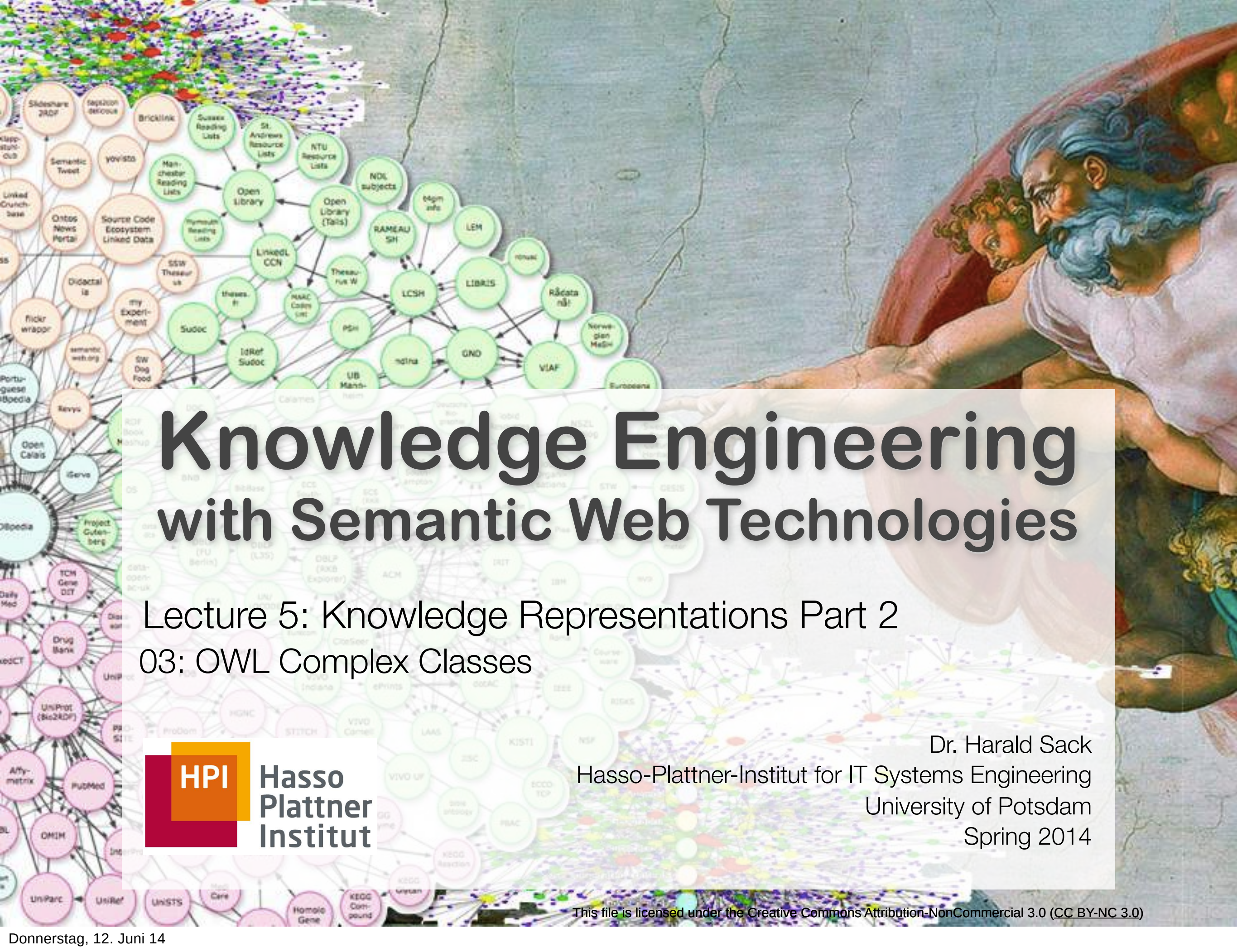




Knowledge Engineering with Semantic Web Technologies

Lecture 5: Knowledge Representations Part 2 03: OWL Complex Classes



Dr. Harald Sack
Hasso-Plattner-Institut for IT Systems Engineering
University of Potsdam
Spring 2014



Lecture 5: Knowledge Representations II

Open HPI Course: Knowledge Engineering with Semantic Web Technologies



03 OWL Complex Classes

Open HPI - Course: Knowledge Engineering with Semantic Web Technologies
Lecture 5: Knowledge Representations II

OWL – Closed Classes (Nominals)

```
:Novel a owl:Class ;  
:AnimalFarm a :Novel .  
:NineteenEightyfour a :Novel .  
:NovelsInStore a owl:Class ;  
    owl:oneOf  
    ( :AnimalFarm  
      :NineteenEightyfour ) .
```

$\text{NovelsInStore} \sqsubseteq \{\text{AnimalFarm}, \text{NineteenEightyfour}\}$

- This says that there are only two novels available in the store.

OWL – Logical Class Constructors

5

- logical AND (conjunction): `owl:intersectionOf` $\sqcap$
- logical OR (disjunction): `owl:unionOf` $\sqcup$
- logical negation: `owl:complementOf` $\neg$
- Logical constructors are applied to create complex class descriptions from atomic classes.

$\text{BooksInStore} \equiv \text{ThingsInStore} \sqcap \text{Books}$

```
:Book a owl:Class .  
:ThingsInStore a owl:Class .  
:BooksInStore a owl:Class ;  
    owl:intersectionOf (:Book :ThingsInStore) .
```

- The class “BooksInStore” results from the intersection of all individuals of the classes “ThingsInStore” and “Book”

Book $\equiv$ Novel $\sqcup$ Poetry $\sqcup$ Nonfiction

```
:Book a owl:Class ;  
    owl:equivalentClass [  
        owl:unionOf ( :Novel  
                        :Poetry  
                        :NonFiction )  
    ] .
```

- Novels, poetry, and non-fiction are also books

Book $\sqsubseteq \neg$ Writer

```
:Book a owl:Class ;  
    rdfs:subClassOf [  
        owl:complementOf :Writer  
    ] .
```

- semantically equivalent assertion:

```
:Book a owl:Class ;  
    owl:disjointWith :Writer .
```

- are used to describe complex classes via properties
- restrictions on values:
 - `owl:hasValue`
 - `owl:allValuesFrom`
 - `owl:someValuesFrom`
- restrictions on cardinality:
 - `owl:cardinality`
 - `owl:minCardinality`
 - `owl:maxCardinality`

OrwellsBooks $\sqsubseteq$ author.(GeorgeOrwell)

```
:OrwellsBooks a owl:Class ;  
               rdfs:subClassOf  
               [  
                 a owl:Restriction ;  
                 owl:onProperty :author ;  
                 owl:hasValue :GeorgeOrwell  
               ] .
```

- Class „OrwellsBooks“ is described via fixed value assignment (=constant) of the individual „GeorgeOrwell“ to the property „author“

```
:Writer a owl:Class .  
:Book a owl:Class .  
:  
:GeorgeOrwell a :Person .  
:author a owl:ObjectProperty ;  
        rdfs:domain :Book ;  
        rdfs:range :Writer .
```

OWL – Property Restrictions

- **owl:allValuesFrom**

fixes all instances of a specific class C as allowed range for a property P $\rightarrow$ (Universal Quantification) $\forall P.C$

Poetry $\sqsubseteq \forall \text{author.Poet}$

```
:Poetry a owl:Class ;
      rdfs:subClassOf
      [ a owl:Restriction ;
        owl:onProperty :author ;
        owl:allValuesFrom :Poet
      ] .
```

```
:Poet a owl:Class .
      rdfs:subClassOf :Writer .
:Poetry a owl:Class
      rdfs:subClassOf :Book .
:author a owl:ObjectProperty ;
      rdfs:domain :Book ;
      rdfs:range :Writer .
```

OWL – Property Restrictions

- **owl:someValuesFrom**

describes that there must exist an individual for property **P** and fixes its range to class **C**

→ (existential quantification) $\exists P.C$

Reader $\sqsubseteq \exists \text{reads}.\text{Book}$

```
:Reader a owl:Class ;  
    rdfs:subClassOf  
    [  
        a owl:Restriction ;  
        owl:onProperty :reads ;  
        owl:someValuesFrom :Book  
    ] .
```

```
:Reader a owl:Class .  
:Book a owl:Class .  
:reads a owl:ObjectProperty ;  
    rdfs:domain :Person ;  
    rdfs:range :ReadingMaterial .
```

OWL – Cardinality Restrictions

13

- `owl:cardinality` restricts to an exact number
- `owl:minCardinality`, `owl:maxCardinality` restricts to upper / lower bounds

`Tetralogy  $\sqsubseteq$  (=4)hasVolumes`

```
:Tetralogy a owl:Class ;  
  rdfs:subClassOf  
  [  
    a owl:Restriction ;  
    owl:onProperty :hasVolumes ;  
    owl:cardinality 4  
  ] .
```

```
:hasVolumes a owl:DatatypeProperty ;  
  rdfs:domain :Thing ;  
  rdfs:range xsd:integer .
```



04 OWL Property Restrictions

Open HPI - Course: Knowledge Engineering with Semantic Web Technologies
Lecture 5: Knowledge Representations II