

3. Beschreibung von Metadaten mit RDF und RDF-Schema

Lernziele:

- Verstehen, warum XML für die Wissensrepräsentation unzureichend ist,
- [Grundbegriffe der Wissensrepräsentation kennen](#),
- die Repräsentationskonstrukte von [RDF und RDF-Schema](#) und deren Notation in XML [kennen](#),
- mit diesen Repräsentationskonstrukten [Wissen repräsentieren können](#) und
- Konzepte von [RDF-Anfragesprachen einsetzen können](#).

97

Warum ist XML nicht ausreichend?

- XML ist eine [universelle Sprache](#) für das [Markup](#).
 - Es stehen [leistungsfähige Werkzeuge](#) bereit, um XML-basiert Daten zwischen Anwendungen auszutauschen, aber
- ☞ ... XML stellt [keine Mittel](#) bereit, um die [Semantik](#) der Daten zu definieren.
- ☞ ... die Interpretation der einzelnen Tags bleibt den Applikationen überlassen.

98

Repräsentation in XML

```
<vorlesung name="Graphentheorie">
  <dozent>Peter Becker</dozent>
</vorlesung>
```

Wir wollen den folgenden Sachverhalt repräsentieren:

Peter Becker ist Dozent der Veranstaltung Graphentheorie.

```
<dozent name="Peter Becker">
  <lehrt>Graphentheorie</lehrt>
</dozent>

<lehrangebot>
  <dozent>Peter Becker</dozent>
  <vorlesung>Graphentheorie</vorlesung>
</lehrangebot>
```

99

Weiteres Beispiel:

```
<academicStaffMember>Grigoris Antoniou</academicStaffMember>

<professor>Michael Maher</professor>

<course name="Discrete Mathematics">
  <isTaughtBy>David Billington</isTaughtBy>
</course>
```

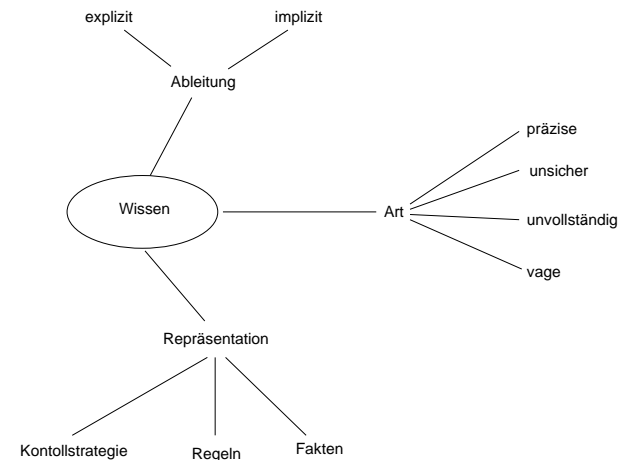
- XPath-Ausdruck um alle "academic staff members" zu ermitteln:

```
//academicStaffMember
```

100

- **Resultat:** nur Grigoris Antoniou
- obwohl
 - alle Professoren auch zu den “academic staff members” gehören und
 - Kurse nur von “academic staff members” gehalten werden.
- Um dies zu berücksichtigen, muß spezifisches Wissen der Anwendungsdomäne repräsentiert werden, z.B. in der Form
Alle Professoren gehören zu den “academic staff members”.

Arten von Wissen



Grundbegriffe der Wissensrepräsentation

Wissensrepräsentation (als Vorgang) ist nach Reimer:

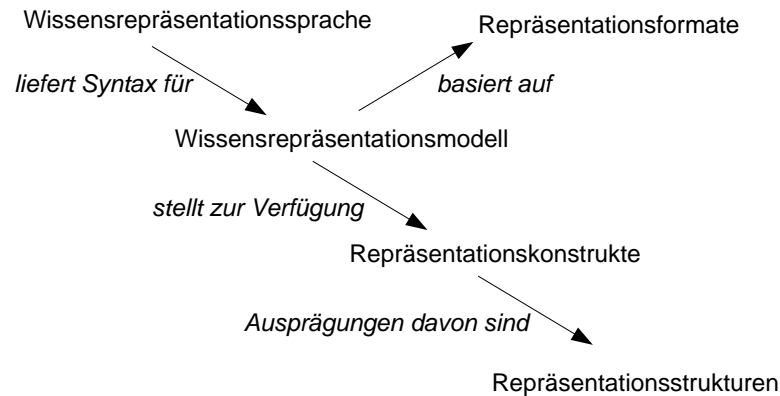
- das Aufschreiben von Symbolen — den *Repräsentationsstrukturen* — die in einer erkennbaren Weise einem Ausschnitt einer zu repräsentierenden Welt entsprechen.
- Es muß eine *Interpretationsvorschrift* vorliegen, die mindestens die Formulierung und Auswertung auf diesen Strukturen zuläßt.

Solche Strukturen zusammen mit der Interpretationsvorschrift heißen *Wissensrepräsentation* (als Gegenstand).

Wissensebenen (1)

- **kognitive Ebene** (z.B. Erfahrung von Experten, Arbeitsanweisungen)
 - **Repräsentationsebene** (z.B. Relationen, Prädikatenlogik)
 - **Implementierungsebene** (z.B. Datenbankabfragen, Prolog-Statements)
- ☞ Bei der **Wissensverarbeitung** und der **Künstlichen Intelligenz** stehen die Repräsentationsebene und die Implementierungsebene im Vordergrund (Schließen der KI-Lücke).
- ☞ Beim **Wissensmanagement** stehen die kognitive Ebene und die Repräsentationsebene im Vordergrund.

Wissensrepräsentationsmodell



105

- Ein **Repräsentationsformat** ist eine Klasse gleichartiger Repräsentationsstrukturen (z.B. netzartige Strukturen mit Konzepten und Beziehungen).
- Ein **Wissensrepräsentationsmodell** basiert auf einem oder mehreren Repräsentationsformaten und legt die **konstruierbaren Repräsentationsstrukturen** in formaler Weise fest, sowie die darauf ausführbaren Operationen.
- Die zur Verfügung stehenden Grundtypen von Repräsentationsstrukturen heißen **Repräsentationskonstrukte**.
- Mittels einer Syntax legt eine **Wissensrepräsentationssprache** fest, wie die Repräsentationsstrukturen notiert werden können.

106

Der Begriff eines **Wissensrepräsentationsmodells** ist analog zum Begriff eines **Datenmodells**, wie er im Datenbankbereich benutzt wird.

107

Wissensebenen (2)

- **Linguistische Ebene:** Ein Konstrukt steht für eine grammatikalische Konstruktion, für ein Wort oder für eine Phrase in einer natürlichen Sprache.
- **Konzeptuelle Ebene:** elementare **domänenabhängige** Konzepte, unabhängig von der natürlichsprachlichen Formulierung eines Sachverhalts.
- **Epistemologische Ebene:** Komplexere Konstrukte, die nicht auf ein einzelnes logisches Konstrukt abgebildet werden können, die aber **domänenunabhängig** sind (z.B. Vererbung).

108

- **Logikebene:** Konstrukte einer mathematischen Logik
- **Implementierungsebene:** Datenstrukturen, Algorithmen

Ein *epistemisches Primitiv* steht für eine Klasse gleichartiger Sachverhalte, die so allgemein sind, daß ihr Auftreten nicht an einen bestimmten Diskursbereich gebunden ist. Epistemische Primitive sind also domänenunabhängig.

Zum *Grundvokabular* gehören Repräsentationsstrukturen, die auf der konzeptuellen Ebene angesiedelt sind (domänenabhängig) und in gleicher Form mehrfach verwendet werden können.

Beispiel: Handlungen:

- haben immer einen ausführenden **Agenten**,
- ein von der Handlung **betroffenes Objekt**,
- **Zeit**,
- **Ort**.

☞ Verwendet man für alle Handlungsrepräsentationen dasselbe Grundvokabular, werden verschiedene solcher Repräsentationen miteinander vergleichbar.

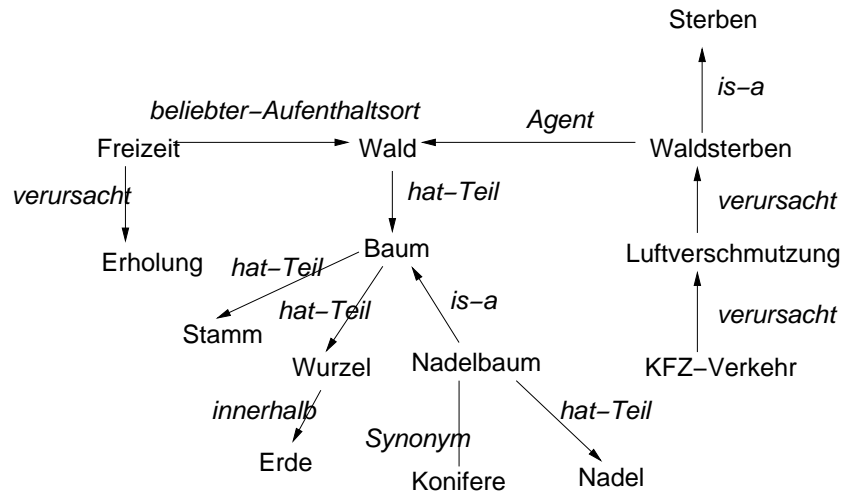
Semantische Netze

- gehen zurück auf Modelle des menschlichen Gedächtnisses in der Kognitionspsychologie
- zugrundeliegende Annahme: Konzepte (Begriffe), die in Beziehung stehen, werden durch Strukturen repräsentiert, die miteinander verbunden sind.
- **assoziative Beziehungen, Assoziationsmodelle**
- Durch Erinnerungsvorgänge findet entlang der Beziehungen eine Ausbreitung statt.

- Veranschaulichung durch Graphen bzw. Netze, Knoten sind Konzepte (Begriffe), Kanten beschreiben die assoziativen Beziehungen zwischen den Konzepten.

Arten von Assoziationsmodellen:

- **Assoziatives Netzwerk:** nur ein Kantentyp, Kanten haben einen numerischen Wert für die Assoziationsstärke.
- **semantisches Netzwerk:** verschiedene Kantentypen für verschiedene Arten der Beziehung, kein numerischer Indikator für die Assoziationsstärke.



113

- epistemische Primitive: is-a, verursacht, hat-Teil, Synonym
- Grundvokabular: innerhalb, beliebter-Aufenthaltsort, Agent
- transitive Kanten, Vererbung

114

RDF: Grundlegende Konzepte

- *Resource Description Framework (RDF)*
- RDF ist in Verbindung mit RDF-Schema ein *Wissensrepräsentationsmodell/Datenmodell*.
- RDF kann *unabhängig von XML* genutzt werden. Die *RDF/XML-Syntax* beschreibt, wie RDF-Repräsentationen in XML notiert werden.
- *RDF ist domänenunabhängig*. RDF Schema kann dazu genutzt werden, eine Terminologie/ein Grundvokabular zu definieren.

115

Ressourcen

- *Ressourcen* sind die Dinge, über die wir Wissen repräsentieren möchten, z.B. ...
- ... Autoren, Bücher, Verlage, Orte, Menschen, Hotels, ...
- Jede Ressource wird durch einen *URI* identifiziert. Dies kann, muß aber kein URL sein.
- Die *Identifikation ist entscheidend*, nicht der Zugriff auf die Ressource, z.B. ISBN.

116

Properties

- *Properties* sind eine spezielle Art einer Ressource,
- sie beschreiben **Beziehungen** zwischen oder **Eigenschaften** von Ressourcen,
- z.B. “geschrieben von”, “verheiratet mit”, “Alter”.
- Die **Identifikation** einer Property erfolgt ebenfalls über einen **URI**.
- Dies führt zu einer **Eindeutigkeit** der Beziehungen.

117

Statements

- *Statements* repräsentieren Aussagen über Ressourcen.
- **Subjekt-Prädikat-Objekt-Tripel**
- Das Objekt kann wieder eine Ressource sein oder ist ein einfacher Wert (**literal value**).

Beispiel:

<http://www.example.org/index.html> has a **creator** whose value is John Smith.

118

- Statements können als **Tripel der Form (Subjekt,Prädikat,Objekt)** formuliert werden.

(<http://www.example.org/index.html>, creator, “John Smith”)

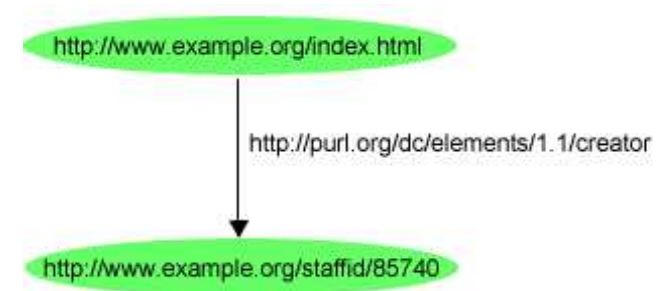
- In der Logik entspricht dies einem **zweistelligen Prädikat**:

creator(<http://www.example.org/index.html>,”John Smith”)

- RDF bietet **ausschließlich** zweistellige Prädikate zur Repräsentation an!
- Andere Notation: **graphisch!** Dies entspricht den **semantischen Netzen**.

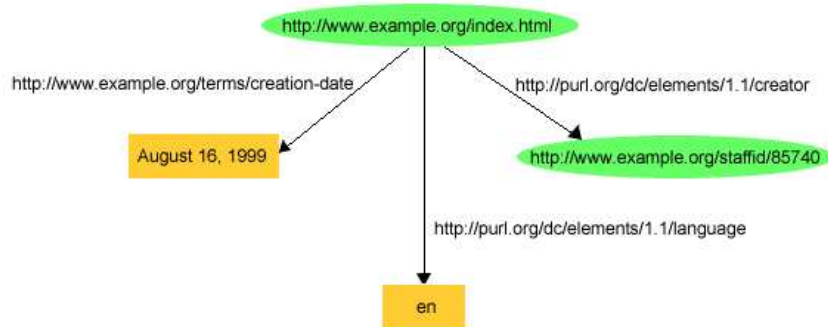
119

- **Subjekt:** <http://www.example.org/index.html>
- **Prädikat:** <http://purl.org/dc/elements/1.1/creator>
- **Objekt:** <http://www.example.org/staffid/85740>



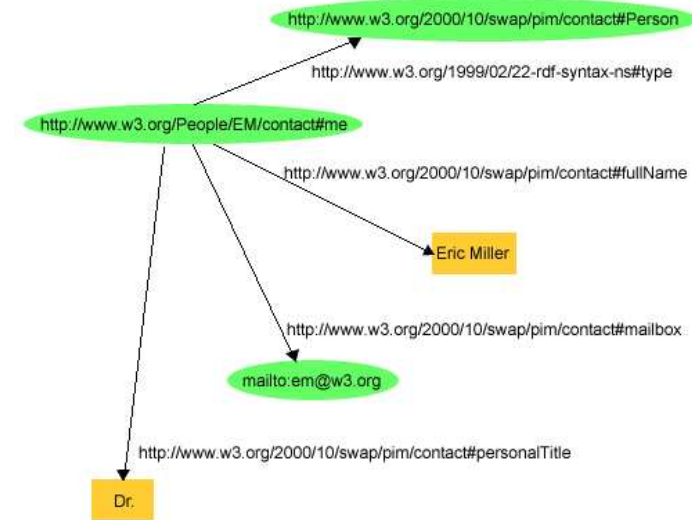
120

Gruppen von **zusammengehörenden Statements** können im Graphen ebenfalls zusammengefasst werden:



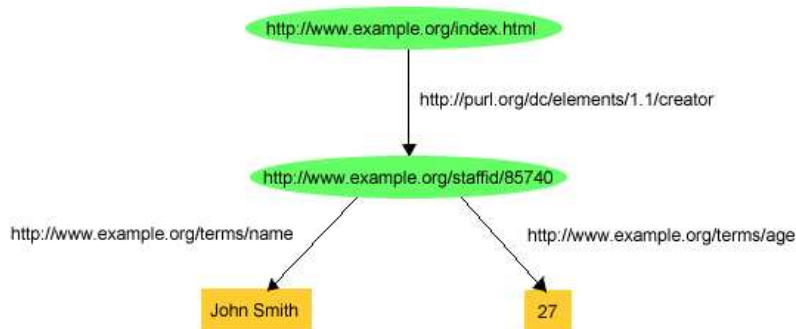
“August 16, 1999” und “en” sind sogenannte *plain literals*.

121



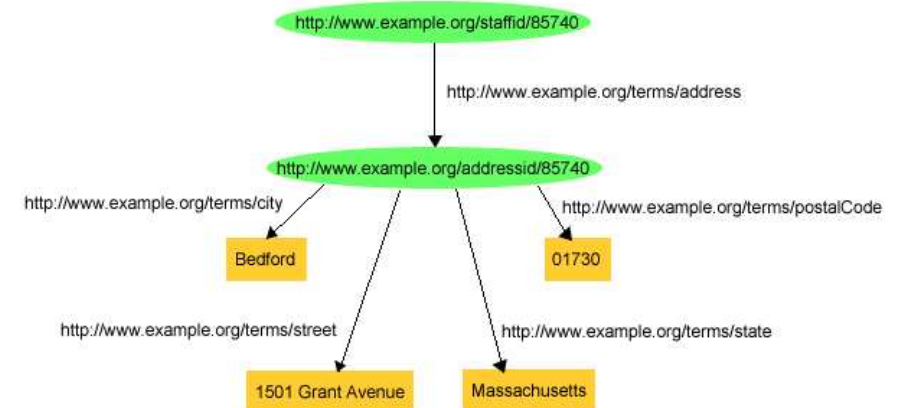
123

Ressourcen können **sowohl als Objekte als auch als Subjekte** fungieren.



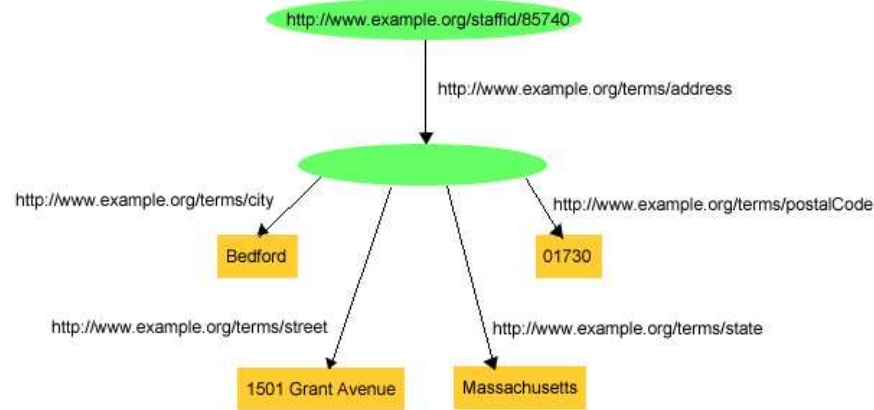
122

Darstellung einer strukturierten Information (Adresse) mit Hilfe von einzelnen Statements:



124

Definition eines **blank node** bzw. einer **anonymous resource** zur Darstellung der Adresse:



125

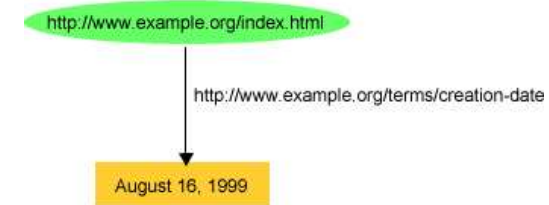
Literale können mit Typinformation versehen werden (**typed literals**).



Als Datentypen bieten sich die Typen aus XML-Schema an.

126

RDF/XML-Syntax



```

<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:exterm="http://www.example.org/terms/">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <exterm:creation-date>August 16, 1999</exterm:creation-date>
  </rdf:Description>
</rdf:RDF>
  
```

127

Ein **rdf:RDF**-Element kann beliebig viele Statements enthalten.

```

<?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/"
  xmlns:exterm="http://www.example.org/terms/">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <exterm:creation-date>August 16, 1999</exterm:creation-date>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.example.org/index.html">
    <dc:language>en</dc:language>
  </rdf:Description>
</rdf:RDF>
  
```

128

Statements mit dem gleichen Subjekt können in einem `rdf:Description`-Element zusammengefasst werden.

```
<?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/"
  xmlns:exterms="http://www.example.org/terms/">

  <rdf:Description rdf:about="http://www.example.org/index.html">
    <exterms:creation-date>August 16, 1999</exterms:creation-date>
    <dc:language>en</dc:language>
    <dc:creator rdf:resource="http://www.example.org/staffid/85740"/>
  </rdf:Description>

</rdf:RDF>
```

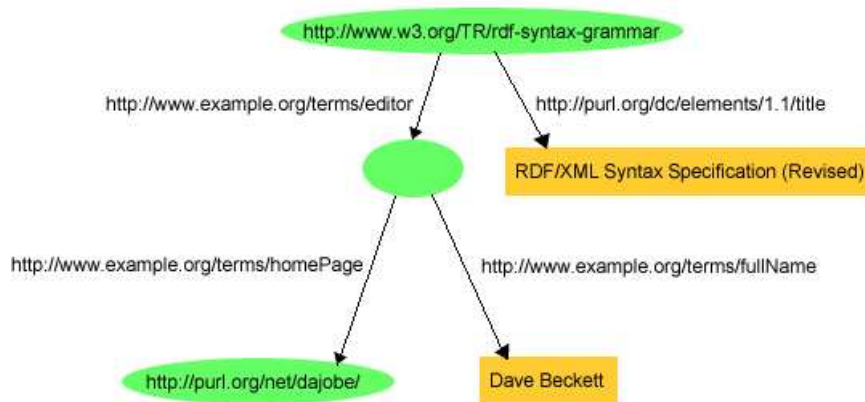
```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:exterms="http://example.org/stuff/1.0/">

  <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
    <dc:title>RDF/XML Syntax Specification (Revised)</dc:title>
    <exterms:editor rdf:nodeID="abc"/>
  </rdf:Description>

  <rdf:Description rdf:nodeID="abc">
    <exterms:fullName>Dave Beckett</exterms:fullName>
    <exterms:homePage rdf:resource="http://purl.org/net/dajobe/">
  </rdf:Description>

</rdf:RDF>
```

Darstellung von **blank nodes**:



Nutzung eines **typed literal**:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:exterms="http://www.example.org/terms/">

  <rdf:Description rdf:about="http://www.example.org/index.html">
    <exterms:creation-date rdf:datatype="
      http://www.w3.org/2001/XMLSchema#date">1999-08-16
    </exterms:creation-date>
  </rdf:Description>

</rdf:RDF>
```

Und mit Definition eines XML-Entities:

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:exterms="http://www.example.org/terms/">

  <rdf:Description rdf:about="http://www.example.org/index.html">
    <exterms:creation-date rdf:datatype="&xsd:date">1999-08-16
    </exterms:creation-date>
  </rdf:Description>

</rdf:RDF>
```

133

Organisation von URIs

- Wenn man Ressourcen definieren/einführen möchte, verwendet man das Attribut `rdf:ID` statt `rdf:about` innerhalb eines `rdf:Description`-Elements.
- Das Argument zu ID ist eine lokal eindeutige Kennung (*fragment identifier*).
- Die **URI der Ressource** ergibt sich dann aus der Kombination
 - der **URI des Dokuments**, in dem das `rdf:Description`-Element enthalten ist,
 - dem **Trennzeichen #**
 - und der **lokalen Kennung**.

134

Beispiel: Produktkatalog in `http://www.example.com/2002/04/products`

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:exterms="http://www.example.com/terms/">
  <rdf:Description rdf:ID="item10245">
    <exterms:model rdf:datatype="&xsd:string">Overnighter</exterms:model>
    <exterms:sleeps rdf:datatype="&xsd:integer">2</exterms:sleeps>
    <exterms:weight rdf:datatype="&xsd:decimal">2.4</exterms:weight>
    <exterms:packedSize rdf:datatype="&xsd:integer">784</exterms:packedSize>
  </rdf:Description>
  ... other product descriptions ...
</rdf:RDF>
```

Die URI des beschriebenen Produkts als Ressource lautet dann:

`http://www.example.com/2002/04/products#item10245`

135

- Auf diese Ressource kann in anderen `rdf:Description`-Elementen verwiesen werden.
- **URI als Objekt** mittels `rdf:resource`.
- **URI als Subjekt** mittels `rdf:about`, d.h. es können weitere Eigenschaften der Ressource an anderer Stelle angegeben werden.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:sportex="http://www.exampleRatings.com/terms/">

  <rdf:Description rdf:about="http://www.example.com/2002/04/products#item10245">
    <sportex:ratingBy rdf:datatype="&xsd:string">Richard Roe</sportex:ratingBy>
    <sportex:numberStars rdf:datatype="&xsd:integer">5</sportex:numberStars>
  </rdf:Description>
</rdf:RDF>
```

136

- Manchmal möchte man eine **Basis-URI** explizit angeben, z.B. wenn Web-Seiten gespiegelt werden bzw. über verschiedene URLs zugreifbar sind.
- Innerhalb des `rdf:RDF`-Elements kann mit dem Attribut `xml:base` eine URI-Basisadresse angegeben werden.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:externs="http://www.example.com/terms/"
  xml:base="http://www.example.com/2002/04/products">
  <rdf:Description rdf:ID="item10245">
    <externs:model rdf:datatype="&xsd:string">Overnighter</externs:model>
    <externs:sleeps rdf:datatype="&xsd:integer">2</externs:sleeps>
    ...
  </rdf:Description>
</rdf:RDF>
```

137

Typzuordnung in RDF

- Die in RDF verwendeten Vokabulare definieren u.a. auch Kategorien/Klassen.
- Mit dem `rdf:type`-Element hat man die Möglichkeit, eine Ressource einer Klasse zuzuordnen.
- Vergleichbar zur OO-Programmierung: Man gibt an, daß die Ressource eine **Instanz** einer bestimmten Kategorie/Klasse ist.
- Ressourcen, die Klassen zugeordnet sind, heißen *typed nodes* bzw. *typed node elements*.
- Die Klassen werden definiert mit Hilfe von RDF Schema (hierzu später mehr).

138

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:externs="http://www.example.com/terms/"
  xml:base="http://www.example.com/2002/04/products">

  <rdf:Description rdf:ID="item10245">
    <rdf:type rdf:resource="http://www.example.com/terms/Tent"/>
    <externs:model rdf:datatype="&xsd:string">Overnighter</externs:model>
    <externs:sleeps rdf:datatype="&xsd:integer">2</externs:sleeps>
    <externs:weight rdf:datatype="&xsd:decimal">2.4</externs:weight>
    <externs:packedSize rdf:datatype="&xsd:integer">784</externs:packedSize>
  </rdf:Description>

  ...other product descriptions...

</rdf:RDF>
```

139

☞ Eine Ressource kann **mehr als einer Klasse** zugeordnet werden.

☞ Kurzschreibweise: **Klassenname** statt `rdf:Description` sorgt für eine implizite Zuordnung der Ressource zu der genannten Klasse.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:externs="http://www.example.com/terms/"
  xml:base="http://www.example.com/2002/04/products">

  <externs:Tent rdf:ID="item10245">
    <externs:model rdf:datatype="&xsd:string">Overnighter</externs:model>
    ...
  </externs:Tent>
</rdf:RDF>
```

140