

Vorlesung Semantic Web



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Vorlesung im Wintersemester 2012/2013

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering

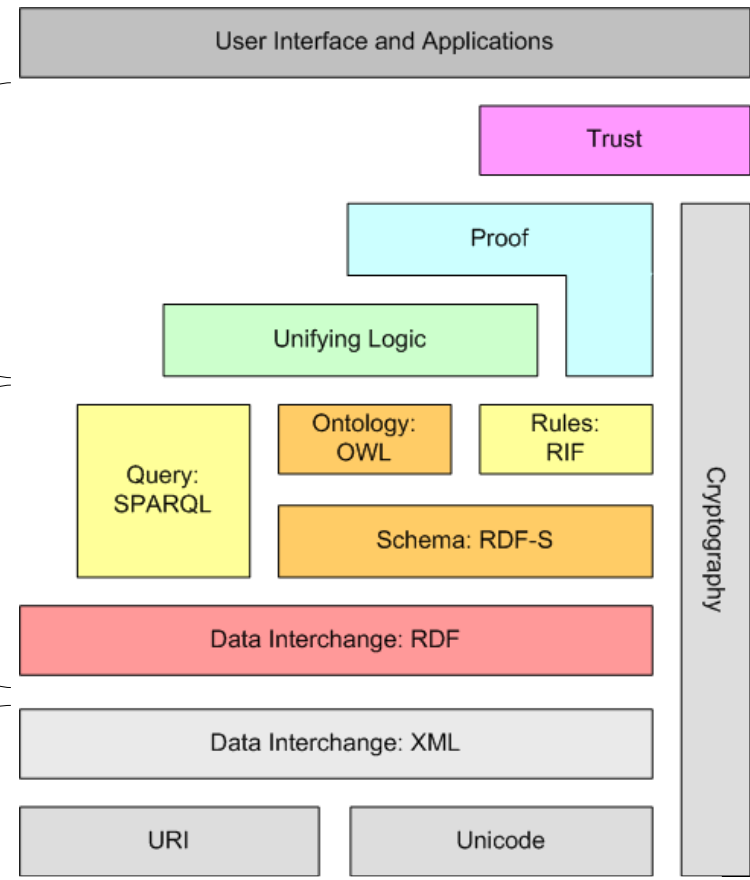
Semantic Web – Aufbau



here be dragons...

Semantic-Web-
Technologie
(Fokus der Vorlesung)

Technische
Grundlagen



Berners-Lee (2009): *Semantic Web and Linked Data*
<http://www.w3.org/2009/Talks/0120-campus-party-tbl/>

Resource Description Framework (RDF)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Geschichte
- Codierungen
- Semantik und Prinzipien
- RDF und HTML

Geschichte: Metadaten im Web

- Ziel: bessere Bewertung von Web-Seiten, z.B. durch Suchmaschinen
- Wer hat die Seite erstellt?
- Wann wurde sie geändert?
- Was ist das Thema der Seite?
- Unter welcher Lizenz stehen die Inhalte?
- Zu welchen anderen Seiten steht sie in Beziehung?

Metadaten im Web: Dublin Core

- Entwickelt 1995 auf einem Workshop in Dublin, Ohio
- 15 vordefinierten Tags
- Standardisiert (ISO 15836:2009)



- kann in HTML eingebettet werden:

```
<html>
  <head profile="http://dublincore.org/documents/2008/08/04/dc-html/">
    <title>Semantic Web</title>
    <link rel="schema.DC"      href="http://purl.org/dc/elements/1.1/" >
    <meta name="DC.publisher"  content="TU Darmstadt" />
    <meta name="DC.subject"   content="Semantic Web" />
    <meta name="DC.creator"   content="Heiko Paulheim" />
    <meta name="DC.relation"  content="http://www.w3.org/2001/sw/" />
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

Metadaten im Web: Dublin Core



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Identifier
- Format
- Type
- Language
- Date
- Title
- Subject
- Coverage
- Description
- Creator
- Publisher
- Contributor
- Rights
- Source
- Relation

Metadaten-Codierung mit RDF/XML



TECHNISCHE
UNIVERSITÄT
DARMSTADT

```
<html>
  <head>
    <link rel="meta" type="application/rdf+xml" title="DC" href="dc.rdf" />
  </head>
  <body>
    ...
  </body>
</html>
```



```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">
    <dc:publisher>TU Darmstadt</dc:publisher>
    <dc:subject>Semantic Web</dc:subject>
    <dc:creator>Heiko Paulheim</dc:creator>
    <dc:relation rdf:resource="http://www.w3.org/2001/sw/" />
  </rdf:Description>
</rdf:RDF>
```

Metadaten im Web: RDF Site Summary (RSS 1.0)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Gibt eine Zusammenfassung der Informationen auf einer Webseite
- Weit verbreitetes Metadatenformat
- Insbesondere bei Blogs verwendet:



```
<channel rdf:about="http://www.xml.com/xml/news.rss">
  <title>XML.com</title>
  <link>http://xml.com/pub</link>
  <description>
    XML.com features a rich mix of information and services for the XML community.
  </description>
  <items>
    <rdf:Seq>
      <rdf:li resource="http://xml.com/pub/2000/08/09/xslt/xslt.html" />
      <rdf:li resource="http://xml.com/pub/2000/08/09/rdfdb/index.html" />
    </rdf:Seq>
  </items>
</channel>
```

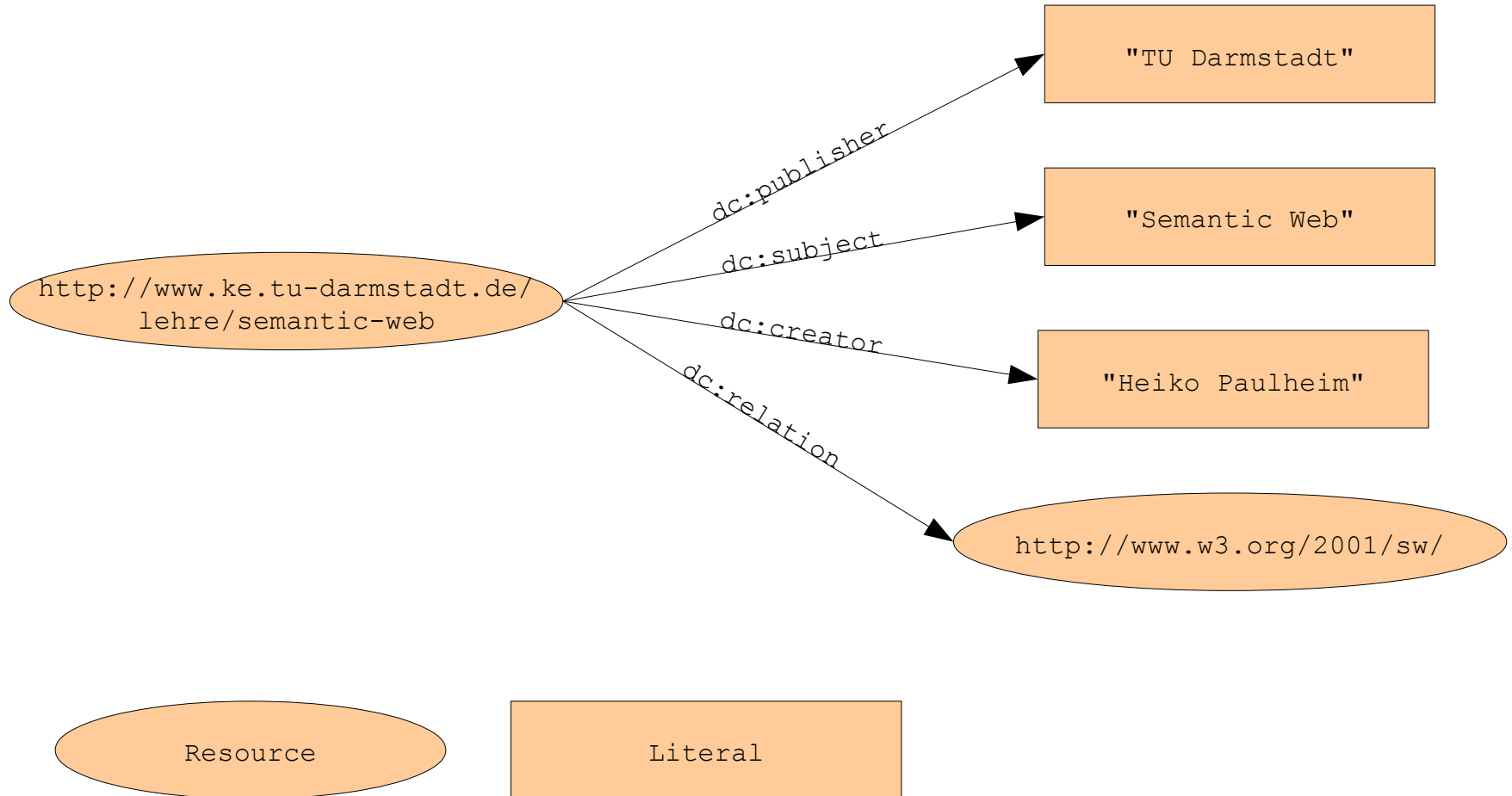
<http://web.resource.org/rss/1.0/spec>

Was ist RDF?

- „Resource Description Framework“
- Standardisiert vom W3C (2004)
- Beschreibung für beliebige Daten
- Interpretation 1: Sätze der Form <Subject, Prädikat, Objekt>
„Heiko arbeitet bei der TU Darmstadt.“
- Interpretation 2: Gerichtete Graphen mit Kantenbeschriftungen



Metadaten in RDF: Graphen

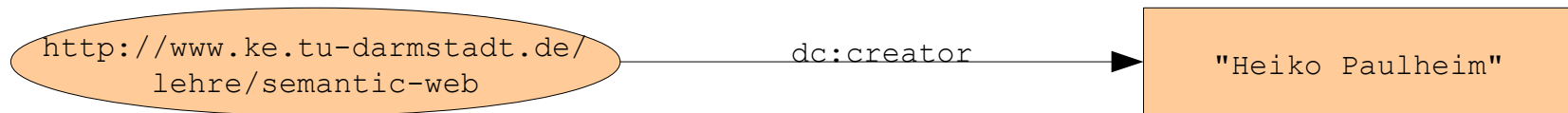


Grundbausteine von RDF

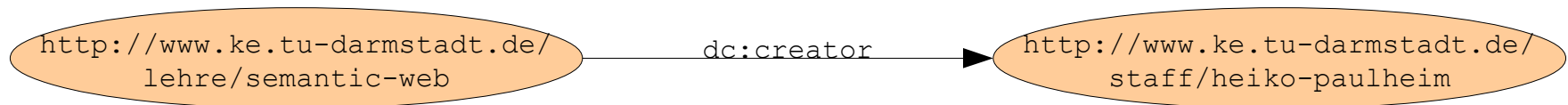
- Ressourcen
 - bezeichnen ganz allgemein Dinge
 - identifiziert durch einen URI
 - können einen oder mehrere Typen haben
- Literale
 - sind Werte, z.B. Strings oder Integer
 - können nur Objekt sein, nicht Subjekt oder Prädikat (d.h., nur eingehende Kanten besitzen)
 - können Datentyp oder Sprachmarkierung haben (aber nicht beides)
 - sonst heißen sie "blanke" Literale
- Properties (Prädikate)
 - Verbinden Ressourcen und Literale

Resource vs. Literal

- Ein Literal ist ein einfacher Wert
 - kann nicht selbst Subjekt sein
 - d.h., an Literalen endet der Graph immer

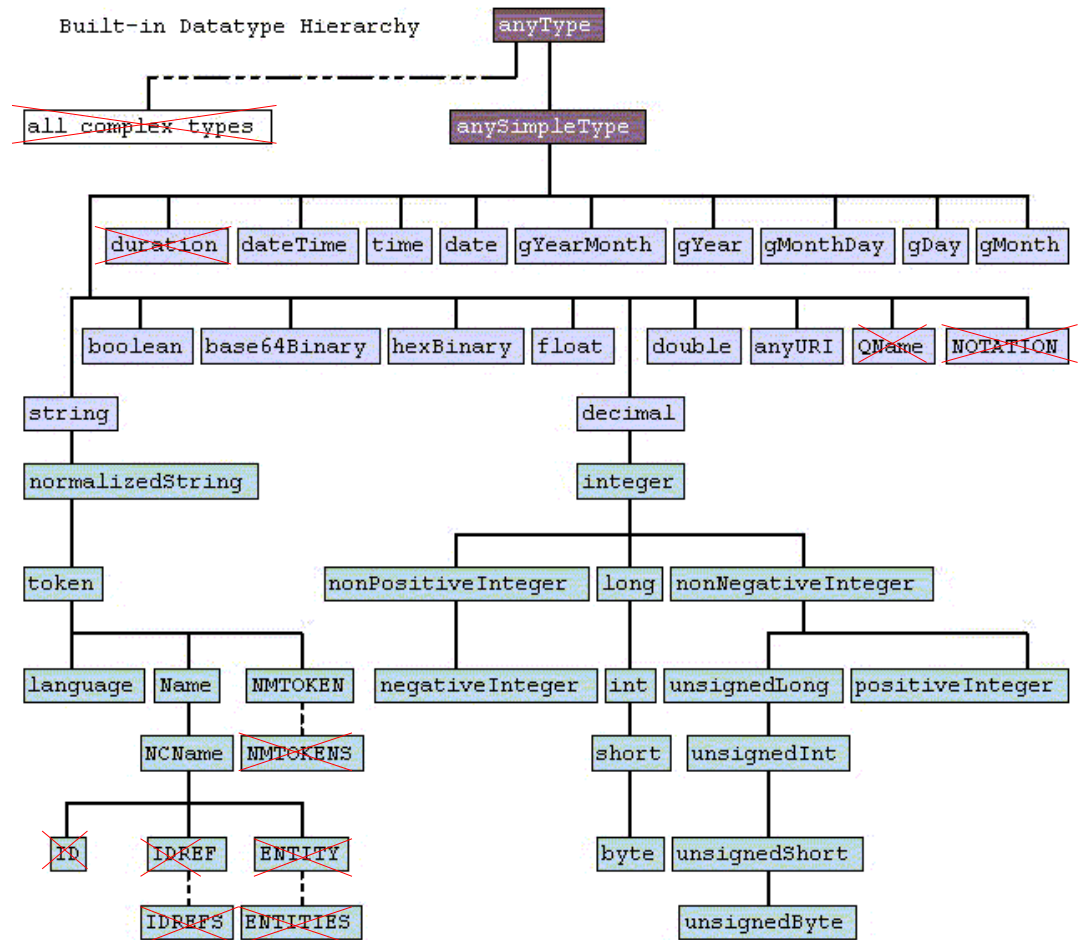


- Eine Resource kann selbst wieder Subjekt sein



Datentypen für Literale

- Es können (fast) alle einfachen Datentypen aus XML Schema verwendet werden
- Ausnahmen:
 - XML-spezifische Typen
 - Der unterspezifizierte Typ "duration"
 - Sequenz-Typen



XML-spezifische Datentypen vs. RDF



▪ XML: Datentypen für ID und IDREF

```
<author id="0815">
  <name>Thomas Glavinic</name>
</author>

<book>
  <title>Die Arbeit der Nacht</title>
  <author idref="0815"/>
</book>
```

▪ RDF: alles hat einen URI

```
<author rdf:about="http://mylibrary.org/author/Thomas_Glavinic">
  <name>Thomas Glavinic</name>
</author>

<book rdf:about "http://mylibrary.org/book/Die_Arbeit_der_Nacht">
  <title>Die Arbeit der Nacht</title>
  <author rdf:resource="http://mylibrary.org/author/Thomas_Glavinic"/>
</book>
```

Sprachmarkierungen für Literale



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- Literale können in mehreren (natürlichen) Sprachen auftreten
 - "München"@de
 - "Munich"@en
- Diese können markiert werden
- Das Semantic Web kann so mehrsprachig implementiert werden!

- Sprachcodes gemäß ISO 963
 - ISO 963-1 (1963): zweistellige Codes, 136 Sprachen
 - ISO 963-2 (1998): dreistellige Codes, 464 Sprachen
 - wenn beides existiert, muss ISO 963-1 verwendet werden

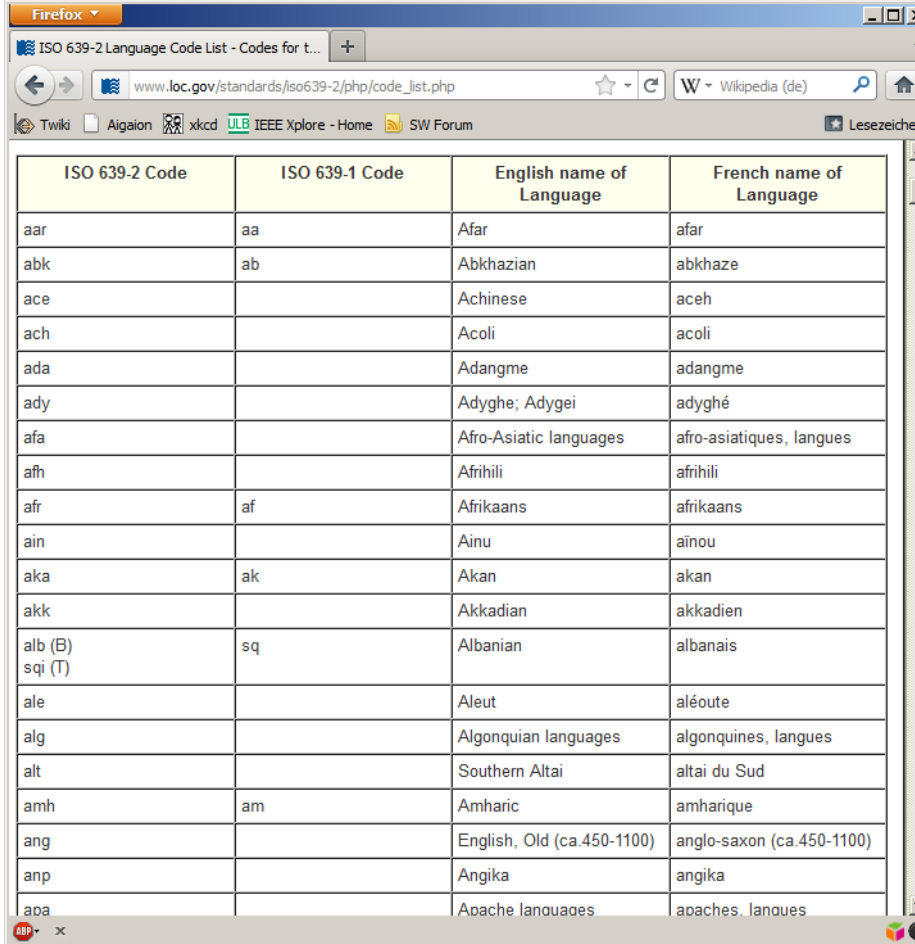


International
Organization for
Standardization

http://www.loc.gov/standards/iso639-2/php/English_list.php



Sprachmarkierungen für Literale



ISO 639-2 Code	ISO 639-1 Code	English name of Language	French name of Language
aar	aa	Afar	afar
abk	ab	Abkhazian	abkhaze
ace		Achinese	aceh
ach		Acoli	acoli
ada		Adangme	adangme
ady		Adyghe; Adygei	adyghé
afa		Afro-Asiatic languages	afro-asiatiques, langues
afh		Afrihili	afrihili
afr	af	Afrikaans	afrikaans
ain		Ainu	ainou
aka	ak	Akan	akan
akk		Akkadian	akkadien
alb (B) sqi (T)	sq	Albanian	albanais
ale		Aleut	aléoute
alg		Algonquian languages	algonquines, langues
alt		Southern Altai	altai du Sud
amh	am	Amharic	amharique
ang		English, Old (ca.450-1100)	anglo-saxon (ca.450-1100)
anp		Angika	angika
apa		Apache languages	apaches, langues

Datentypen in RDF

- Beispiele:
 - :Muenchen :hasName "München"@de .
 - :Muenchen :hasName "Munich"@en .
 - :Muenchen :hasPopulation "1356594"^^xsd:integer .
 - :Muenchen :hasFoundingYear "1158-01-01"^^xsd:date .
- Achtung: es gibt keine default-Datentypen!
- Die folgenden drei sind unterschiedliche Literale:
 - "München"
 - "München"@de
 - "München"^^xsd:string .

Tripel-Notation

- W3C-standardisiert (2004)
- Ein Tripel besteht aus Subjekt, Prädikat, Objekt
- Tripel bilden eine *ungeordnete* Menge



- Einfaches Tripel:

```
<http://www.ke.tu-darmstadt.de/lehre/semantic-web>  
<http://purl.org/dc/elements/1.1/relation>  
<http://www.w3.org/2001/sw/> .
```

- Literal mit Sprachangabe:

```
<http://www.ke.tu-darmstadt.de/lehre/semantic-web>  
<http://purl.org/dc/elements/1.1/subject>  
"Semantisches Web"@de .
```

- Literal mit Typ:

```
<http://www.ke.tu-darmstadt.de/lehre/semantic-web>  
<http://www.informatik.tu-darmstadt.de/mhb/sws>  
"4"^^<http://www.w3.org/2001/XMLSchema#integer> .
```

Vereinfachte Tripel-Notation (N3/Turtle)



- Entwickelt von Tim Berners-Lee 1998
- Kein Standard, aber weit verbreitet
- Vereinfacht die Tripel-Notation

- Namensräume (auch für Default-Präfix) zentral definieren:

```
@prefix dc: <http://purl.org/dc/elements/1.1/>  
@prefix : <http://www.ke.tu-darmstadt.de/lehre/>  
:semantic-web dc:subject "Semantisches Web"@de .
```

- Mehrere Tripel mit gemeinsamem Subjekt / Prädikat:

```
:semantic-web dc:subject "Semantisches Web"@de ,  
                "Semantic Web"@en ;  
                dc:creator "Heiko Paulheim".
```

- Kurznotation für `rdf:type`:

```
:semantic-web a :lecture .
```

Notation RDF/XML

- W3C-standardisiert (2004)
- Codierung von RDF in XML
- Ideal für maschinelle Verarbeitung



- Definition von Ressourcen:

```
<rdf:Description rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <dc:creator>Heiko Paulheim</dc:creator>  
</rdf:Description>
```

- Ressourcen mit Typ:

```
<rdf:Description rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <rdf:type rdf:resource="http://www.informatik.tu-darmstadt.de/mhb/Lecture"/>  
</rdf:Description>
```

- Alternative Darstellung:

```
<mhb:Lecture rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web"  
  xmlns:mhb="http://www.informatik.tu-darmstadt.de/mhb/" />
```

Notation RDF/XML



▪ Beziehungen zwischen Ressourcen durch Schachtelung:

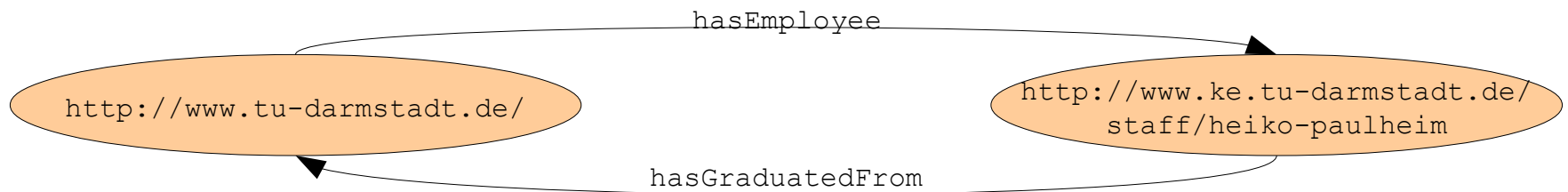
```
<mhb:Lecture rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <mhb:givenBy>  
    <mhb:Lecturer rdf:about="http://www.ke.tu-darmstadt.de/staff/heiko"/>  
  </mhb:givenBy>  
</mhb:Lecture>
```

▪ Beziehungen zwischen Ressourcen durch Verweise:

```
<mhb:Lecturer rdf:about="http://www.ke.tu-darmstadt.de/staff/heiko" />  
  
<mhb:Lecture rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <mhb:givenBy rdf:resource="http://www.ke.tu-darmstadt.de/staff/heiko"/>  
</mhb:Lecture>
```

Notation RDF/XML

- Ein RDF-Graph darf Zyklen enthalten
- Dann kommt man bei RDF/XML nicht ohne Verweise aus:



```
<mhb:University rdf:about="http://www.tu-darmstadt.de">  
  <mhb:hasEmployee>  
    <mhb:UniversityMember rdf:about="http://www.ke.tu-darmstadt.de/staff/heiko-paulheim">  
      <mhb:hasGraduatedFrom rdf:resource="http://www.tu-darmstadt.de"/>  
    </mhb:UniversityMember>  
  </mhb:hasEmployee>  
</mhb:University>
```

Notation RDF/XML

- Literale mit Typ oder Sprachangabe:

```
<mhb:Lecture rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <mhb:title xml:lang="de">Das Semantische Web</mhb:title>  
  <mhb:sws rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2</mhb:sws>  
</mhb:Lecture>
```

- Blanke Literale kann man alternativ als XML-Attribut notieren:

```
<mhb:Lecturer rdf:about="http://www.ke.tu-darmstadt.de/staff/heiko"  
  mhb:name="Heiko Paulheim" />
```

Gleichheit von RDF-Graphen



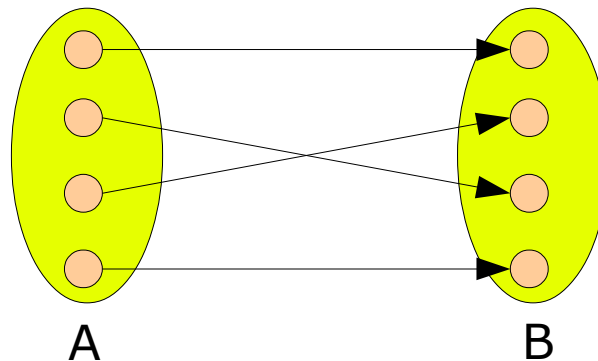
- Wir haben gesehen:
 - es gibt verschiedene Notationen
 - und verschiedene Varianten innerhalb einer Notation
- Wann sind eigentlich zwei RDF-Graphen gleich?
 - d.h.: sie codieren dieselbe Information

Gleichheit von RDF-Graphen

- Zwei RDF-Graphen G und G' sind gleich, wenn es eine bijektive Abbildung M gibt, so dass
 - $M(L) = L$ für alle Literale $L \in G$
(Gleichheit inkl. Sprachmarkierung und Datentyp!)
 - $M(U) = U$ für alle URIs $U \in G$
 - $\langle s, p, o \rangle \in G \Leftrightarrow \langle M(s), p, M(o) \rangle \in G'$
- Vereinfacht gesprochen:
alle Tripel aus G sind in G' enthalten und umgekehrt
- Ein RDF-Graph kann verschiedene Serialisierungen haben
⇒ Verschiedene Serialisierungen sind nicht zwingend verschiedene Graphen!

Exkurs: bijektive Abbildung

- Eine bijektive Abbildung $M: A \rightarrow B$ ist gleichzeitig injektiv und surjektiv
- Injektiv:
 - Jeder Wert in B wird höchstens einmal angenommen
- Surjektiv:
 - Jeder Wert in B wird mindestens einmal angenommen



Beispiel

▪ Graph A:

```
@prefix ke: <http://www.ke.tu-darmstadt.de/>  
ke:staff/heiko-paulheim ke:teaches ke:lehre/semantic-web
```

▪ Graph B:

```
<rdf:Description  
  rdf:about="http://www.ke.tu-darmstadt.de/staff/heiko-paulheim"  
  xmlns:ke="http://www.ke.tu-darmstadt.de/">  
  <ke:teaches>  
    <rdf:Description  
      rdf:about="http://ke.tu-darmstadt.de/lehre/semantic-web"/>  
  </rdf:Description>
```

Beispiel

- Mapping-Funktion M

$M(\text{http://www.ke.tu-darmstadt.de/staff/heiko-paulheim})$

$= \text{http://www.ke.tu-darmstadt.de/staff/heiko-paulheim}$

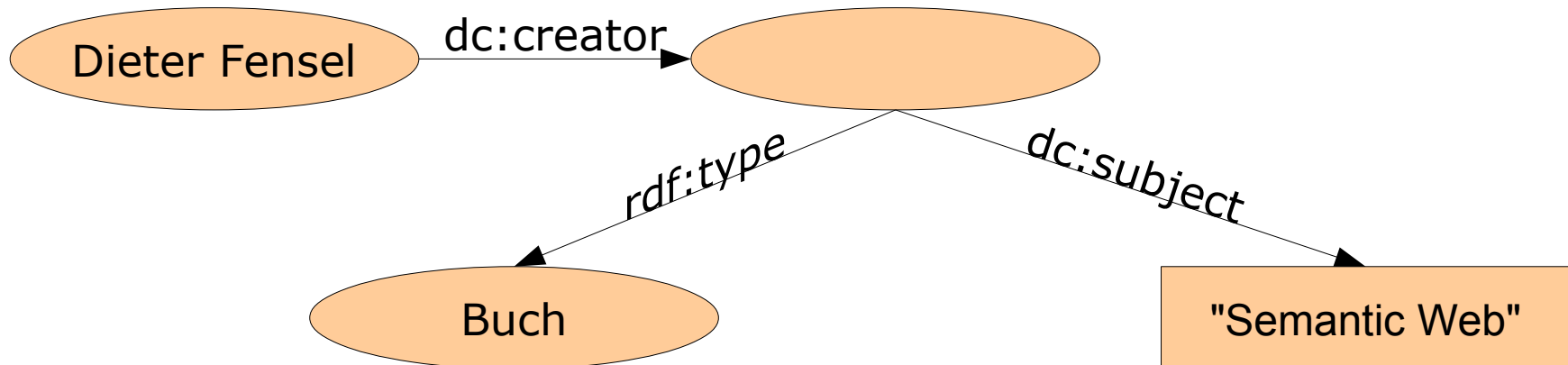
$M(\text{http://www.ke.tu-darmstadt.de/lehre/semantic-web})$

$= \text{http://www.ke.tu-darmstadt.de/lehre/semantic-web}$

- alle URIs und Literale in beiden Graphen abgedeckt
- alle Mappings auf identische URIs
- Relation $A \text{ ke:teaches } B \iff M(A) \text{ ke:teaches } M(B)$.

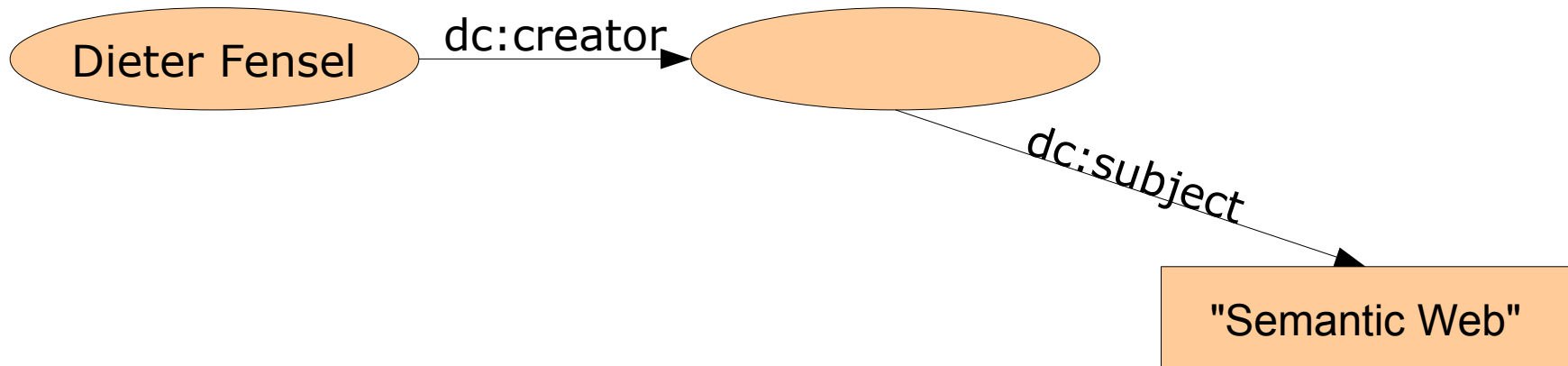
Blank Nodes (Leere Knoten)

- Dinge, die man nicht genau spezifizieren will oder kann:
 - "Dieter Fensel hat ein Buch über das Semantic Web geschrieben."



Blank Nodes (Leere Knoten)

- Dinge, die man nicht genau spezifizieren will oder kann:
 - "Dieter Fensel hat *etwas* über das Semantic Web geschrieben."



Blank Nodes in N3



- Variante 1: explizit benannt mit Underscore

```
:Dieter_Fensel dc:creator _:x .  
_:x a :Book ;  
    dc:subject "Semantic Web" .
```

- Variante 2: unbenannt mit eckigen Klammern

```
:Dieter_Fensel dc:creator  
[ a :Book;  
  dc:subject "Semantic Web" ].
```

Blank Nodes in RDF/XML

▪ Variante 1: durch Weglassen von rdf:about

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator>  
    <lib:Book>  
      <dc:subject>Semantic Web</dc:subject>  
    </lib:Book>  
  </dc:creator>  
</rdf:Description>
```

mit Typ

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator>  
    <rdf:Description>  
      <dc:subject>Semantic Web</dc:subject>  
    </rdf:Description>  
  </dc:creator>  
</rdf:Description>
```

ohne Typ
(rdf:about fehlt)

Blank Nodes in RDF/XML



▪ Variante 2: mit `rdf:parseType="Resource"` (ohne Typ)

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator rdf:parseType="Resource">  
    <dc:subject>Semantic Web</dc:subject>  
  </dc:creator>  
</rdf:Description>
```

▪ Variante 3: mit expliziten Bezeichnern

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator rdf:nodeID="x" />  
</rdf:Description>  
<rdf:Description rdf:nodeID="x">  
  <dc:subject>Semantic Web</dc:subject>  
</rdf:Description>
```

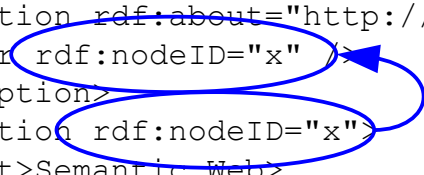
Blank Nodes in RDF/XML

- Variante 2: mit `rdf:parseType="Resource"` (ohne Typ)

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator rdf:parseType="Resource">  
    <dc:subject>Semantic Web</dc:subject>  
  </dc:creator>  
</rdf:Description>
```

- Variante 3: mit expliziten Bezeichnern

```
<rdf:Description rdf:about="http://www.mylibrary.org/Dieter_Fensel">  
  <dc:creator rdf:nodeID="x" />  
</rdf:Description>  
<rdf:Description rdf:nodeID="x">  
  <dc:subject>Semantic Web</dc:subject>  
</rdf:Description>
```



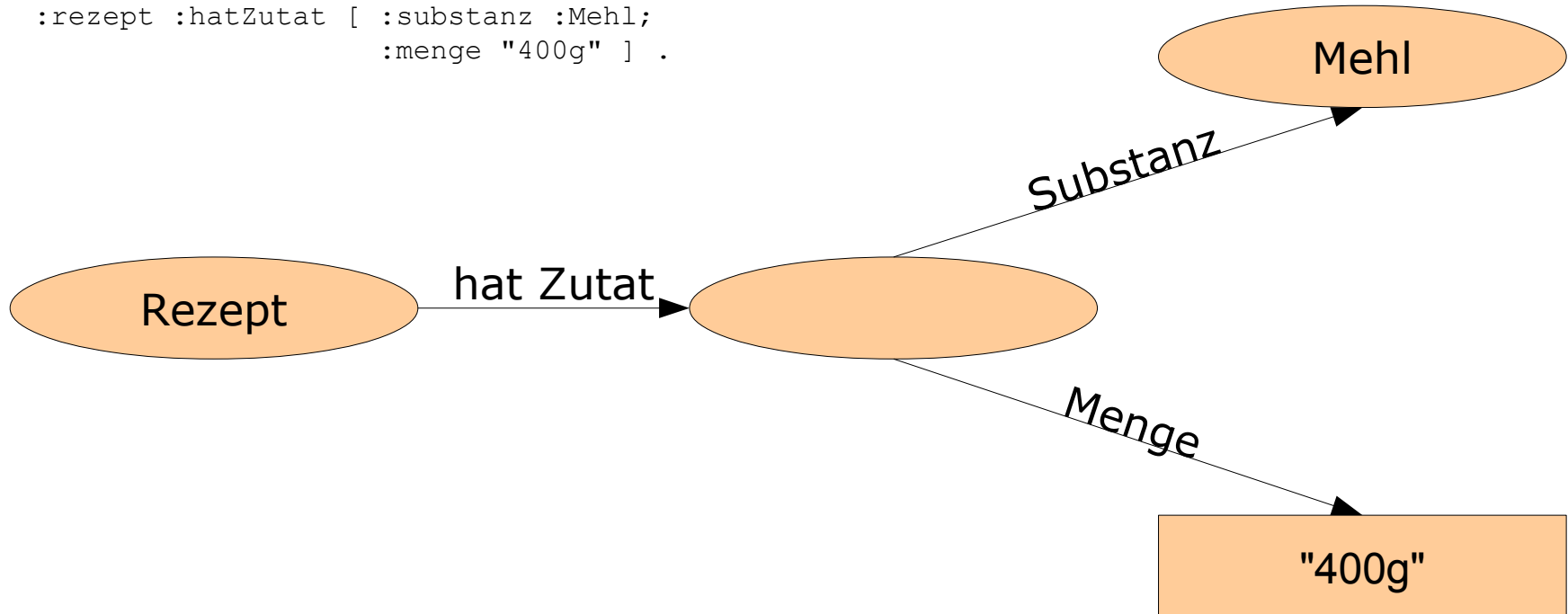
Anwendung für Blank Nodes: mehrwertige Prädikate

- RDF-Prädikate verbinden immer Subjekt und Objekt
 - sind daher *zweistellig* im Sinne der Prädikatenlogik:
 - `:Heiko :arbeitet_bei :TUD .`
 - ⇔ `arbeitet_bei(Heiko, TUD) .`
- Manchmal braucht man aber auch mehrstellige Prädikate
 - `hat_Zutat(Rezept, Mehl, 400g)`

Anwendung für Blank Nodes: mehrwertige Prädikate

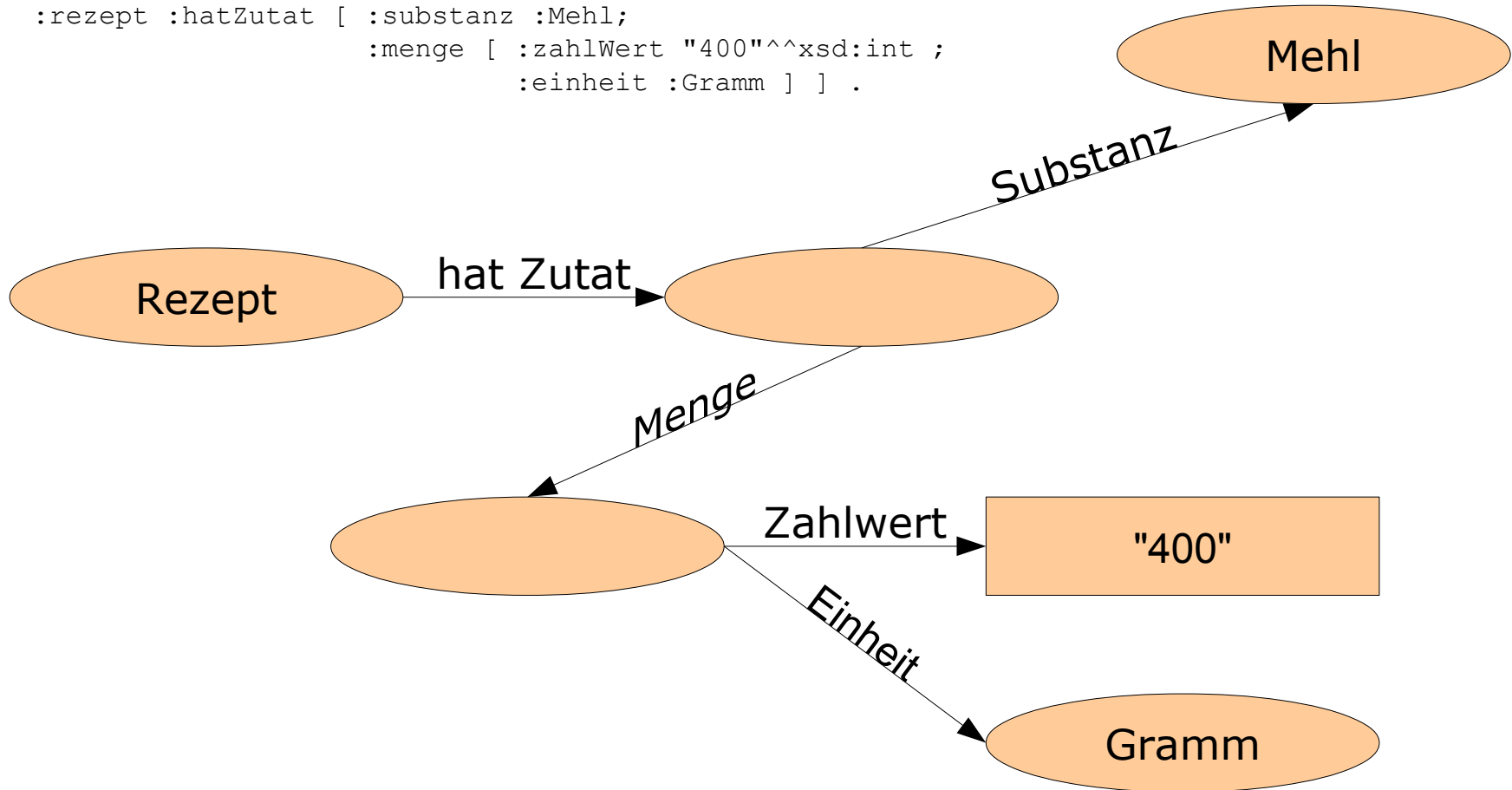


```
:rezept :hatZutat [ :substanz :Mehl;  
                  :menge "400g" ] .
```



Anwendung für Blank Nodes: mehrwertige Prädikate

```
:rezept :hatZutat [ :substanz :Mehl;  
                  :menge [ :zahlWert "400"^^xsd:int ;  
                          :einheit :Gramm ] ] .
```



Blank Nodes und Gleichheit von Graphen

- Recap: Zwei RDF-Graphen G und G' sind gleich, wenn es eine bijektive Abbildung M gibt, so dass
 - $M(L) = L$ für alle Literale $L \in G$
(Gleichheit inkl. Sprachmarkierung und Datentyp!)
 - $M(U) = U$ für alle URIs $U \in G$
 - $\langle s, p, o \rangle \in G \Leftrightarrow \langle M(s), p, M(o) \rangle \in G'$
 - $M(b) = b', M(b_1) \neq M(b_2)$ für alle Blank Nodes $b, b_1 \neq b_2 \in G, b' \in G'$
- Das heißt im Klartext: Blank Nodes können umbenannt werden!

Blank Nodes und Gleichheit von Graphen

- Variante 1a: explizit benannt mit Underscore

```
:Dieter_Fensel dc:creator _:x .  
_:x a :Book ;  
    dc:subject "Semantic Web" .
```

- Variante 1b: explizit benannt mit Underscore

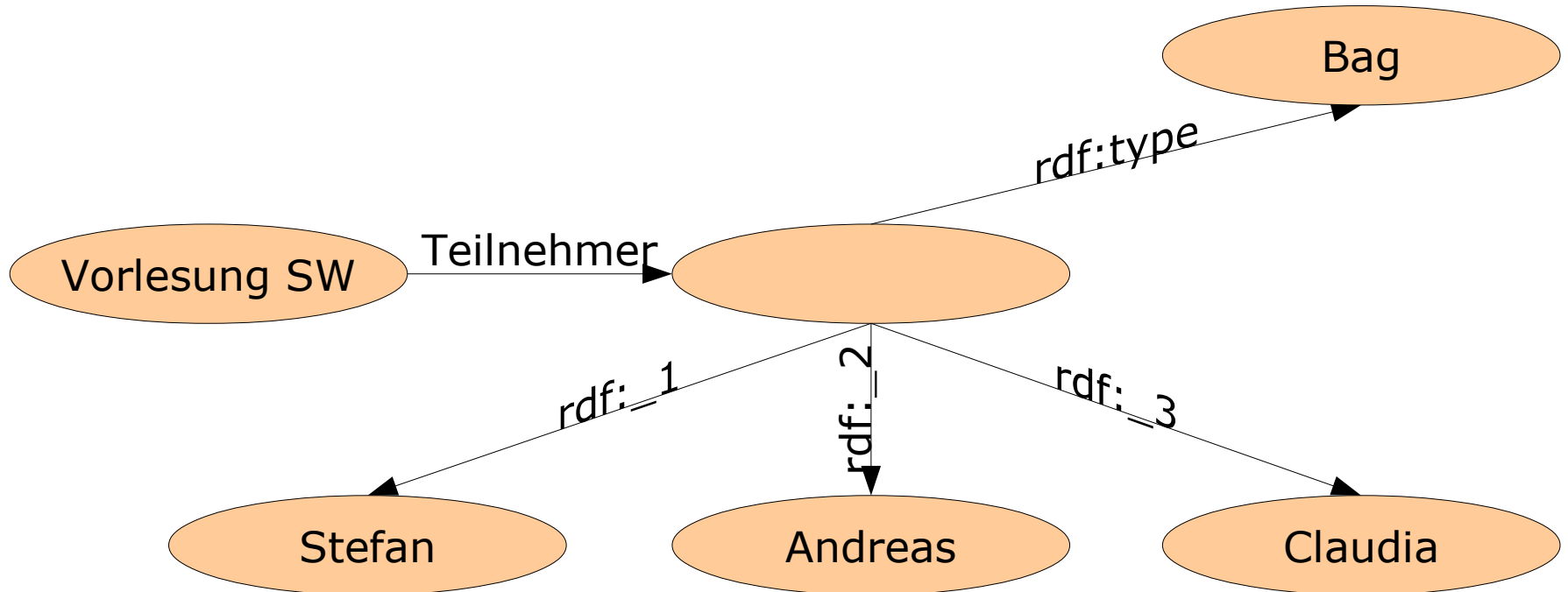
```
:Dieter_Fensel dc:creator _:genid01 .  
_:genid01 a :Book ;  
    dc:subject "Semantic Web" .
```

- Variante 2: unbenannt mit eckigen Klammern

```
:Dieter_Fensel dc:creator  
[ a :Book;  
  dc:subject "Semantic Web" ].
```

- ...alle diese Varianten codieren denselben Graphen!

Ungeordnete Mengen: Bags



Codierung von Bags

▪ In N3:

```
:Vorlesung_SW :hatTeilnehmer [ a rdf:Bag ;  
                                rdf:_1 :Stefan;  
                                rdf:_2 :Andreas;  
                                rdf:_3 :Claudia ] .
```

▪ In RDF/XML:

```
<rdf:Description rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <hatTeilnehmer>  
    <rdf:Bag>  
      <rdf:_1 rdf:resource="http://www.tu-darmstadt.de/students/Stefan">  
      <rdf:_2 rdf:resource="http://www.tu-darmstadt.de/students/Andreas">  
      <rdf:_3 rdf:resource="http://www.tu-darmstadt.de/students/Claudia">  
    </rdf:Bag>  
  </hatTeilnehmer>  
</rdf:Description>
```

- Der Bag selbst ist (in diesem Beispiel) ein Blank Node
- Sämtliche Varianten für Blank Nodes sind natürlich möglich

Kurznotation für Bags

- Statt `rdf:_1`, `rdf:_2`, `rdf:_3`,... kann man in RDF/XML auch `rdf:li` verwenden
- Das ermöglicht auch folgende Notation:

```
<rdf:Description rdf:about="http://www.ke.tu-darmstadt.de/lehre/semantic-web">  
  <hatTeilnehmer>  
    <rdf:Bag>  
      <rdf:li rdf:resource="http://www.tu-darmstadt.de/students/Stefan">  
      <rdf:li rdf:resource="http://www.tu-darmstadt.de/students/Andreas">  
      <rdf:li rdf:resource="http://www.tu-darmstadt.de/students/Claudia">  
    </rdf:Bag>  
  </hatTeilnehmer>  
</rdf:Description>
```

- An dieser Stelle (und nur an dieser!) ist die Reihenfolge der Aussagen im RDF/XML-Dokument relevant!

Semantic von Bags



- Vorsicht, Falle!
- Bags sind zwar ungeordnet
- Die folgenden Graphen sind dennoch nicht äquivalent:

```
:Vorlesung_SW :hatTeilnehmer  
  [ a rdf:Bag ;  
    rdf:_1 :Stefan;  
    rdf:_2 :Andreas;  
    rdf:_3 :Claudia ] .
```

```
:Vorlesung_SW :hatTeilnehmer  
  [ a rdf:Bag ;  
    rdf:_1 :Andreas;  
    rdf:_2 :Claudia;  
    rdf:_3 :Stefan ] .
```

<http://www.nichtlustig.de/toondb/110915.html>



Semantik von Bags



<http://www.cartoonstock.com/newscartoons/cartoonists/ama/lowres/aman212l.jpg>

Semantik von Bags

- Bags sind eigentlich ungeordnet
- Dennoch beschreiben die folgenden Definitionen verschiedene Bags:

```
:Vorlesung_SW :hatTeilnehmer
[ a rdf:Bag ;
  rdf:_1 :Stefan;
  rdf:_2 :Andreas;
  rdf:_3 :Claudia ] .
```

```
:Vorlesung_SW :hatTeilnehmer
[ a rdf:Bag ;
  rdf:_1 :Andreas;
  rdf:_2 :Claudia;
  rdf:_3 :Stefan ] .
```

- Warum? Und warum nicht immer `rdf:li` verwenden?

Semantik von Bags

- rdf:li existiert nur als "Abkürzung" in der RDF/XML-Serialisierung
- In RDF selbst gibt es nur rdf:_1, rdf:_2, ...
- rdf:li wird beim Parsen von XML aufgelöst
 - daraus folgt: im XML-Dokument kommt es hier auf die Reihenfolge an
 - (ansonsten ist die Reihenfolge in RDF/XML fast immer egal)

Semantik von Bags

- rdf:li existiert nur als "Abkürzung" in der RDF/XML-Serialisierung
- In RDF selbst gibt es nur rdf:_1, rdf:_2, ...
- rdf:li wird beim Parsen von XML aufgelöst
 - daraus folgt: im XML-Dokument kommt es hier auf die Reihenfolge an
 - (das ist übrigens die einzige Stelle!)

Semantik von Bags

- Stimmen von der W3C-Semantic-Web-Mailingliste...
- Toby:
 - *rdf:Bag is ordered, but the order is not considered significant.*
 - *Modern RDF never uses rdf:Bag [...] anyway.*
- Pat:
 - *No, [they] are different graphs. But to emphasise again, this does not mean that they are describing different bags. The graph is only a **description**. The same thing - in this case, a small bag - can often be described in a number of different ways.*
 - *The numbers in the property names `_1`, `_2`, etc. should not be interpreted as imposing an order on the elements.*

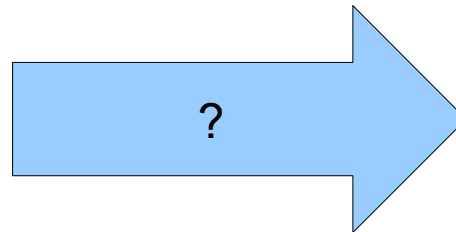
Semantik von Bags

- Stimmen von der Semantic-Web-Mailingliste
- Antoine:
 - *[The graphs] are not equivalent either in terms of formal semantics, but at least they have a certain level of "informal equivalence" in the sense that they "intend" to describe the same set.*
- Dieter:
 - *Is it then fair to say that the formal RDF semantics is "broken" not reflecting the intuitive semantics of RDF?*

Semantik von Bags

- Michael:
 - *In addition, [...] trying to formalize "intuitive semantics" for Bags can easily lead to very unintuitive results.*

```
:Vorlesung_SW :hatTeilnehmer  
[ a rdf:Bag ;  
  rdf:_1 :Stefan;  
  rdf:_2 :Andreas;  
  rdf:_3 :Claudia ] .
```



```
:Vorlesung_SW :hatTeilnehmer  
[ a rdf:Bag ;  
  rdf:_1 :Andreas;  
  rdf:_2 :Claudia;  
  rdf:_3 :Stefan ] .
```

:Stefan == :Andreas

Bags vs. wiederholte Properties



- Variante 1 mit Bags:

```
:Vorlesung_SW :hatTeilnehmer [ a rdf:Bag ;  
                                rdf:_1 :Stefan;  
                                rdf:_2 :Andreas;  
                                rdf:_3 :Claudia ] .
```

- Variante 2 mit wiederholten Properties:

```
:Vorlesung_SW :hatTeilnehmer :Stefan.  
:Vorlesung_SW :hatTeilnehmer :Andreas.  
:Vorlesung_SW :hatTeilnehmer :Claudia.
```

- Diese beiden Varianten codieren unterschiedliche Graphen!
- Semantischer Unterschied (in diesem Fall etwas fuzzy):
 - Bei Variante 1 geht es um eine Menge von Teilnehmern, die als Menge gemeinsam teilnimmt
 - Bei Variante 2 geht es um Teilnehmer, die jeder als Individuum teilnehmen

Bags vs. wiederholte Properties

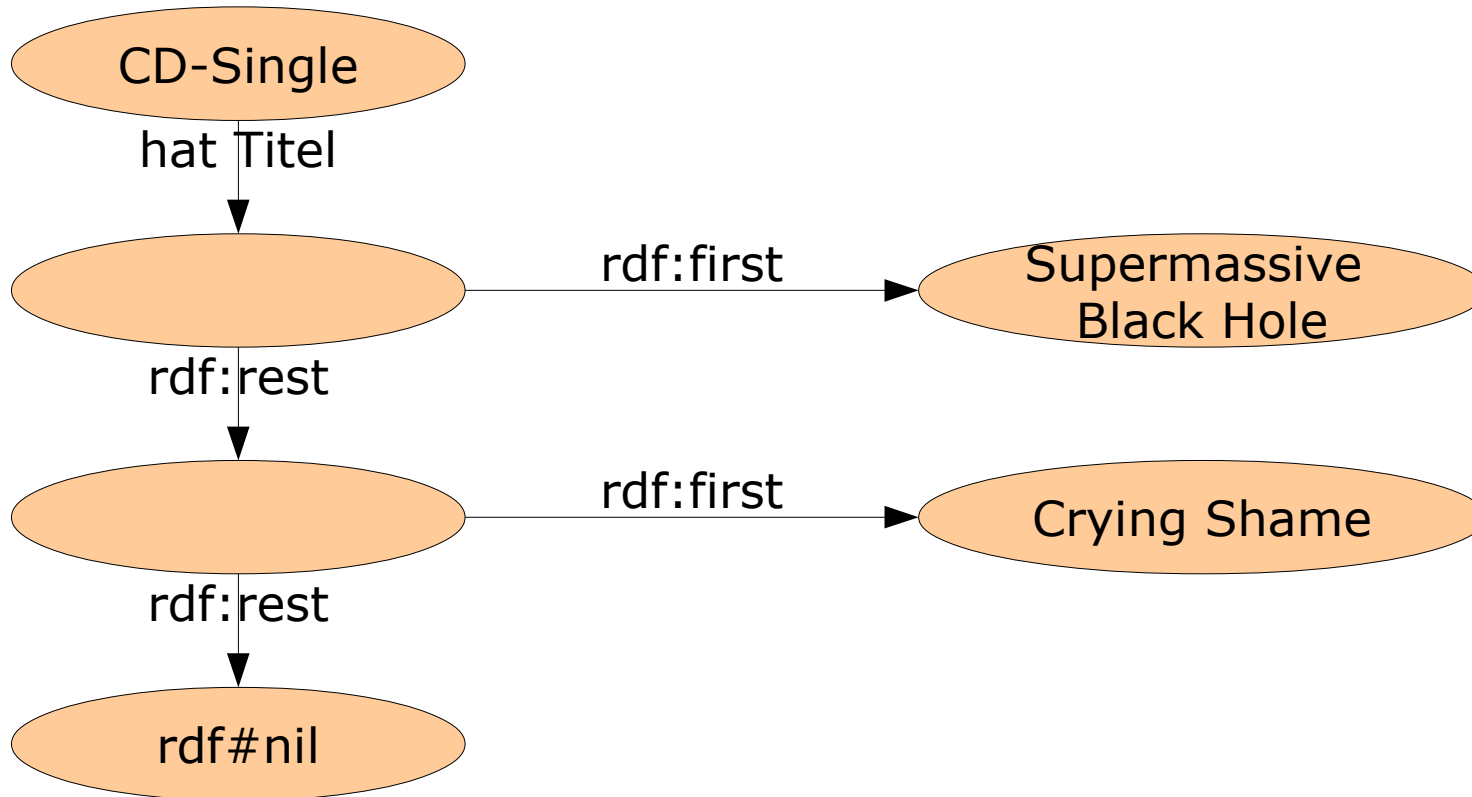
- Es gibt aber auch eindeutigere Fälle...
- Bags:
 - Ein Gremium von Menschen trifft eine Entscheidung (und nicht einer allein)
 - Zwei Autos sind in einen Zusammenstoß verwickelt (und nicht jedes für sich)
- Wiederholte Properties:
 - Jemand hat eine Menge von Büchern aus der Bibliothek ausgeliehen (können auch einzeln entliehen und zurückgegeben werden)
 - Meine Freunde (habe ich in der Regel unabhängig voneinander kennen gelernt und habe zu jedem eine individuelle Beziehung)

Bags vs. wiederholte Properties



- Es gibt aber auch eindeutigere Fälle...
- Bags:
 - Ein Gremium von Menschen trifft eine Entscheidung (und nicht einer allein)
 - Zwei Autos sind in einen Zusammenstoß verwickelt (und nicht jedes für sich)
- Wiederholte Properties:
 - Jemand hat eine Menge von Büchern aus der Bibliothek ausgeliehen (können auch einzeln entliehen und zurückgegeben werden)
 - Meine Freunde (habe ich in der Regel unabhängig voneinander kennen gelernt und habe zu jedem eine individuelle Beziehung)

Geordnete Mengen: Listen



Codierung von Listen

- In N3 ganz einfach:

```
:Single_SMBH :hatPlaylist ( :Supermassive_Black_Hole :Crying_Shame ) .
```

- In RDF/XML (umständliche Variante):

```
<rdf:Description rdf:about="http://www.mymusic.org/Muse/Single_SMBH">
  <hatPlaylist>
    <rdf:Description>
      <rdf:first rdf:resource="http://www.mymusic.org/Muse/Supermassive_Black_Hole" />
      <rdf:rest>
        <rdf:Description>
          <rdf:first rdf:resource="http://www.mymusic.org/Muse/Crying_Shame" />
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:Description>
      </rdf:rest>
    </rdf:Description>
  </hatPlaylist>
</rdf:Description>
```

- Die Liste selbst ist ein Blank Node

- Sämtliche Varianten für Blank Nodes sind natürlich möglich

Codierung von Listen

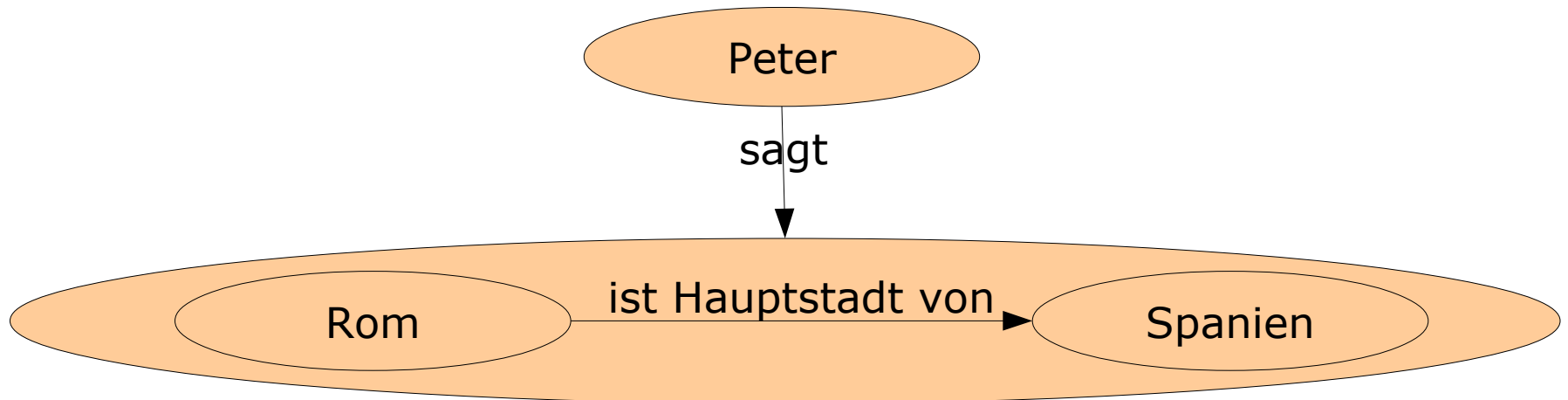
- In RDF/XML (vereinfachte Variante):

```
<rdf:Description rdf:about="http://www.mymusic.org/Muse/Single_SMH">  
  <hatPlaylist rdf:parseType="Collection">  
    <rdf:Description rdf:resource="http://www.mymusic.org/Muse/Supermassive_Black_Hole" />  
    <rdf:Description rdf:resource="http://www.mymusic.org/Muse/Crying_Shame" />  
  </hatPlaylist>  
</rdf:Description>
```

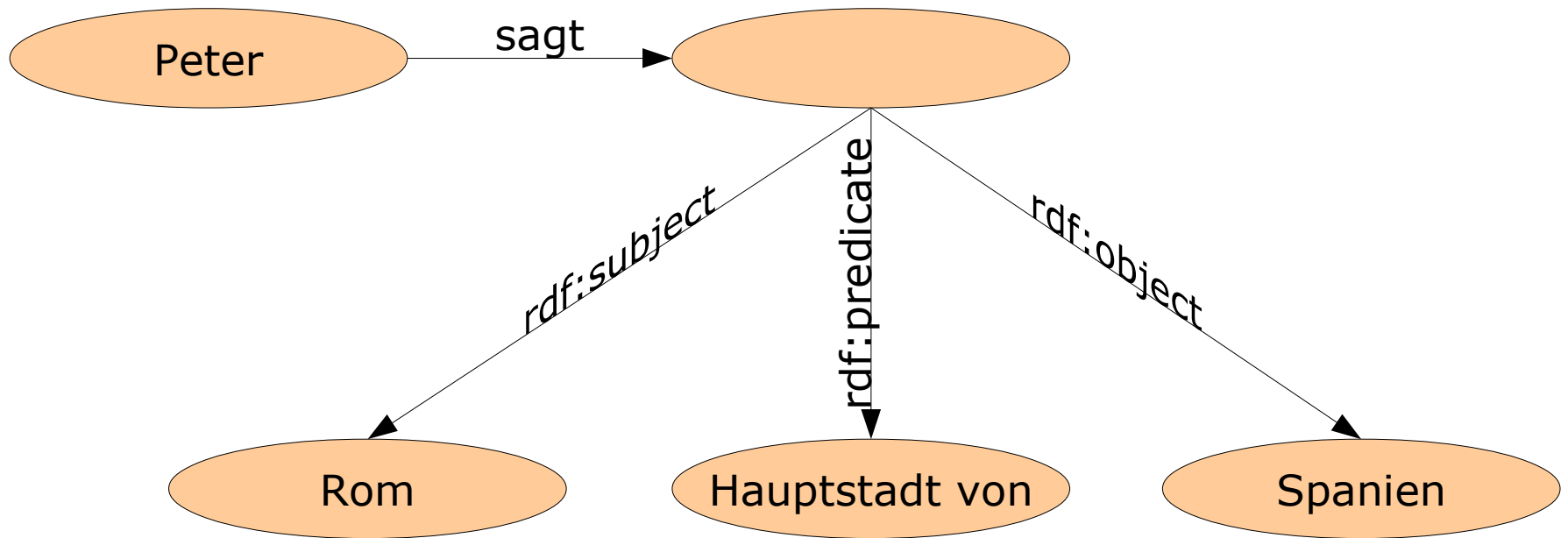
- Listen sind stets geordnet
- definieren also implizit noch eine Reihenfolge
- Achtung: auch hier ist die Reihenfolge in RDF/XML entscheidend!

- Lat. *res* ("Sache"), *facere* ("machen")
 - Vergegenständlichung, Konkretisierung
 - Eine Vorstellung, Aussage etc. zum Gegenstand einer Betrachtung machen
- In RDF: Aussagen über andere Aussagen machen
- "Peter sagt, Rom sei die Hauptstadt von Spanien."
- Umsetzung:
 - RDF-Statements werden auch als Ressourcen betrachtet
 - können dann Subjekt oder Objekt in anderen Statements werden

Reifikation in RDF



Codierung von Reifikation



Codierung von Reifikation



- In N3 – Variante mit benanntem Statement:

```
:tripell1 rdf:type rdf:Statement ;  
          rdf:subject :Rom ;  
          rdf:predicate :hauptstadt_von ;  
          rdf:object :Spanien .  
:Peter :sagt :tripell1 .
```

- In N3 – Variante mit unbenanntem Statement:

```
:Peter :sagt [  
  a rdf:Statement ;  
  rdf:subject :Rom ;  
  rdf:predicate :hauptstadt_von ;  
  rdf:object :Spanien .  
]
```

Codierung von Reifikation

- In RDF/XML (Variante ohne explizites rdf:Statement):

```
<rdf:Description rdf:about="http://www.geo.org/Rom">  
  <hat_hauptstadt rdf:ID="tripell">  
    <rdf:Description rdf:about="http://www.geo.org/Spanien" />  
  </hat_hauptstadt>  
</rdf:Description>  
  
<rdf:Description rdf:about="http://www.personen.de/Peter">  
  <sagt rdf:resource="#tripell" />  
</rdf:Description>
```

- Reifikationen können theoretisch auch geschachtelt werden
 - "Peter sagt, dass seine Schwester sagt, bei Wikipedia stehe, dass Rom die Hauptstadt von Spanien ist."

Semantische Prinzipien von RDF

- Im Web kann jeder alles über alles sagen
 - "Anybody can say anything about anything"
 - Das AAA-Prinzip (Allemang & Hendler)

- Dieses Prinzip gilt auch im Semantic Web!



Semantische Prinzipien von RDF

- Gleiche Dinge müssen nicht zwingend gleich benannt sein
 - :Munich :capitalOf :Bavaria .
 - :München :capitalOf :Bayern .
- Im Semantic Web gibt es keine eindeutigen Namen
 - Non-unique name assumption
- Folge: Nur, weil zwei Dinge unterschiedlich heißen, müssen sie nicht zwingend unterschiedlich sein!

Semantische Prinzipien von RDF

- Wir kennen (vermutlich) nicht alle Inhalte des Semantic Web
- Es kann also immer noch mehr Informationen zu einem Thema geben

- Dieses Prinzip heißt "Open World Assumption"

Semantische Prinzipien von RDF

- Wir kennen (vermutlich) nicht alle Inhalte des Semantic Web
- Es kann also immer noch mehr Informationen zu einem Thema geben

- Dieses Prinzip heißt "Open World Assumption"

RDF: Intuitive und tatsächliche Semantik

- Betrachten wir nochmals folgendes Beispiel:
 - :Julia :kindVon :Peter .
 - :Stefan :kindVon :Peter .
- Wie viele Kinder hat Peter?
- Intuitiv liest man hier: "Peter hat zwei Kinder".
- Er könnte aber auch drei oder mehr haben (oder auch nur eines, wie wir bereits gelernt haben)

Zusammenfassung RDF I



- Sprache zur Beschreibung von Informationen
 - graphenbasiert
 - Notationen: RDF/XML, N3
 - jeder Graph kann i.d.R. auf viele Arten notiert werden
- Besondere Sprachmittel
 - Leere Knoten
 - Mengen und Listen
 - Reifikation
- Semantik
 - Non-unique name assumption
 - Open world assumption

Ausblick auf Teil II

- Tools für RDF
- Gemeinsame Verwendung von RDF und HTML
- Beispiele für RDF "in der freien Wildbahn"

Vorlesung Semantic Web



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Vorlesung im Wintersemester 2012/2013

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering