendung, eine spezifisches Gebiet, das "Weltmodell"

[Gruber 1993] - T.R. Gruber, Toward Principles for the Design of Ontologies Used for Knowledge Sharing, Formal Analysis in Conceptual Analysis and Knowledge Representation, Kluwer, 1993.

Konzept vs Instanz

- **Konzept** Concept / Class / Universal (Metaphysics)

Person

- Eine abstrakte oder generelle Idee gefolgert oder abgeleitet aus spezifischen Instanzen

- **Instanz** / Instance / Individual / Particular (Metaphysics)

Person

Name: Adrian Paschke

Position: Professor

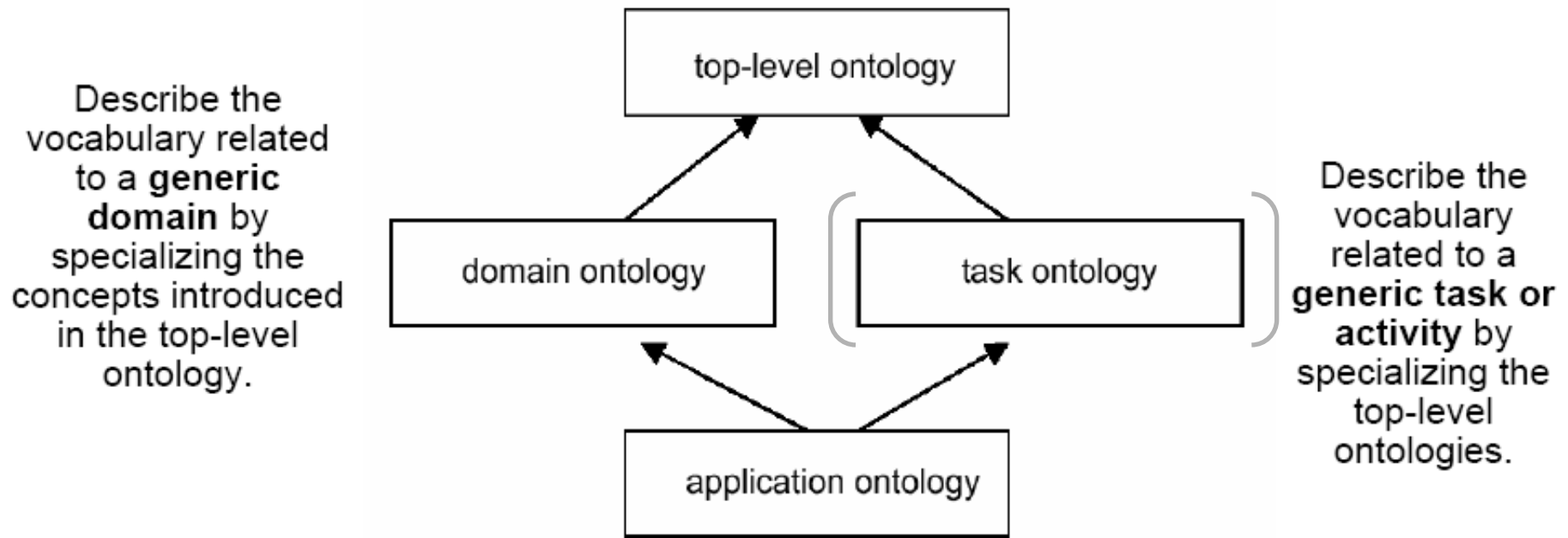
LebtIn: Berlin

ArbeitetBei: FU Berlin

- Objekt in der Realität, eine Ausprägung eines abstrakten Konzepts mit realen Werten der Eigenschaften

Typen von Ontologien

Describe **very general concepts** like space, time, event, which are independent of a particular problem or domain. It seems reasonable to have unified top-level ontologies for large communities of users.



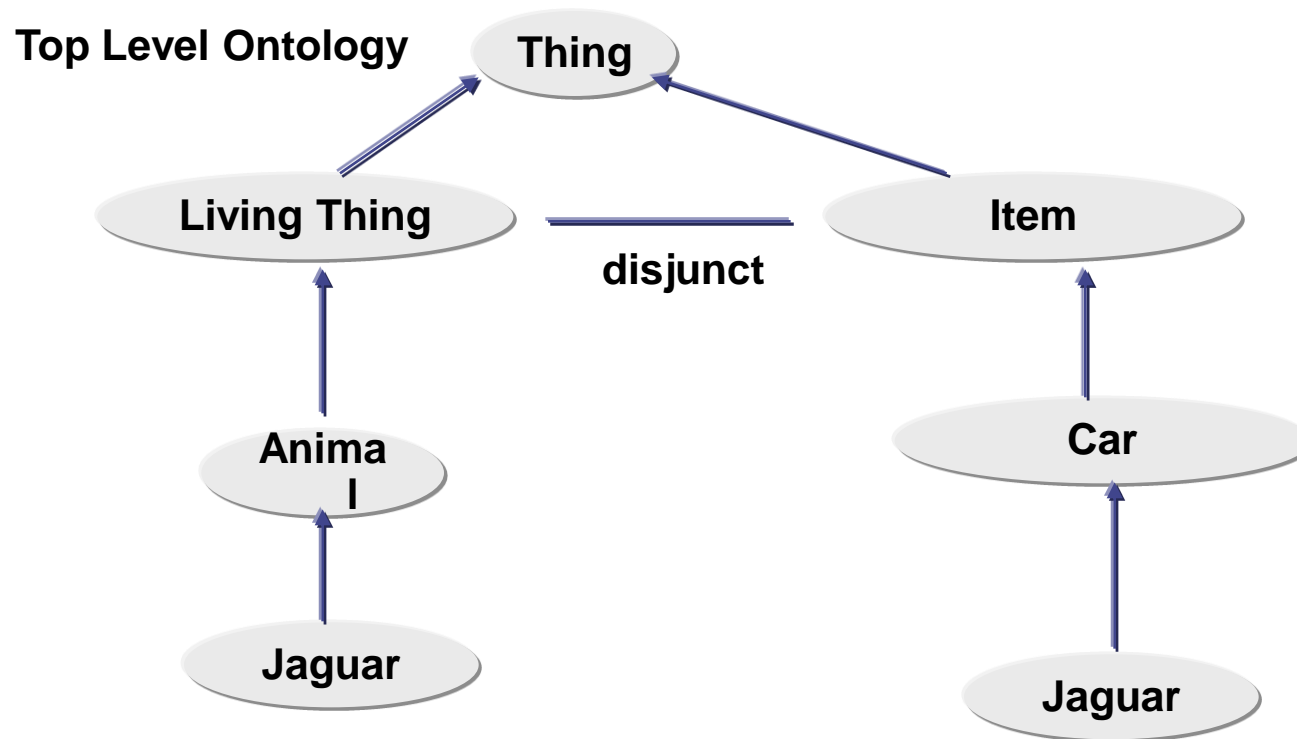
These are the most specific ontologies. Concepts in application ontologies often correspond to **roles played by domain entities while performing a certain activity**.



Existierende Ontologien

- General purpose ontologies:
 - WordNet, <http://www.cogsci.princeton>. Semantische Lexikon für Englische Sprache
 - EuroWordNet – Multilinguale Datenbank mit Wordnets mit Europäischen Sprachen
 - GermaNet - Semantisches Lexikon für Deutsche Sprache
- Upper level ontologies: e.g. Zeit, Ort
 - Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)
 - Upper-Cyc Ontology, <http://www.cyc.com/> (300.000+ Konzepte, 3.000.000 Fakten und Regeln basierend auf 26000+ Relationen)
 - IEEE Standard Upper Ontology, <http://suo.ieee.org>
- Domänen- und Anwendungs-spezifische Ontologien:
 - RDF Site Summary RSS, <http://groups.yahoo.com/group/rss-dev/files/schema.rdf>
 - Unified Medical Language System (UMLS), <http://www.nlm.nih.gov/research/>
 - Gene Ontology <http://www.geneontology.org>

Beispiel



Example: Unified Modelling Language (UMLS)

- “Verstehen” der Bedeutung der Sprachen in der Biomedizin und Gesundheitswesen
- Benutzt für z.B.:
 - Patient records*
 - Scientific literature*
 - Guidelines*
 - Public health data*
- Gepflegt durch:

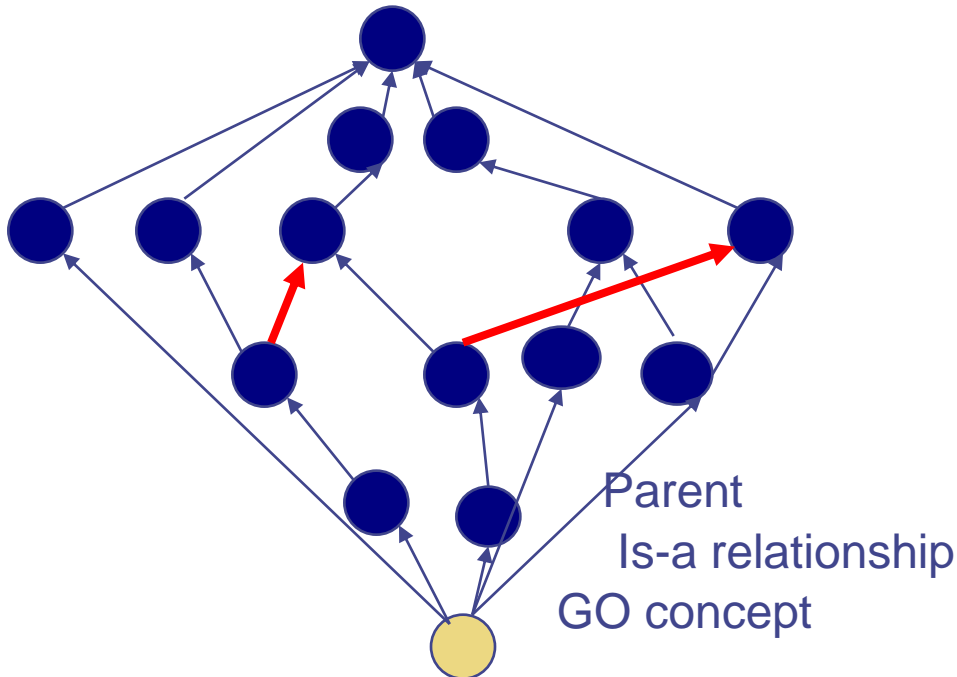


United States
National Library of Medicine
National Institutes of Health



GeneOntology

- Proteinsequenzen und Strukturen ausgezeichnet mit der GeneOntology
- ca. 19.000 Terme
 - Cellular Components
 - Biological Process
 - Molecular Function



Gene Ontology Consortium - Mozilla

GENE ONTOLOGY CONSORTIUM

[What is the Gene Ontology?](#) [Download the Ontologies](#)

The goal of the Gene Ontology™ (GO) Consortium is to produce a controlled vocabulary that can be applied to all organisms even as knowledge of gene and protein roles in cells is accumulating and changing. GO provides three structured [networks](#) of defined terms to describe gene product attributes. GO is one of the controlled vocabularies of the [Open Biological Ontologies](#).

- Submit new GO term suggestions via the [Curator Requests Tracker](#) at [SourceForge](#). [Help with new term submission](#) is available.
- Send comments and questions to go@geneontology.org.

Search Terms and Annotations

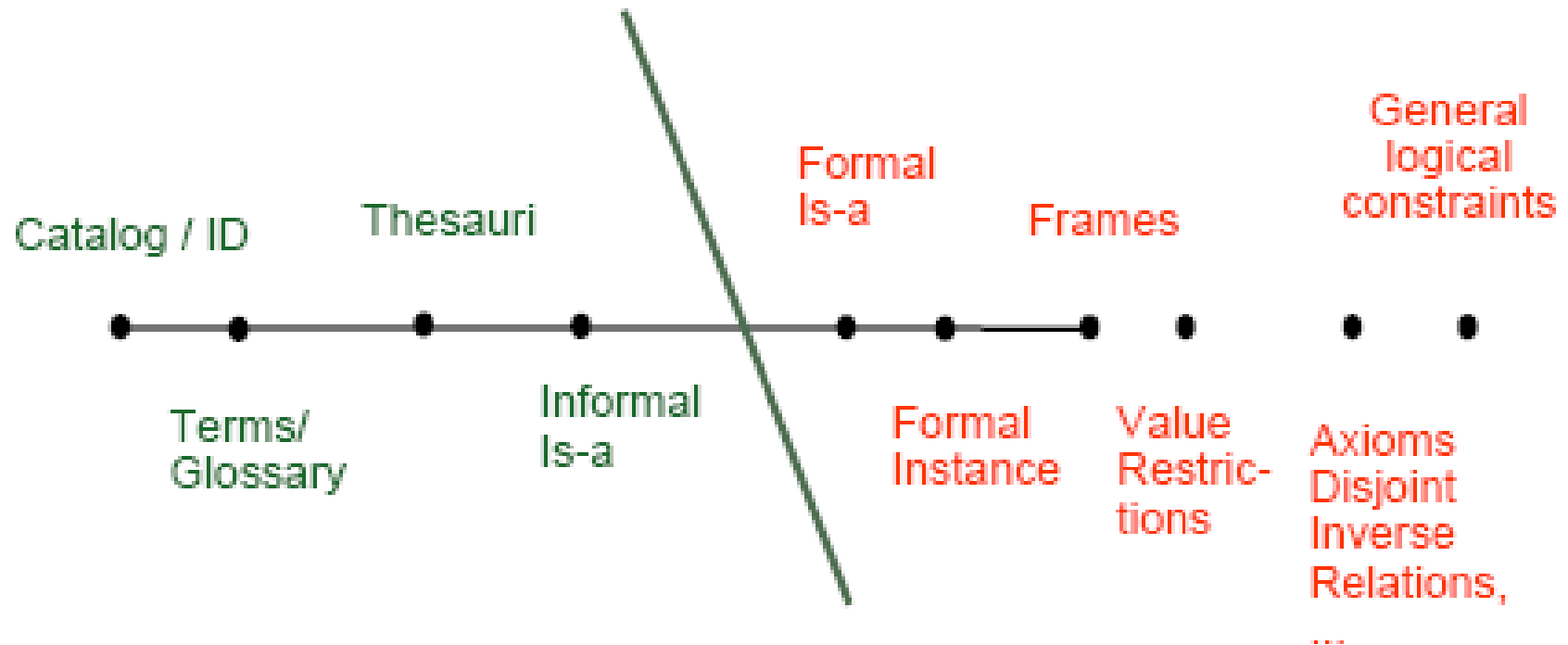
This search uses the [AmiGO](#) browser. You can also use one of the many other [GO Browsers](#)

What's New?

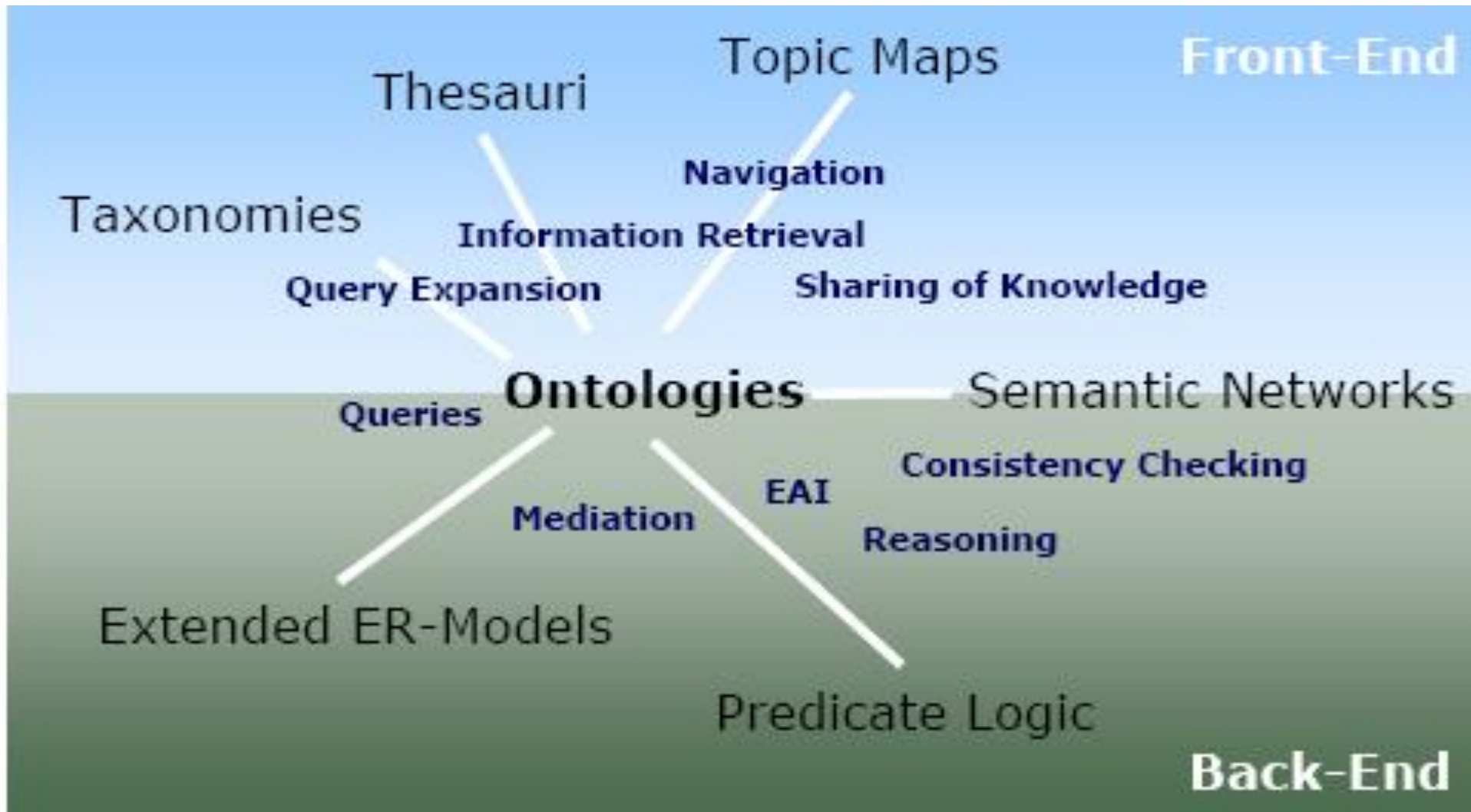
- We are pleased to announce that the four currently maintained GO slims, those for plant, yeast, GOA and the generic GO slim, have now been incorporated into the OBO-format flat file. These GO slims will continue to be supplied in the old format for those who have not yet converted to the new OBO format. (posted July 21, 2004)
- A mapping of [Reactome](#) biological processes to GO terms has been added to the [external2go directory](#). (posted July 12, 2004)
- We are pleased to announce the next **Gene Ontology Users Meeting on Thursday, October 14, 2004** at Northwestern University's Chicago campus. The deadline for abstract submission is September 17, 2004, and the deadline for registration is October 1, 2004. Please visit the [meeting website](#) for more information. (posted June 10, 2004)
- A mapping of [COG \(Clusters of Orthologous Groups\)](#) functional categories to GO terms has been added to the [external2go directory](#). (posted June 10, 2004)

[What's new archive](#) (plain text document)

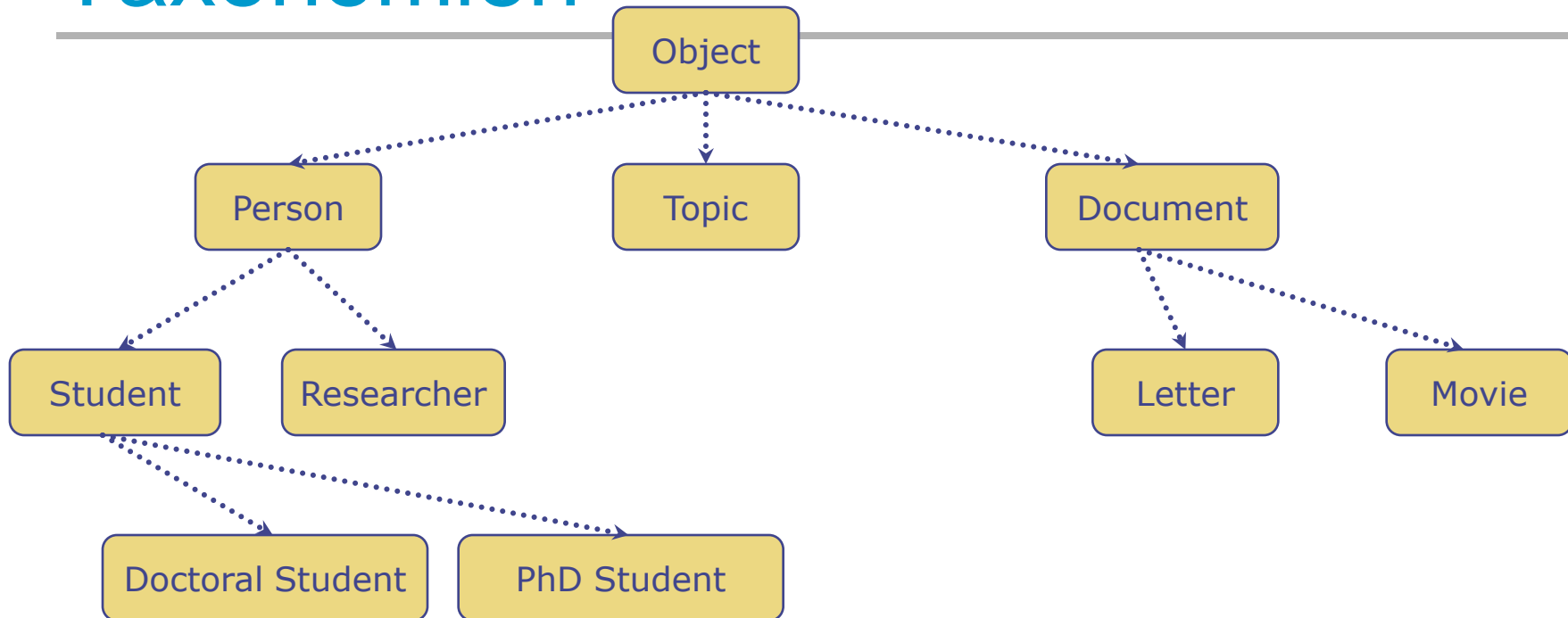
Überblick: Wie formal sind Ontologien?



Überblick: Benutzung von Ontologien

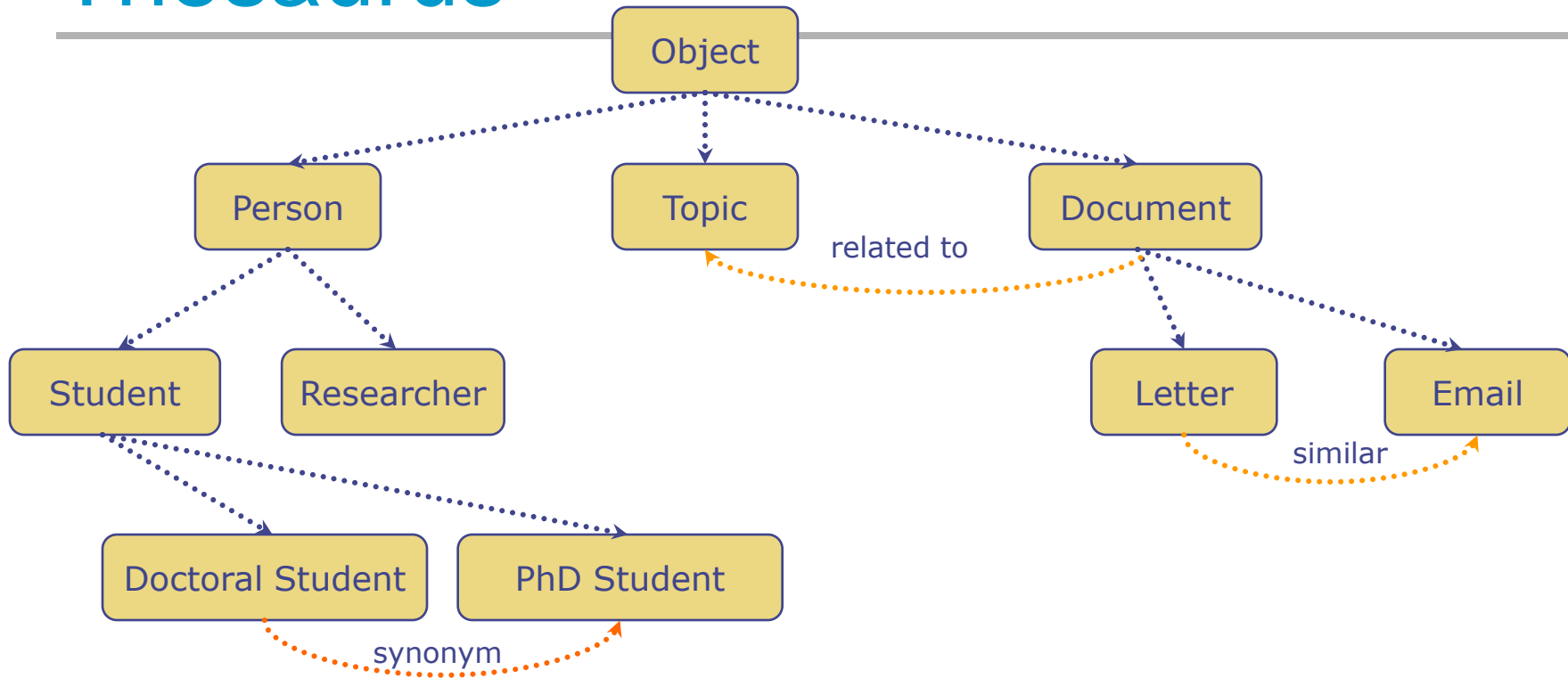


Taxonomien



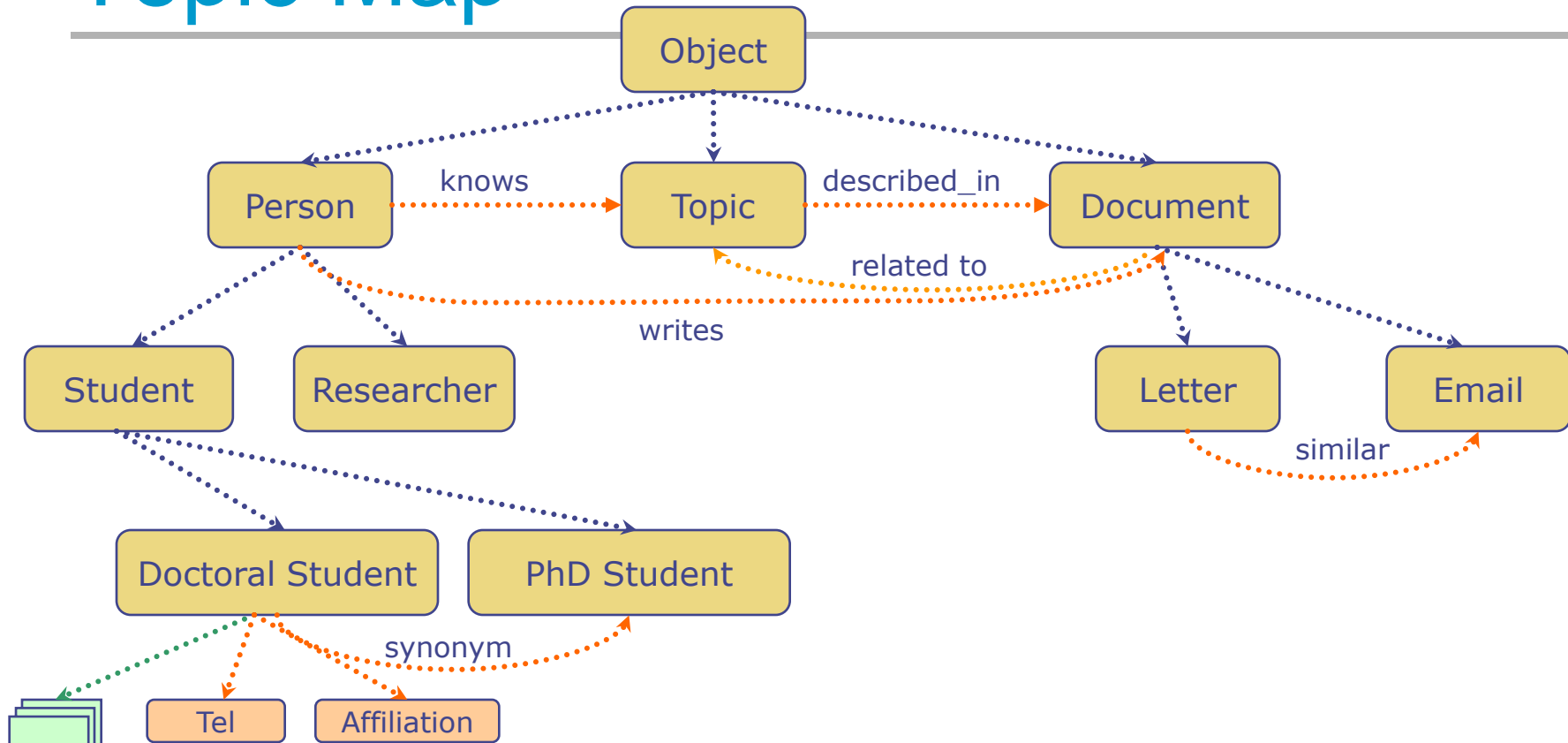
- Taxonomien := Segmentierung, Klassifizierung und Ordnung von Elementen in ein Klassifikationssystem entsprechend der Subklassenbeziehungen

Thesaurus



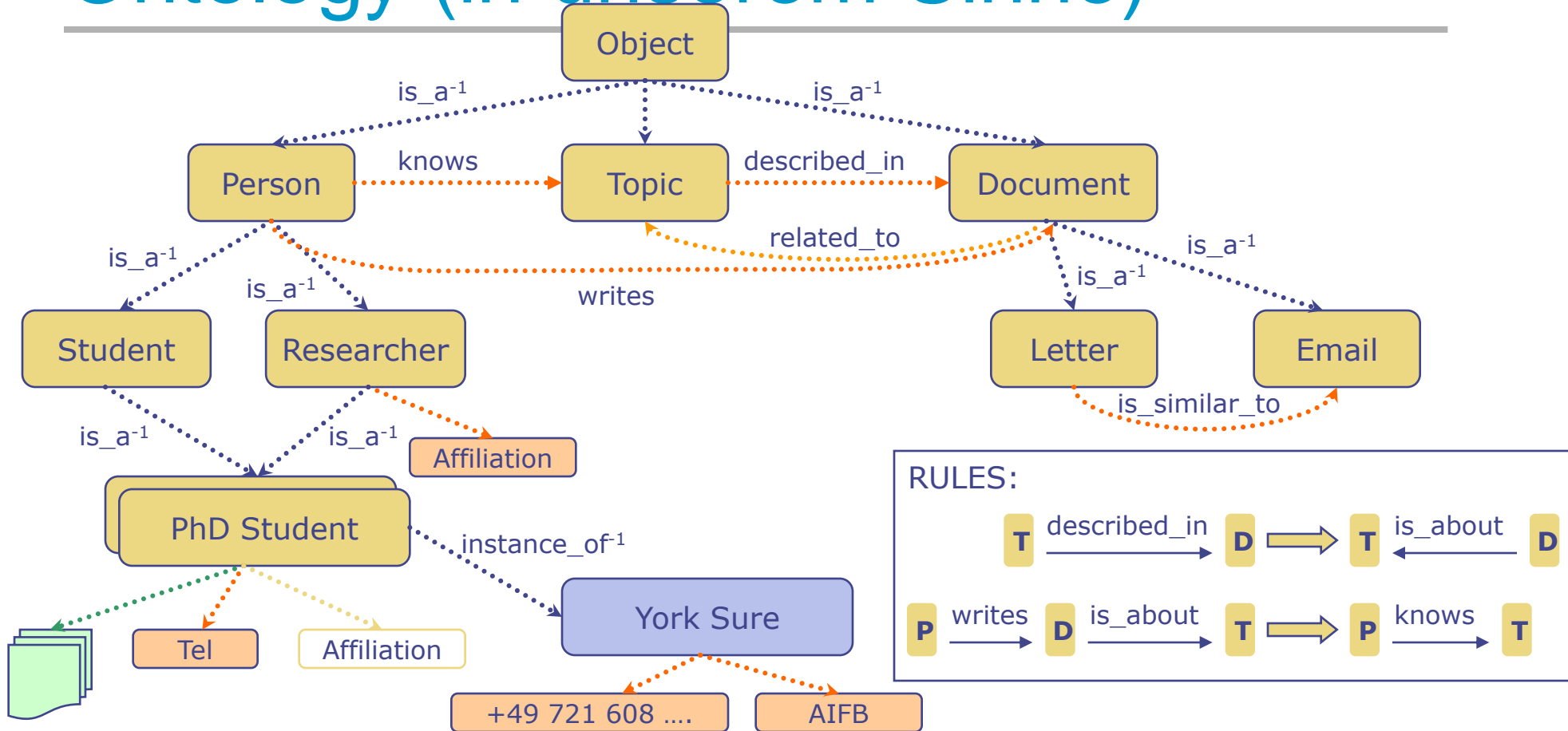
- Terminologie für eine spezifische Domäne
- Taxonomie plus feste Beziehungen (z.B. similar, synonym, related to)
- Kommt aus der Bibliographie

Topic Map



- Topics (Knoten), Beziehungen, und Vorkommen von Dokumenten
- ISO-Standard
- Typischerweise für die Navigation und Visualisierung

Ontology (in unserem Sinne)



- Repräsentationssprachen: RDF(S); OWL; Prädikatenlogik; Frame-Logik (F-Logic)

Ontologiesprachen

- Ontologiesprachen erlauben die Beschreibung einer expliziten, formalen Konzeptualisierung eines Domänenmodells
- Anforderungen:
 - Wohl-definierte Syntax
 - Formale Semantik
 - Unterstützung von automatischen Schlussfolgerungen
 - Adequate Ausdruckstärke für die Domäne
 - Benutzbarkeit der Ausdrücke

Ontologiebeschreibungssprachen

- Entity Relationship Modell
 - UML mit OCL
 - Frame Based Systems
 - Prädikatenlogik
 - Description Logic (formale Semantik, Reasoning)
-
- XML-basierte: SHOE, XOL, OML, **RDFS**, DAML+OIL, **OWL**

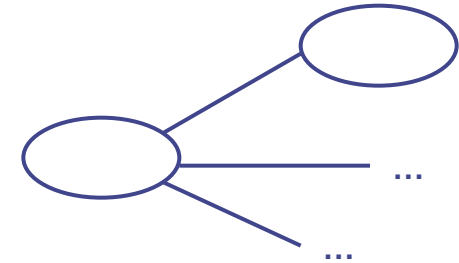
Keine speziellen Ontologiebeschreibungssprachen. Können aber in gewissem Umfang dafür genutzt werden.

Tradeoff zwischen Ausdrucksstärke und Schlussfolgerungsmächtigkeit

- Je ausdrucksmächtiger die Sprache ist, um so ineffizienter wird das Schlussfolgern
- Manchmal wird die Grenze zur **Nicht-Berechenbarkeit** überschritten
- Es wird ein Kompromiss benötigt:
 - Eine Sprache, die durch effiziente Schlussfolgerer (reasoner) unterstützt wird
 - Eine Sprache, die mächtige Ontologieklassen und Wissen ausdrücken kann

Exkurs: Description Logics

- DL = Description Logics = Beschreibungslogik
 - Entscheidbares Subset der Prädikatenlogik (first order logic)
 - Unzählige Varianten von DLs
- Mit DL lassen sich Graphen/Netze beschreiben
 - Konzept (\approx Knoten)
 - Rolle (\approx Kante)
- DL Inferenzsysteme
 - aka DL-Theorembeweiser, DL-Reasoner
 - erlauben Schlüsse aus explizit vorhandenem Wissen abzuleiten
 - Ist A ein Superkonzept von B ? (Subsumption)
 - Is A äquivalent mit B? (Equivalence Checking)
 - Ist a eine Instanz von Konzept A? (Instance Checking)
- **Formale Semantik für OWL**
 - OWL Lite = SHIQ(D)
 - OWL DL = SHOIN(D)



Schlussfolgerung über Wissen in Ontologiesprachen

- Klassenzugehörigkeit
 - “If x is an instance of a class C , and C is a subclass of D , then we can infer that x is an instance of D ”
- Äquivalenz von Klassen
 - “If class A is equivalent to class B , and class B is equivalent to class C , then A is equivalent to C , too”

Schlussfolgerung über Wissen in Ontologiesprachen (2)

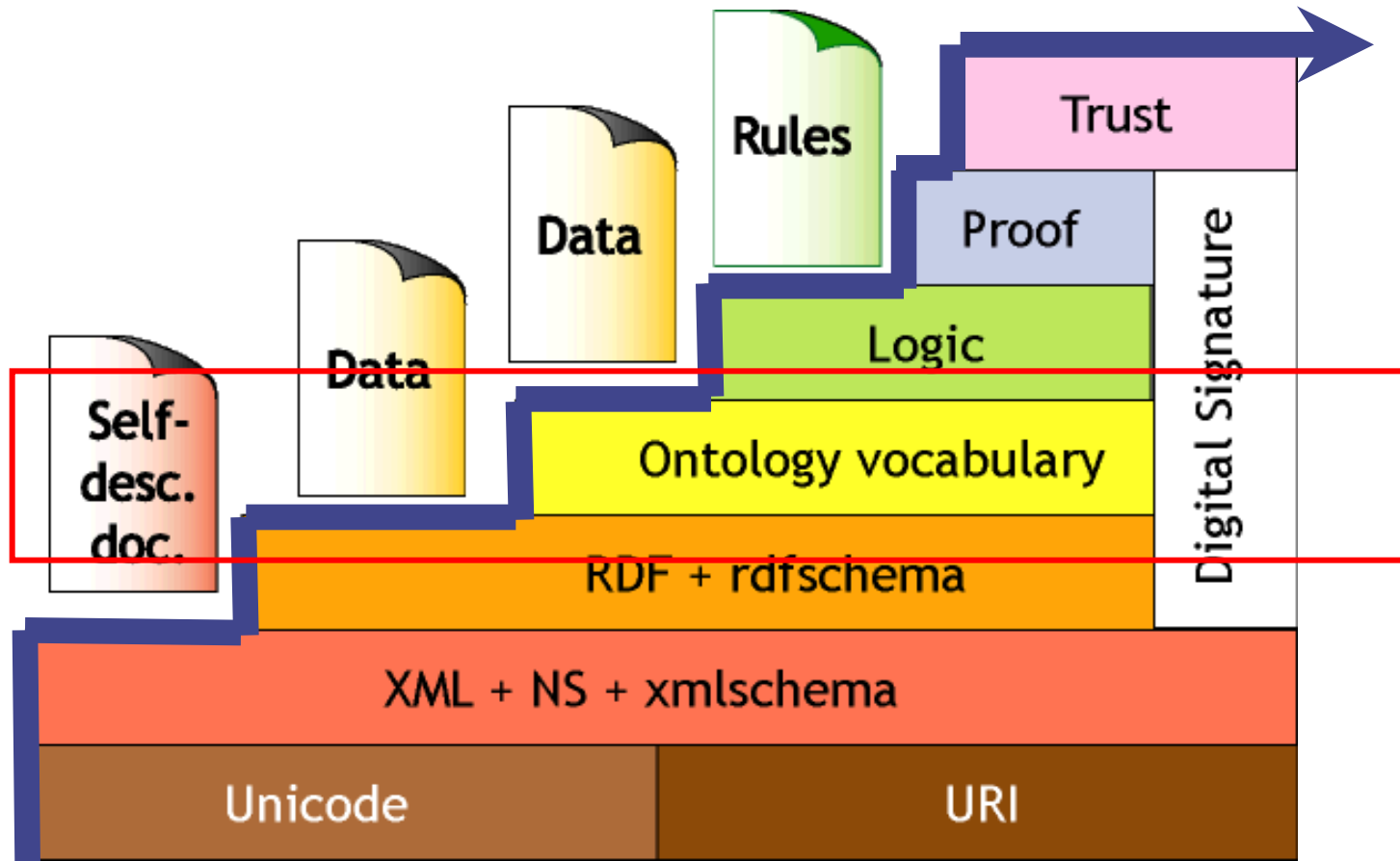
- **Konsistenz**
 - “X instance of classes A and B, but A and B are disjoint”
 - Dies ist ein Indikator für einen Fehler in der Ontologie
- **Klassifikation**
 - Bestimmte Eigenschaft-Werte Pare sind eine ausreichende Kondition für die Zugehörigkeit in eine Klasse A, wenn ein Individuum X dieses erfüllt; dann kann geschlussfolgert werden, dass x eine Instanz von A ist.

Nutzen der Schlussfolgerung

- Schlussfolgerung (Reasoning) ist wichtig für
 - Überprüfung der Konsistenz einer Ontologie und des Wissens
 - Überprüfung der nicht gewollten Relationen zwischen Klassen
 - Automatische Klassifizierung von Instanzen zu Klassen
- Nutzen für
 - Entwicklung großer Ontologien, mit vielen Autoren
 - Integration und gemeinsame Benutzung von Ontologien aus mehreren Quellen

W3C OWL
Web Ontology Language

Semantic Web



DAML+OIL

- RDFS reicht nicht, um komplexere Ontologien zu definieren
- DAML+OIL wurde vom W3C-Konsortium im Jahr 2000 veröffentlicht
- Symbiose aus dem amerikanischen Standard “DARPA Agent Markup Language” (*DAML*) und dem europäischen Standard “Ontology Interface Layer” (*OIL*)



- DAML wurde unter Leitung des “*Defense Advanced Research Projects Agency*” im Auftrag des “*Department Of Defense*” (DoD) entwickelt
- Aktueller Standard, wurde von OWL abgelöst (W3C Recommendation, 10 Feb 2004).

OWL - Web Ontology Language

- Schaffung eines Ontologie-Sprach-Standards unter Verwendung der bestehenden Standards XML, RDF, RDF-Schema
- Sprache soll Konstrukte beinhalten, um Definitions- und Wertebereiche sowie **Kardinalitäten** spezifizieren zu können
- Sprache soll Mechanismen beinhalten, so dass **Klassen von einander abgegrenzt** und zueinander in Beziehung gesetzt werden können
- Die Sprache soll von Maschinen lesbar / interpretierbar sein, so dass Anwendungen darauf aufbauen können



OWL (Web Ontology Language)

- OWL bietet zusätzliche Möglichkeiten zu RDFS
 - Definition von Beziehungen zwischen Klassen
 - Definition von Einschränkungen und Kardinalitäten
 - Einschränkungen von Eigenschaften: exists, forall, cardinality
 - Definition von Äquivalenzen zwischen Klassen (unterschiedlicher Ontologien)
 - Eigenschaften von Eigenschaften
 - Boolesche Kombination von Klassen und Einschränkungen
 - ...

Limitationen der Ausdruckstärke von RDF Schema

- Lokaler Zuständigkeitsbereich (scope) der Eigenschaften
 - **rdfs:range** definiert den Wertebereich einer Eigenschaft (z.B. isst)
 - In RDF Schema können keine Restriktionen auf den Wertebereich (range restrictions) deklariert werden, die nur auf einige Klassen anwendbar sind
 - z.B. es kann nicht ausgedrückt werden, dass “Kühe nur pflanzen essen, während andere Tiere auch Fleisch essen

Limitationen der Ausdruckstärke von RDF Schema (2)

- Disjunkte Klassen
 - z.B. **männlich** und **weiblich** als disjunkte Klassen
- Boolesche Kombinationen von Klassen
 - Bildung neuer Klassen durch Kombination von anderen Klassen (union, intersection, complement)
 - z.B. Eine **Person** ist die disjunkte Union der Klasse “**männlich**” and “**weiblich**”

Limitationen der Ausdruckskraft von RDF Schema (3)

- Kardinalitätsrestriktionen
 - z.B. eine Person hat genau zwei Eltern
- Spezielle Charakteristiken von Eigenschaften
 - Transitive property (z.B. “greater than”)
 - Unique property (z.B. “is mother of”)
 - Inverse property (z.B. “eats” und “is eaten by”)

Kombination von OWL mit RDF Schema

- Idealerweise würde OWL RDF Schema erweitern
 - Konsistent mit der Schichtenarchitektur des Semantic Webs
- **Aber** einfache Erweiterung von RDF Schema würde gegen das Ziel hoher Ausdrucksstärke und effiziente Schlussfolgerung sein
 - Kombination von RDF Schema mit mächtigen Logikkonzepten führt zu unkontrollierbaren rechenintensiven Eigenschaften

Varianten von OWL 1.0

- OWL Lite:
 - Ziel: Einfachheit für Tool-Hersteller
 - Beinhaltet die wichtigsten Merkmale von OWL und DAML+OIL
 - Bietet nützliche Erweiterungen zu RDFS
 - Bietet Klassenhierarchien und einfache Einschränkungen
 - Z.B. Kardinalitätsbeschränkungen
- OWL DL (*Description Logic*):
 - Bietet das vollständige OWL Vokabular
 - Klassen können nicht Eigenschaften oder Individuen sein
 - Eigenschaften können keine Individuen sein
 - Äquivalenzen zwischen Klassen
- OWL Full:
 - Volles OWL Vokabular

Kompatibilität zwischen den OWL Varianten

- Jede legale OWL Lite Ontologie ist eine OWL DL Ontologie
- Jede legale OWL DL Ontologie ist eine legale OWL Full Ontologie
- Jede valide OWL Lite Schlussfolgerung ist eine valide OWL DL Schlussfolgerung
- Jede valide OWL DL Schlussfolgerung ist eine valid OWL Full Schlussfolgerung

OWL Syntaktische Varianten

- OWL baut auf RDF auf und benutzt RDF's XML-Syntax
- Andere syntaktische Formen für OWL wurden auch definiert:
 - Eine alternative, besser lesbare XML-basierte Syntax
 - Eine abstrakte Syntax, die kompakter und lesbarer ist, als die XML Sprachen
 - Eine graphische Syntax basierend auf den Konventionen von UML

OWL XML/RDF Syntax: Header

<rdf:RDF

xmlns:owl = "http://www.w3.org/2002/07/owl#"

xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"

xmlns:xsd = "http://www.w3.org/2001/XMLSchema#">

- Eine OWL Ontologie kann mit einer Sammlung an Erklärungen (assertions) starten, definiert im **owl:Ontology** Element

owl:Ontology

```
<owl:Ontology rdf:about="">  
  <rdfs:comment>An example OWL ontology </rdfs:comment>  
  <owl:priorVersion  
    rdf:resource="http://www.mydomain.org/uni-ns-old"/>  
  <owl:imports  
    rdf:resource="http://www.mydomain.org/persons"/>  
  <rdfs:label>University Ontology</rdfs:label>  
</owl:Ontology>
```

- **owl:imports** ist eine transitive Eigenschaft

Klassen

- Klassen werden durch **owl:Class** definiert
 - **owl:Class** ist eine Subklasse von **rdfs:Class**
- Disjunktheit ist definiert durch **owl:disjointWith**

```
<owl:Class rdf:about="#associateProfessor">  
  <owl:disjointWith rdf:resource="#professor"/>  
  <owl:disjointWith  
    rdf:resource="#assistantProfessor"/>  
</owl:Class>
```

Klassen (2)

- **owl:equivalentClass** definiert äquivalente Klassen

```
<owl:Class rdf:ID="faculty">
```

```
  <owl:equivalentClass rdf:resource=
    "#academicStaffMember"/>
```

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

- **owl:Thing** ist die generellste Klasse, welche alles enthält
- **owl:Nothing** ist die leere Klasse

Eigenschaften

- In OWL gibt es zwei Arten von Eigenschaften (properties)
 - **Object properties**, welche Objekte zu Objekten in Beziehung setzen
 - z.B. wird-gelehrt-von
 - **Data type properties**, welche Objekte zu Datentypwerten in Beziehung setzen
 - z.B. Telefon, Titel, Alter, etc.

Datentype Eigenschaften

- OWL benutzt XML Schema Datentypen

```
<owl:DatatypeProperty rdf:ID="age">  
  <rdfs:range rdf:resource=  
    "http://www.w3.org/2001/XMLSchema  
    #nonNegativeInteger"/>  
</owl:DatatypeProperty>
```

Objekt Eigenschaften

- Benutzerdefinierte Datentypen

```
<owl:ObjectProperty rdf:ID="isTaughtBy">  
  <owl:domain rdf:resource="#course"/>  
  <owl:range rdf:resource=  
    "#academicStaffMember"/>  
  <rdfs:subPropertyOf  
    rdf:resource="#involves"/>  
</owl:ObjectProperty>
```

Inverse Eigenschaften

```
<owl:ObjectProperty rdf:ID="teaches">  
  <rdfs:range rdf:resource="#course"/>  
  <rdfs:domain rdf:resource=  
    "#academicStaffMember"/>  
  <owl:inverseOf rdf:resource="#isTaughtBy"/>  
</owl:ObjectProperty>
```

Äquivalenz Eigenschaften

owl:equivalentProperty

<owl:ObjectProperty rdf:ID="lecturesIn">

**<owl:equivalentProperty
rdf:resource="#teaches"/>**

</owl:ObjectProperty>

Eigenschaftsrestriktionen

- In OWL ist es möglich zu deklarieren, dass die Klasse C bestimmte Konditionen erfüllen muss
 - Alle Instanzen von C erfüllen die Konditionen
- Die is äquivalent zur Aussage, dass C eine Subklasse von C' ist, wobei C' alle Objekte umfasst die diese Eigenschaften aufweisen
 - C' kann anonym bleiben (anonymous class)

Eigenschaftsrestriktionen (2)

- Eine (Restriktions-) Klasse wird durch das **owl:Restriction** Element erreicht
- Dieses Element enthält das **owl:onProperty** Element und eine oder mehrere **Restriktionsdeklarationen**
- Ein Typ definiert **Kardinalitätsrestriktionen** (z.B. at least one, at most 3,...)

Eigenschaftsrestriktionen (3)

- Die anderen Typen definieren Restriktionen auf der Art der Eigenschaftswerte
 - **owl:allValuesFrom** spezifiziert universelle Quantifizierung
 - **owl:hasValue** spezifiziert einen speziellen Wert
 - **owl:someValuesFrom** spezifiziert existentielle Quantifizierung

owl:allValuesFrom

```
<owl:Class rdf:about="#firstYearCourse">  
  <rdfs:subClassOf>  
    <owl:Restriction>  
      <owl:onProperty rdf:resource="#isTaughtBy"/>  
      <owl:allValuesFrom  
rdf:resource="#Professor"/>  
    </owl:Restriction>  
  </rdfs:subClassOf>  
</owl:Class>
```

owl:hasValue

```
<owl:Class rdf:about="#mathCourse">  
  <rdfs:subClassOf>  
    <owl:Restriction>  
      <owl:onProperty rdf:resource=  
        "#isTaughtBy"/>  
      <owl:hasValue rdf:resource=  
        "#949352"/>  
    </owl:Restriction>  
  </rdfs:subClassOf>  
</owl:Class>
```

owl:someValuesFrom

```
<owl:Class rdf:about="#academicStaffMember">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#teaches"/>
      <owl:someValuesFrom rdf:resource=
        "#undergraduateCourse"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Kardinalitätsrestriktionen

- Minimum und maximum Werte
owl:minCardinality und
owl:maxCardinality
- Es ist möglich eine genaue Zahl anzugeben
- Aus Komfortgründe bietet OWL auch
owl:cardinality

Cardinality Restrictions (2)

```
<owl:Class rdf:about="#course">  
  <rdfs:subClassOf>  
    <owl:Restriction>  
      <owl:onProperty rdf:resource="#isTaughtBy"/>  
      <owl:minCardinality rdf:datatype=  
        "&xsd;nonNegativeInteger">  
        1  
      </owl:minCardinality>  
    </owl:Restriction>  
  </rdfs:subClassOf>  
</owl:Class>
```

Spezielle Eigenschaften

- **owl:TransitiveProperty** (transitive property)
 - z.B. “has better grade than”, “is ancestor of”
- **owl:SymmetricProperty** (symmetry)
 - Z.B. “has same grade as”, “is sibling of”
- **owl:FunctionalProperty** definiert eine Eigenschaft, die höchstens ein Wert für jedes Objekt hat
 - z.B. “age”, “height”, “directSupervisor”
- **owl:InverseFunctionalProperty** definiert eine Eigenschaft für die zwei unterschiedliche Objekte nicht den selben Wert haben können

Spezielle Eigenschaften (2)

```
<owl:ObjectProperty rdf:ID="hasSameGradeAs">  
  <rdf:type  
    rdf:resource="&owl;TransitiveProperty"/>  
  <rdf:type  
    rdf:resource="&owl;SymmetricProperty"/>  
  <rdfs:domain rdf:resource="#student"/>  
  <rdfs:range rdf:resource="#student"/>  
</owl:ObjectProperty>
```

Boolsche Kombinationen

- Können zur Kombination von Klassen genutzt werden (union, intersection, complement)

```
<owl:Class rdf:about="#course">  
  <rdfs:subClassOf>  
    <owl:Restriction>  
      <owl:complementOf rdf:resource=  
        "#staffMember"/>  
    </owl:Restriction>  
  </rdfs:subClassOf>  
</owl:Class>
```

Boolsche Kombinationen(2)

```
<owl:Class rdf:ID="peopleAtUni">  
  <owl:unionOf rdf:parseType="Collection">  
    <owl:Class rdf:about="#staffMember"/>  
    <owl:Class rdf:about="#student"/>  
  </owl:unionOf>  
</owl:Class>
```

- Die neue Klasse ist keine Subklasse der Union, sondern gleich zur Union

Boolsche Kombinationen (3)

```
<owl:Class rdf:ID="facultyInCS">  
  <owl:intersectionOf rdf:parseType="Collection">  
    <owl:Class rdf:about="#faculty"/>  
    <owl:Restriction>  
      <owl:onProperty  
rdf:resource="#belongsTo"/>  
      <owl:hasValue rdf:resource=  
        "#CSDepartment"/>  
    </owl:Restriction>  
  </owl:intersectionOf>  
</owl:Class>
```

Verschachtelung Boolescher Operatoren

```
<owl:Class rdf:ID="adminStaff">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#staffMember"/>
    <owl:complementOf>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#faculty"/>
        <owl:Class rdf:about=
          "#techSupportStaff"/>
      </owl:unionOf>
    </owl:complementOf>
  </owl:intersectionOf>
</owl:Class>
```

Enumerationen mit owl:oneOf

```
<owl:oneOf rdf:parseType="Collection">  
  <owl:Thing rdf:about="#Monday"/>  
  <owl:Thing rdf:about="#Tuesday"/>  
  <owl:Thing rdf:about="#Wednesday"/>  
  <owl:Thing rdf:about="#Thursday"/>  
  <owl:Thing rdf:about="#Friday"/>  
  <owl:Thing rdf:about="#Saturday"/>  
  <owl:Thing rdf:about="#Sunday"/>  
</owl:oneOf>
```

Deklaration von Instanzen

- Instanzen von Klassen werden in RDF deklariert:

```
<rdf:Description rdf:ID="949352">
```

```
  <rdf:type rdf:resource=  
    "#academicStaffMember"/>
```

```
</rdf:Description>
```

```
<academicStaffMember rdf:ID="949352">
```

```
  <uni:age rdf:datatype="&xsd;integer">
```

```
  39<uni:age>
```

```
</academicStaffMember>
```

Getrennte Objekte

- Um sicher zu stellen, dass Individuen auch tatsächlich als solche erkannt werden, wird explizit ihre Ungleichheit definiert:

```
<lecturer rdf:about="949318">
```

```
  <owl:differentFrom rdf:resource="949352"/>
```

```
</lecturer>
```

Unterschiedliche Objekte (2)

- Ankürzungsnotation von paarweise ungleiche Individuen in einer Liste

<owl:allDifferent>

**<owl:distinctMembers
rdf:parseType="Collection">**

<lecturer rdf:about="949318"/>

<lecturer rdf:about="949352"/>

<lecturer rdf:about="949111"/>

</owl:distinctMembers>

</owl:allDifferent>

Versionsinformationen

- **owl:priorVersion** verweist auf frühere Versionen der aktuellen Ontologie
 - Keine formale Bedeutung, kann für das Ontologiemangement benutzt werden
- **owl:versionInfo** String über die aktuelle Version, z.B. Schlüsselwörter

Versionsinformationen(2)

- **owl:backwardCompatibleWith** enthält Referenz zu einer älteren anderen Ontologie
 - Frühere Versionen haben die selbe vorgesehene Interpretation als die neue Version
 - Das Dokument kann sicher geändert auf die neue Version geändert werden
- **owl:incompatibleWith** das die beinhaltende Ontologie eine spätere Version der referenzierten Ontology ist, aber nicht rückwärts kompatibel mit ihr ist

Kombinationen von Features

- In den unterschiedlichen OWL Sprachen gibt es unterschiedliche Menge von Restriktionen auf die anwendbaren Features
- In **OWL Full** können alle Sprachkonstrukte in allen möglichen Kombinationen benutzt werden, solange diese legales RDF sind

Einschränkungen in OWL DL

- **Partitionierung des Vokabulars**
 - Jede Resource kann nur eine Klasse sein, ein Datentype, oder ein Datentypeeigenschaft, eine Objekteigenschaft, ein Individuum, oder Teil eines built-in Vokabulars, aber nicht mehr als eine dieser
- **Explizite Typisierung**
 - Die Partitionierung alle Ressourcen muss explizit angegeben werden (z.B. eine Klasse muss deklariert sein, wenn sie in Konjunktion mit **rdfs:subClassOf** verwendet wird)

Restriktionen in OWL DL (2)

- **Trennung von Eigenschaften**
 - Die Menge an Objekteigenschaften und Datentypeigenschaften ist disjunkt
 - Demzufolge kann folgendes nicht für Datentypeigenschaften spezifiziert werden:
 - owl:inverseOf**
 - owl:FunctionalProperty**
 - owl:InverseFunctionalProperty**
 - owl:SymmetricProperty**

Restriktionen in OWL DL (3)

- **Nicht transitive Kardinalitätsrestriktionen**
 - Auf transitiven Eigenschaften können keine Kardinalitätsrestriktionen definiert werden
- **Eingeschränkte Anonyme Klassen:**
Anonyme Klassen sind nicht erlaubt in:
 - Anwendungsbereich (domain) und Wertebereich (range) von **owl:equivalentClass** oder **owl:disjointWith**
 - Im Wertebereich (aber nicht dem Anwendungsbereich) von **rdfs:subClassOf**

Restriktionen in OWL Lite

- Alle Restriktionen von OWL DL und...
- **owl:oneOf**, **owl:disjointWith**, **owl:unionOf**, **owl:complementOf** und **owl:hasValue** sind nicht erlaubt
- Kardinalitätsaussagen (minimal, maximal, und exact cardinality) können nur auf den Werten 0 oder 1
- **owl:equivalentClass** Aussagen können nicht länger zwischen anonymen Klassen sondern nur noch zwischen Klassen-IDs gemacht werden

OWL Sprachumfang

- OWL Lite RDF Schema Features Synopsis
 - Class
 - rdf:Property
 - rdfs:subClassOf
 - rdfs:subPropertyOf
 - rdfs:domain
 - rdfs:range
 - Individual
- OWL Lite Equality and Inequality Synopsis
 - sameClassAs
 - samePropertyAs
 - sameIndividualAs
 - differentIndividualFrom
- OWL Lite Property Characteristics Synopsis
 - inverseOf
 - TransitiveProperty
 - SymmetricProperty
 - FunctionalProperty (unique)
 - InverseFunctionalProperty (unambiguous)
 - allValuesFrom (universal local range restrictions; previously toClass)
 - someValuesFrom (existential local range restrictions; previously hasClass)

OWL Sprachumfang

- OWL Lite Restricted Cardinality Synopsis
 - minCardinality
 - maxCardinality
 - cardinality
- OWL Class Axioms Synopsis
 - oneOf (enumerated classes)
 - disjointWith
 - sameClassAs applied to class expressions
 - rdfs:subClassOf applied to class expressions
- OWL Boolean Combinations of Class Expressions Synopsis
 - unionOf
 - intersectionOf
 - complementOf
- OWL Arbitrary Cardinality Synopsis
 - minCardinality
 - maxCardinality
 - cardinality
- OWL Filler Information Synopsis
 - hasValue Descriptions can include specific value information

Abschließendes OWL XML/RDF Beispiel

```
<owl:SubClass owl:name="Person">
  <owl:super> <Class "Animal"/></owl:super>

  <owl:restriction>
    <owl:property resource="hasParent">
      <owl:range>
        <owl:Class owl:ID="Person"/>
      </owl:range>
    </owl:property>
  </owl:restriction>

  <owl:restriction>
    <owl:property resource="hasFather" exactly="1"/>
  </owl:restriction>
</owl:SubClass>
```

Unterstützung der Schlussfolgerung (Reasoning) für OWL

- Formale semantik ist eine Vorbedingung für die Schlussfolgerung
- - Abbildung einer Ontologiesprache in einen bekannten logischen Formalismus
 - Benutzung eines automatisierten Reasoners der bereits für diesen Formalismus existiert
- OWL ist (teilweise) in eine Description Logic abgebildet und benutzt Reasoners dafür wie Pellet, FaCT oder RACER
- Description Logics (Beschreibungslogiken) sind ein Subset der Prädikatenlogik. Für dieses Subset ist effizientes Reasoning möglich.

Ontologie-Werkzeuge

- Jena (<http://jena.sourceforge.net/> in Java)
- Pellet (OWL Reasoner in Java)
- KAON (U. Karlsruhe)
- Racer (TUHH)
- FaCT
- TobRaid Composer
- Protégé-2000 (U. of Stanford)
- WebOnto (Open University, GB)
- Coherence
- ...

Hinweis:

Keine “Unique-Names Assumption”

- OWL bietet keine “Unique-Names Assumption” (UNA) wie Datenbanksysteme
 - Wenn zwei Instanzen unterschiedliche Namen oder IDs haben, heißt das nicht, dass sie unterschiedliche Individuen sind
- Annahme: Jeder Kurs wird von einem Universitätsmitarbeiter gelehrt, und ein bestimmter Kurs wird von zwei Mitarbeitern gelehrt.
 - Ein OWL Reasoner gibt hier keine Fehler
 - Stattdessen wird geschlussfolgert, dass die zwei Ressourcen (Mitarbeiter) gleich sind

OWL 1.1 and 2.0

OWL 1.1

- W3C Submission seit 19. Dezember 2006
- Erweitert die W3C OWL Web Ontology Sprache mit nützlichen zusätzlichen Features
 - Extra „syntaktischer Zucker“
 - Addition Eigenschaft und qualifizierte Kardinalitätskonstruktoren
 - Erweiterter Datentypunterstützung
 - Einfaches Metamodelling
 - Erweiterte Annotationen

OWL 2.0

- OWL 2 WG gestartet 6 September 2007; Charter bis 1 Juli 2009
- OWL 2 ist abwärts kompatibel zur Web Ontology Language (OWL 1.x).
- Mehrere neue Konstrukte
 - z.B. qualified cardinality restrictions, role chains, and expressive data predicates.
- [XML Serialization](#)
 - Für XML tools, z.B., XSLT, schema languages, etc.
- Satz and Teil-[Profilen](#) mit unterschiedlichen Anwendungs- und Berechnungseigenschaften.
 - Profile von OWL DL mit besserer Worst-case Performanz für das Reasoning
<http://www.w3.org/TR/owl2-profiles/>

OWL 2.0

- Unterschiedliche Syntax für OWL 2
- Manchester Syntax [[OWL 2 Manchester Syntax](#)]
 - ist eine OWL Syntax welche einfacher für Nicht-Logiker zu lesen ist.
- Functional-Style Syntax [[OWL 2 Specification](#)]
 - Ist einfacher für Spezifikationszwecke und für Schlussfolgerungswerkzeuge.
- OWL XML Syntax ist eine XML Syntax für OWL definiert durch ein XML Schema [[OWL 2 Specification](#)].
- RDF/XML Syntax für OWL ist nur RDF/XML, mit eine speziellen Übersetzung für OWL Konstrukte [[OWL 2 RDF Mapping](#)].
- Es gibt ein Werkzeug, das zwischen den unterschiedlichen Syntaxen für OWL übersetzen kann.

Zusammenfassung

- OWL ist der vorgeschlagene Standard vom W3C für Web Ontologien
- OWL baut auf RDF und RDF Schema auf:
 - (XML-based) RDF Syntax wird benutzt
 - Instanzen werden definiert durch RDF Beschreibungen
 - Fast alle RDFS Modellierungsprimitive werden benutzt

Zusammenfassung (2)

- Formale Semantik und Schlussfolgerung wird durch die Abbildung von OWL in Logik unterstützt
 - Prädikatenlogik and Beschreibungslogiken (Description Logics) werden hierfür verwendet
- Während OWL mächtig genug zur Beschreibung Ontologien ist, werden für Entscheidungslogiken oder Reaktive Logiken Erweiterungen benötigt
 - weitere logische Features, wie z.B. Regeln

Web Ressourcen

- OWL
 - <http://www.w3.org/TR/owl-ref/>
- OWL 2
 - http://www.w3.org/2007/OWL/wiki/OWL_Working_Group
- Buch
 - Grigoris Antoniou, Frank van Harmelen: Semantic Web Primer