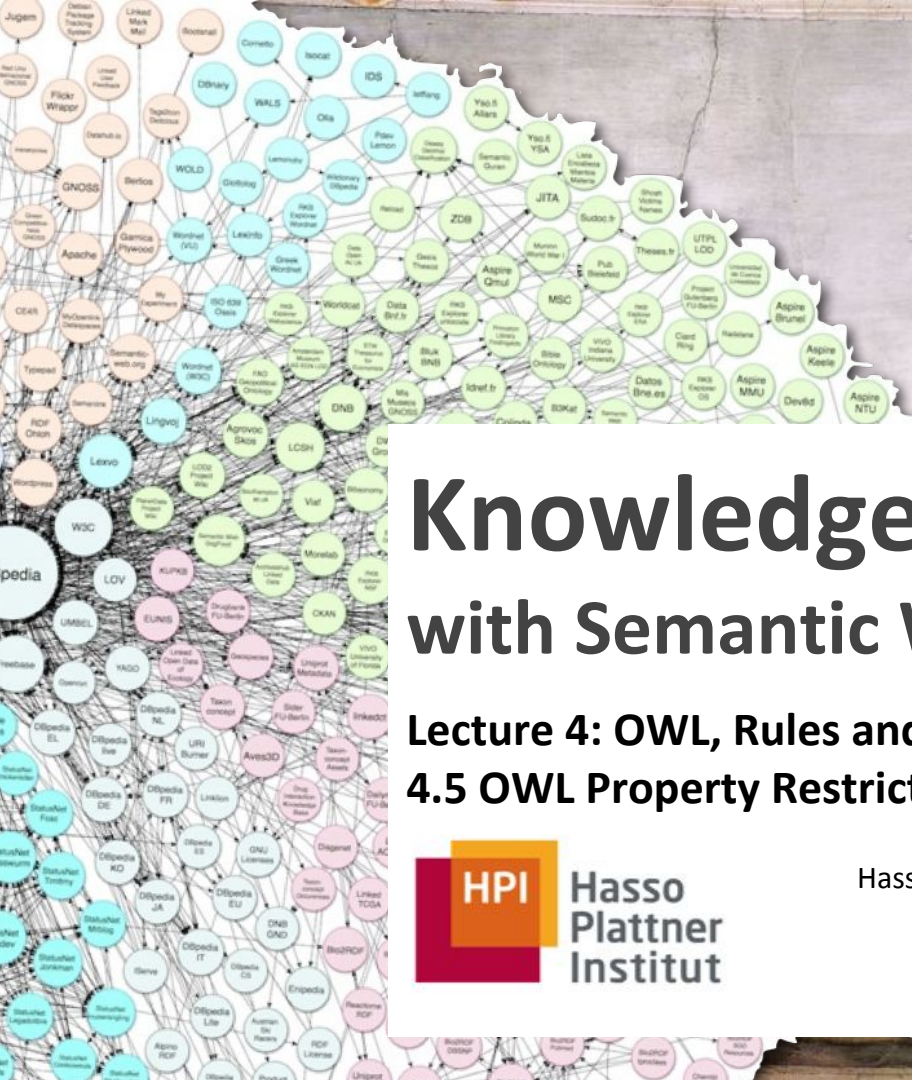


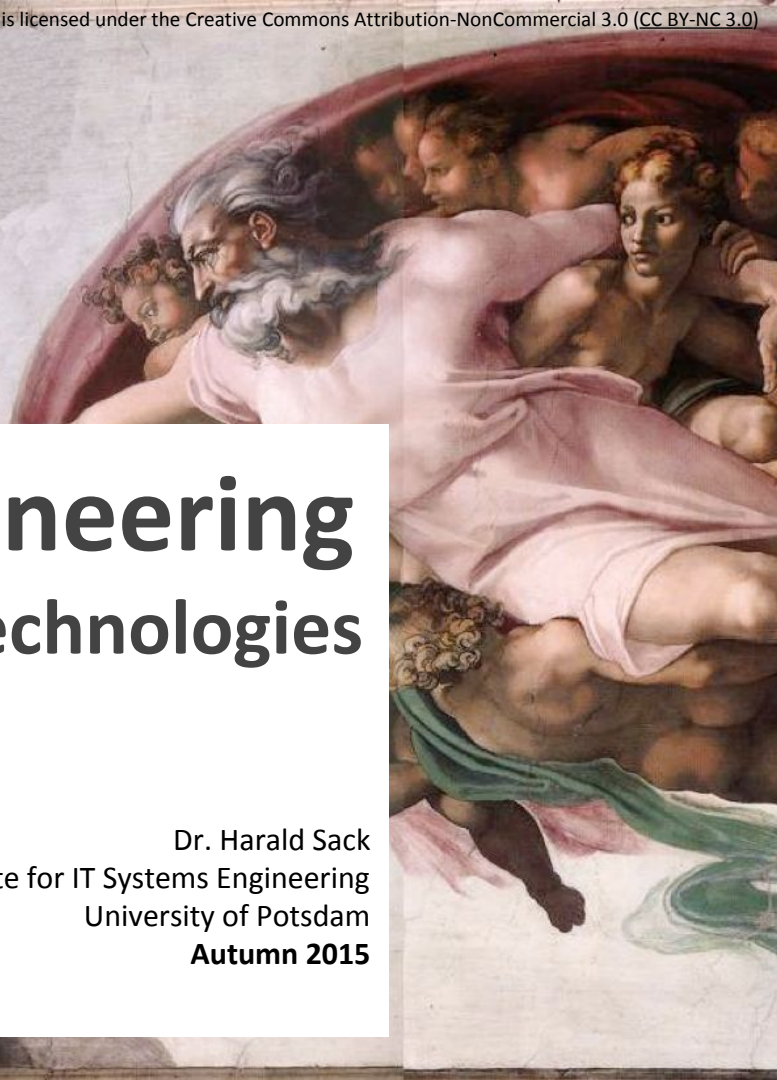


Knowledge Engineering with Semantic Web Technologies

Lecture 4: OWL, Rules and Reasoning 4.5 OWL Property Restrictions



Dr. Harald Sack
Hasso Plattner Institute for IT Systems Engineering
University of Potsdam
Autumn 2015



OWL Property Relationships

- **Property hierarchies** can be created via specializations: `rdfs:subPropertyOf`
- **Inverse properties** are defined via `owl:inverseOf`
- **Identical properties** are defined via `owl:equivalentProperty`

```
:isMadeOf a owl:ObjectProperty ;  
    rdfs:subPropertyOf :consistsOf .  
  
:reads a owl:ObjectProperty ;  
    owl:inverseOf :isReadBy .  
  
:composedOf a owl:ObjectProperty ;  
    owl:equivalentProperty :consistsOf .
```

isMadeOf	⊑	consistsOf
reads-	≡	isReadBy
composedOf	≡	consistsOf

- `owl:TransitiveProperty`

E.g.: if `isPartOf(a,b)` and `isPartOf(b,c)` then it holds that `isPartOf(a,c)`

- `owl:SymmetricProperty`

E.g.: if `isNeighborOf(a,b)`, then it holds that `isNeighborOf(b,a)`

- `owl:FunctionalProperty`

E.g.: if `hasMother(a,b)` and `hasMother(a,c)`, then it holds that `b=c`

- `owl:InverseFunctionalProperty`

E.g.: if `isMotherOf(b,a)` and `isMotherOf(c,a)` then it holds that `b=c`

OWL Transitive Properties

```
:isPublishedBefore a owl:ObjectProperty ;  
    a owl:TransitiveProperty ;  
    rdfs:domain owl:Book ;  
    rdfs:range owl:Book .  
  
:AnimalFarm a owl:Book ;  
    :isPublishedBefore :NineteenEightyfour .  
  
:BraveNewWorld a :Book ;  
    : isPublishedBefore :AnimalFarm .
```

- via inference it can be entailed that „BraveNewWorld“ has also been published before „NineteenEightyfour“ .

OWL More Property Relationships

- **Asymmetric properties** via `owl:AsymmetricProperty`
E.g.: if it holds that `isLeftOf(a,b)`
then it is not possible that also `isLeftOf(b,a)`
- **Reflexive properties** via `owl:ReflexiveProperty`
E.g.: `isRelatedTo(x,x)`
- **Irreflexive properties** via `owl:IrreflexiveProperty`
E.g.: if `isParentOf(x,y)` then $x \neq y$
 $\top \sqsubseteq \neg \exists R. \text{Self}$

OWL Disjunctive Properties

- Two properties R and S are **disjunctive**,
if two individuals x,y are never related via both properties
 - `:hasParent a owl:ObjectProperty ;`
`owl:propertyDisjointWith :hasChild .`
- **Shortcut** for several **disjunctive properties**
 - `[] rdf:type owl:AllDisjointProperties ;`
`owl:members`
`( :hasParent`
`:hasChild`
`:hasGrandchild ) .`

- OWL defines a **universal and empty property** as counterparts for top and bottom classes:
 - `owl:topObjectProperty`
relates each potential pair of individuals, superclass for object properties
 - `owl:bottomObjectProperty`
connects no individuals, subclass of all object properties
 - `owl:topDatatypeProperty`
relates all individuals with all typed literals, superclass for all datatype properties
 - `owl:bottomDatatypeProperty`
connects no individual and no typed literal, subclass of all datatype properties

OWL Qualified Number Restrictions

- Class constructors with **number restrictions on properties connected with a range constraint**

SuccessfulAuthor $\sqsubseteq \geq 1$ notableWork.Bestseller

```
:SuccessfulAuthor a owl:Class;  
  rdfs:subClassOf [  
    a owl:Restriction ;  
    owl:onProperty :notableWork ;  
    owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger;  
    owl:onClass :Bestseller ] .
```

- owl:maxQualifiedCardinality, owl:minQualifiedCardinality,
owl:qualifiedCardinality

OWL Reflexive Property Restriction

- Classes that contain individuals that are **related to themselves** for specific properties

Philosopher $\sqsubseteq$ knows.self

```
:Philosopher a owl:Class ;  
  rdfs:subClassOf [  
    a owl:Restriction ;  
    owl:onProperty :knows ;  
    owl:hasSelf "true"^^xsd:boolean ] .
```

OWL Negated Property Instantiation

- Two individuals can explicitly defined to be **not related with each other** via a given property

`-isBrother(GeorgeOrwell,AldousHuxley)`

```
[ ] rdf:type owl:negativePropertyAssertion ;  
    owl:sourceIndividual :GeorgeOrwell ;  
    owl:assertionProperty :isBrother ;  
    owl:targetIndividual :AldousHuxley .
```



06 OWL Generalized Role Inclusion

OpenHPI - Course Knowledge Engineering with Semantic Web Technologies

Lecture 4: OWL, Rules, and Reasoning