VSR | EDU

XML

XML

Prof. Dr.-Ing. Martin Gaedke
Technische Universität Chemnitz
Fakultät für Informatik
Professur Verteilte und Selbstorganisierende Rechnersysteme

http://vsr.informatik.tu-chemnitz.de

Problem: XML and Data

- "Understanding" of data – Example:

```
<zoo name="BerlinZoo">
    <animal name="Knut">Bear</animal>
</zoo>
```

- Does <animal name="Knut"> describe the same animal?

```
<BerlinZoo>
    <bear>Knut</bear>
</BerlinZoo>
```

- What about…

```
<Bears>
    <Bear><name>Knut</name><location id="BerlinZoo"/></Bear>
</Bears>
```

What is needed? (technically)

- For machines to be able to process data on the Web, the following concepts (and the related technologies) are needed:
  - Unique resource names (possibly also for objects in the real world): **URIs**
  - Data model – to link, describe and access resources: **RDF**
  - "Access" / search: **SPARQL**
  - Definition of vocabularies: **RDF**, **OWL, SKOS**
  - Inference logic: **OWL**, **Rules**
- "Semantic Web" is an extension (not a replacement!) of the Web, providing an infrastructure for integration of data on the Web.

Chapter 15

RESOURCE DESCRIPTION FRAMEWORK (RDF)

RDF - Overview

- Resource Description Framework (RDF)
  - W3C Recommendation
  - http://www.w3.org/TR/rdf-primer/
  - http://www.w3.org/RDF/
- RDF is a data model
  - Enables data description
  - Data on data – metadata
  - RDF enables description of metadata in machine-readable form by means of the according notation

Introduction

- Linking / connection of data in detail…
  - "I have a calendar."
  - "This is my CV."
  - These phrases are understandable for humans, but for machines…
- Statement analysis for machines:
  - <I>, <have>, <calendar>
  - <This>, <is>, <my>, <CV>
  - Is the connection <I> – <calendar> the same as for <I> – <CV>?
- Idea: Statements are connections
  - Connections assign things to other things
  - Connections are data on data
  - Connections should be named
  - <Data> – <CONNECTION> – <Data>
  - →Represent statements as triples…

RDF Triple

- **RDF triple**: Describes a statement in form of a relationship (P) between a subject (S) and an object (O)
  - **Statement** describes a thing S, where a property P is provided with a value of O
  - **RDF triple (S,P,O)**
    - **Subject – Predicate – Object**
    - Or, technically: **Subject – Property – Object**

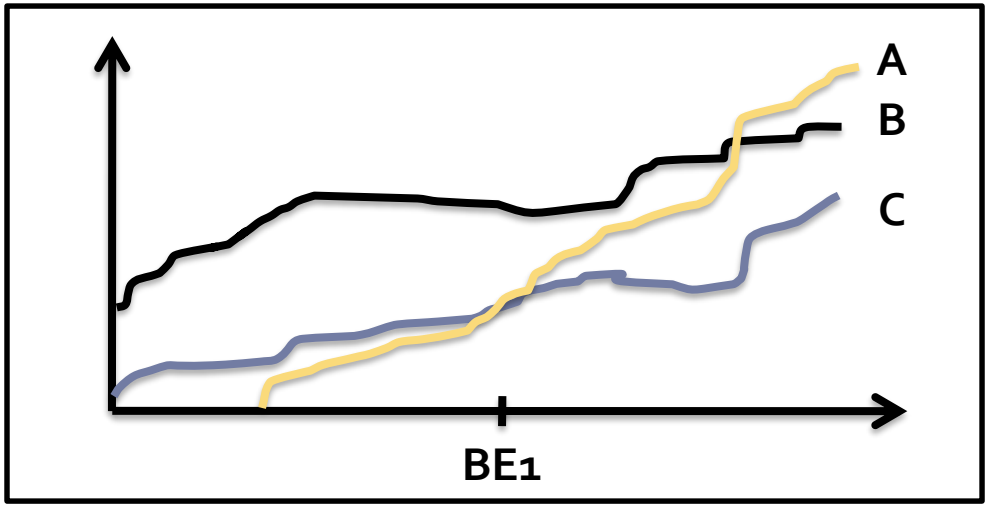
- **RDF triple properties**
  - **S**, **P** are URIs, **O** is a URI or a literal
  - Conceptually, P connects: S and O (directed description)
  - RDF is a model for such triples

RDF Triple – First Example

- Statement: "Gaedke has (a) CV"
  - S,P,O resources: S: <http://gaedke.com/>, P: <http://…/has>, O: <http://…/CV>
- Statement as an RDF triple
  - <http://gaedke.com/>,<http://…/has >,<http://…/CV>
- Resources can be ***any*** URIs
  - http://example.org/people/gaedke.html
  - http://example.org/people/g.html#fragmentId
  - http://example.org/#xpointer(id('g12,))

RDF Triple – Second Example

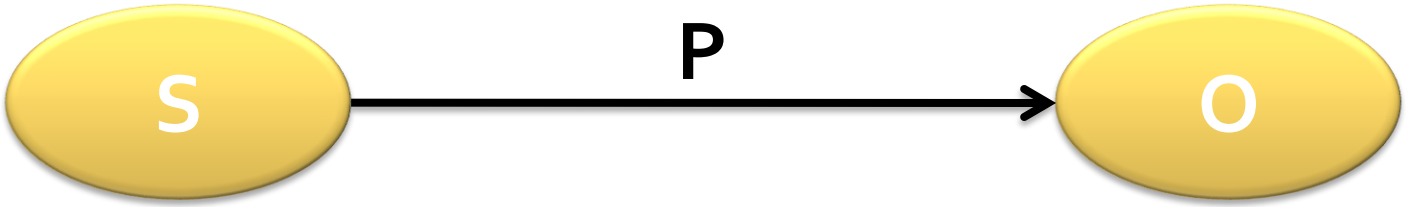
- If this image is encoded in SVG (an XML application), then any element can be addressed by a URI
- Detailed statements about the image are possible



- **<http://.../graph.svg#xpointer(...)> <http://.../shows.html> <http://biz.../def#BreakEven>**

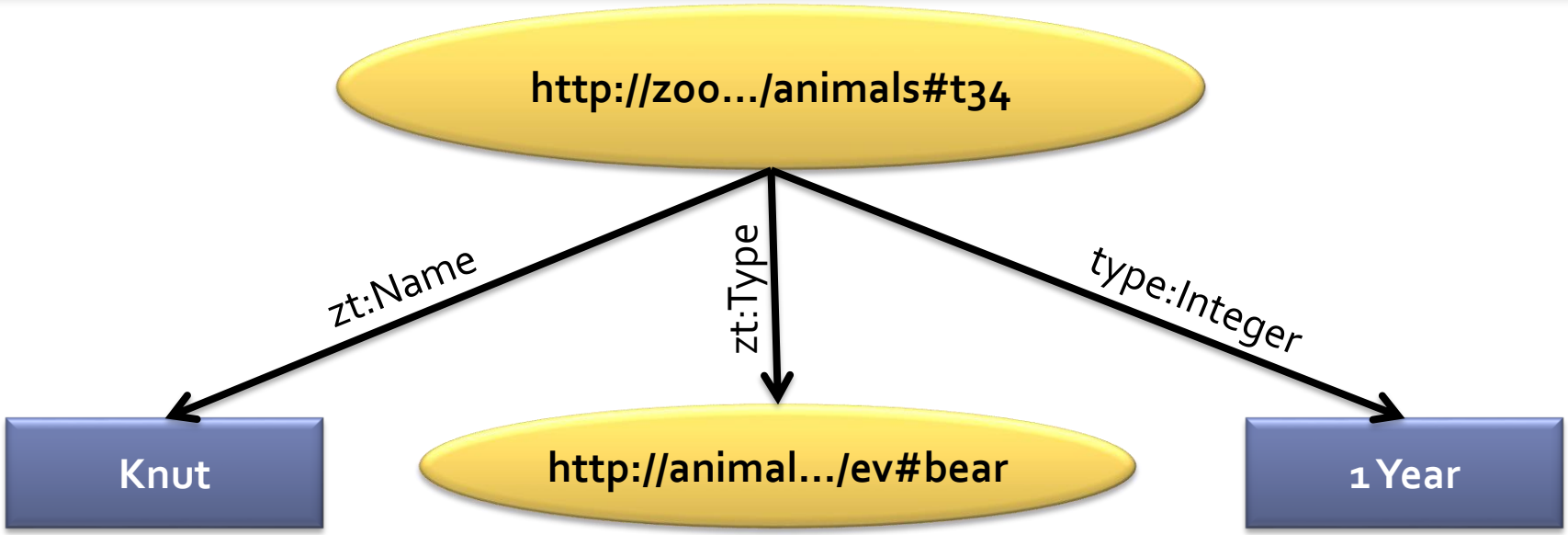
RDF as a Graph

- Triple (S,P,O) can be seen as an edge of a graph
  - Subjects and objects are nodes in a graph
  - Properties are edges



- Note:
  - RDF deals with graphs

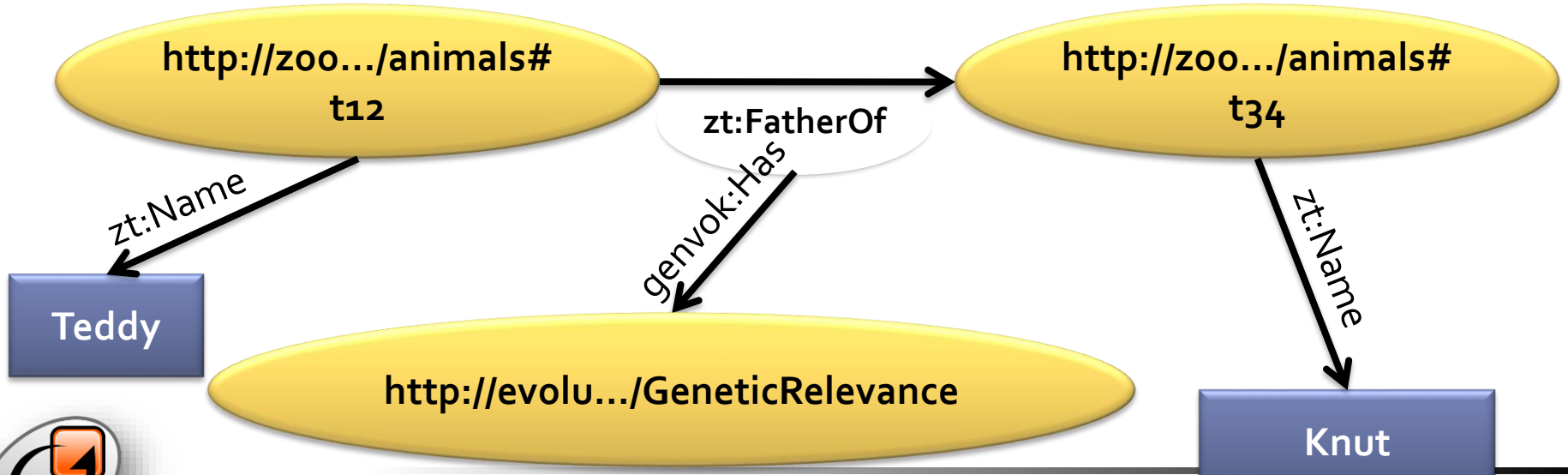
Simple Example of RDF Graphs



- Animal 34 has a name Knut.
  (RDF concept: literal)
- He is 1 year old.
  (RDF concept: literal)
- He is a bear.
  (RDF concept URI-based vocabulary)

Statements about Statements

- Each element of a statement can be some element of another statement.
  - Example:
    Subject of a statement can be an object of another statement
  - Predicate of a statement can be a subject of another statement
- Multiple statements can be viewed as an RDF graph, a so-called semantic network



RDF Machine-Readable…

- RDF is a data model…machine-readable implementations include RDF/XML, Notation 3 (N3), Turtle, etc.
- Example **RDF in XML**
  - Each rdf:Description element describes a resource (subject)
  - Nested elements are **properties**
  - Attribute or content of a property element describes an **Object**

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Vatervon resource="http://zoo.../tiere#t34"/>
  </rdf:Description>
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Name>Teddy</zt:Name>
  </rdf:Description>
</rdf:RDF>
```

RDF/XML

- RDF/XML allows various syntax in XML – the meaning is preserved

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Vatervon resource="http://zoo.../tiere#t34"/>
  </rdf:Description>
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Name>Teddy</zt:Name>
  </rdf:Description>
```


```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Vatervon resource="http://zoo.../tiere#t34"/>
    <zt:Name>Teddy</zt:Name>
  </rdf:Description>
</rdf:RDF>
```


RDF(S) in Turtle

- RDF allows various syntax, here's a Turtle example:

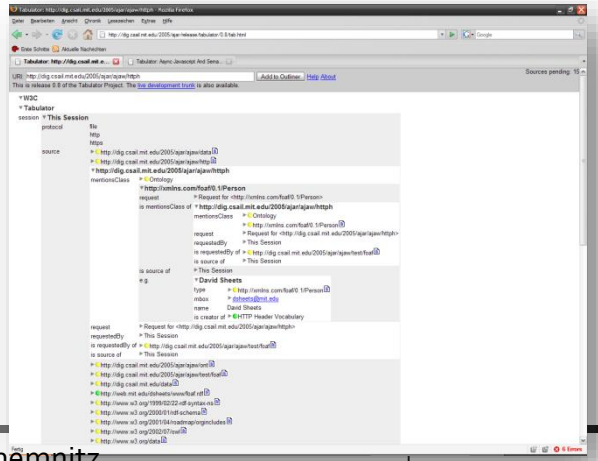
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix zt: <http://example.org/evkonzept#>.
<http://zoo.../animals#t12>
    zt:FatherOf zt:t34;
    zt:Name "Teddy".
```

- Compared to XML/RDF

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../tiere#t12">
    <zt:Vatervon resource="http://zoo.../tiere#t34"/>
    <zt:Name>Teddy</zt:Name>
  </rdf:Description>
</rdf:RDF>
```

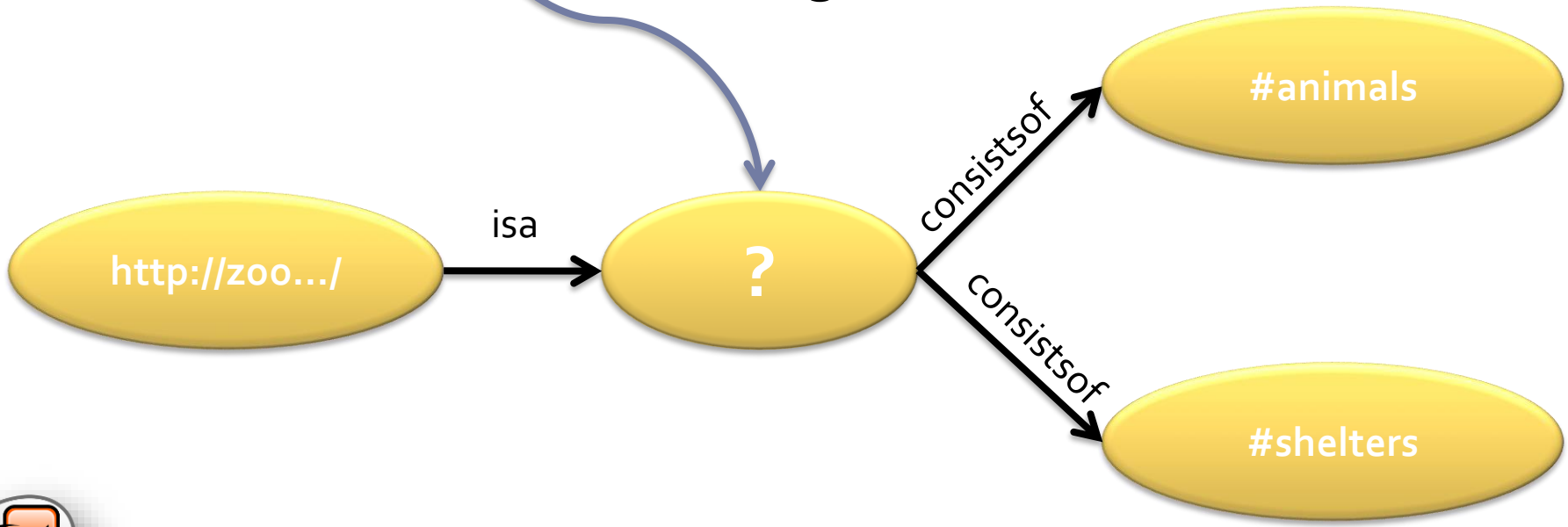
RDF and URIs

- Statements are flexible:
  (S, P, O) → (S, P, (S', P', O'))
- URIs enable statements about external resources
- External resource statements can be merged to produce a new statement graph
- Merge example… "tabulator project"
  - http://www.w3.org/2005/ajar/tab



Internal Nodes

- What if you want to make a statement about a thing?
  - Zoo is a thing, which consists of animals and shelters for those animals.
  - What is the URI of a thing?



Internal Nodes – rdf:ID

- Anything needs a name, even a thing…
- Approach:
  - **Name can be given** by means of an identifier: **rdf:ID** *(in a true sense of a URI)*
  - Not to be confused with xml:id (semantically different term)

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../">
    <zt:istEin rdf:resource="#Ding"/>
  </rdf:Description>

  <rdf:Description rdf:ID="Ding">
    <zt:bestehtAus rdf:resource="#tiere"/>
    <zt:bestehtAus rdf:resource="#behausungen"/>
  </rdf:Description>
</rdf:RDF>
```

Internal Nodes – Blank Nodes

- Anything needs a name, even a thing…
- Approach:
  - By means of an internal identifier **rdf:nodeID** (invisible outside of the resource, so, no URI)
  - Is an internal identifier
  - Caution when merging various RDF resources – i.e. w.r.t. uniqueness/unambiguity of nodeID

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../">
    <zt:istEin rdf:nodeID="88"/>
  </rdf:Description>

  <rdf:Description rdf:nodeID="88">
    <zt:bestehtAus rdf:resource="#tiere"/>
    <zt:bestehtAus rdf:resource="#behausungen"/>
  </rdf:Description>
</rdf:RDF>
```

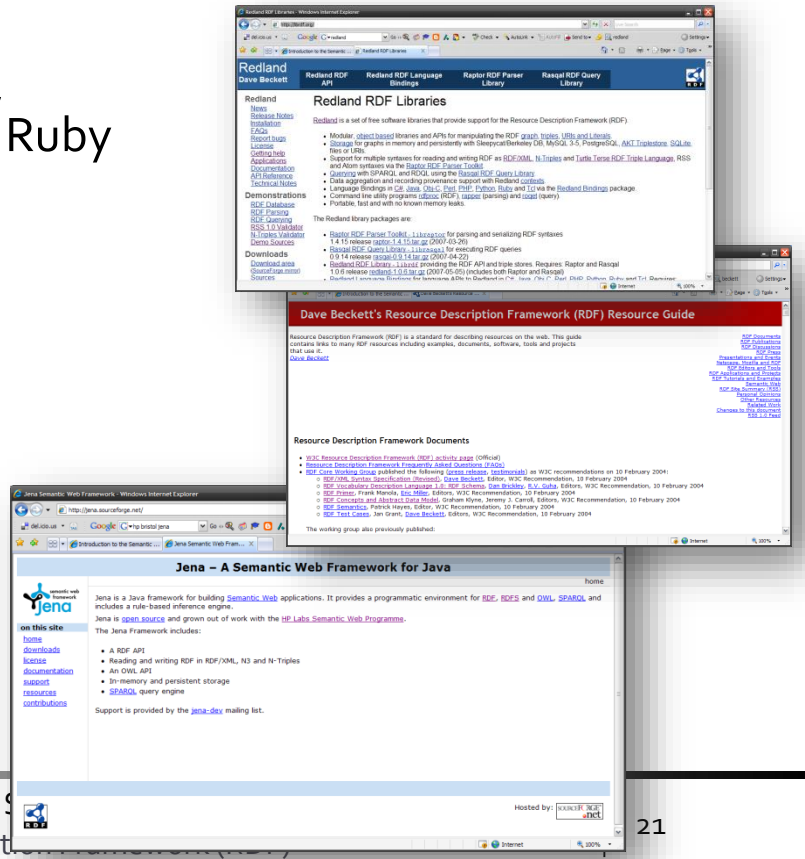
Internal Nodes – Blank Nodes (2)

- Names are smoke and mirrors, but structure…
- Approach:
  - Use system support. System creates an identifier internally, the according description is determined by the XML structure of the statements
  - Logically, this approach represents existential statements ("There is a resource, which")

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description about="http://zoo.../">
    <zt:istEin>
      <rdf:Description>
        <zt:bestehtAus rdf:resource="#tiere"/>
        <zt:bestehtAus rdf:resource="#behausungen"/>
      </rdf:Description>
    </zt:istEin>
  </rdf:Description>
</rdf:RDF>
```

RDF in Action

- Libraries, support tools for the RDF approach already exist, for example:
  - Dave Beckett (Editor RDF- W3C-Standard),
    - Redland RDF Libraries
      - http://librdf.org/
      - Support for different languages, like C#, Java, Perl, PHP, Python, Ruby
    - RDF Resource Guide
      - http://planetrdf.com/guide/
  - HP Labs: Java + Jena
    - http://jena.sourceforge.net/
  - Various RDF TripleStores… Editors… and much more…



RDF – Final Remarks

- RDF – Very simple, very flexible, very powerful
  - Subjects, relations and concepts shape RDF modelling/description
  - Relations are explicit (also while merging)
  - Data model is available in different syntactic forms
- But:
  - No vocabulary definition…

Chapter 16

RDF SCHEMA

Introduction

- Statement reading is easy, but
  - how should they be processed by a program
  - (as in, how should they be understood)?
- Programs must be capable of understanding RDF vocabulary
- One requires something similar to the XML Schema, some kind of an RDF Schema
- To be more specific: **RDF Vocabulary Description Language**

RDF Vocabulary Description Language

- **RDF Vocabulary Description Language 1.0: RDF Schema**
  - W3C Recommendation 10 February 2004
  - Commonly known as **RDF Schema** *or, in short, vocabulary*
  - Specification of the six RDF specifications
  - **xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"**
- Defines rules for
  - Especially: Classes, Properties
  - Also: Domain, Range
- It also provides for reification (objectification, concretisation) of the original RDFMS (RDF Model and Syntax)
  - Defines: Statement, subject, predicate, object
  - As well as helper properties, like, for example, seeAlso

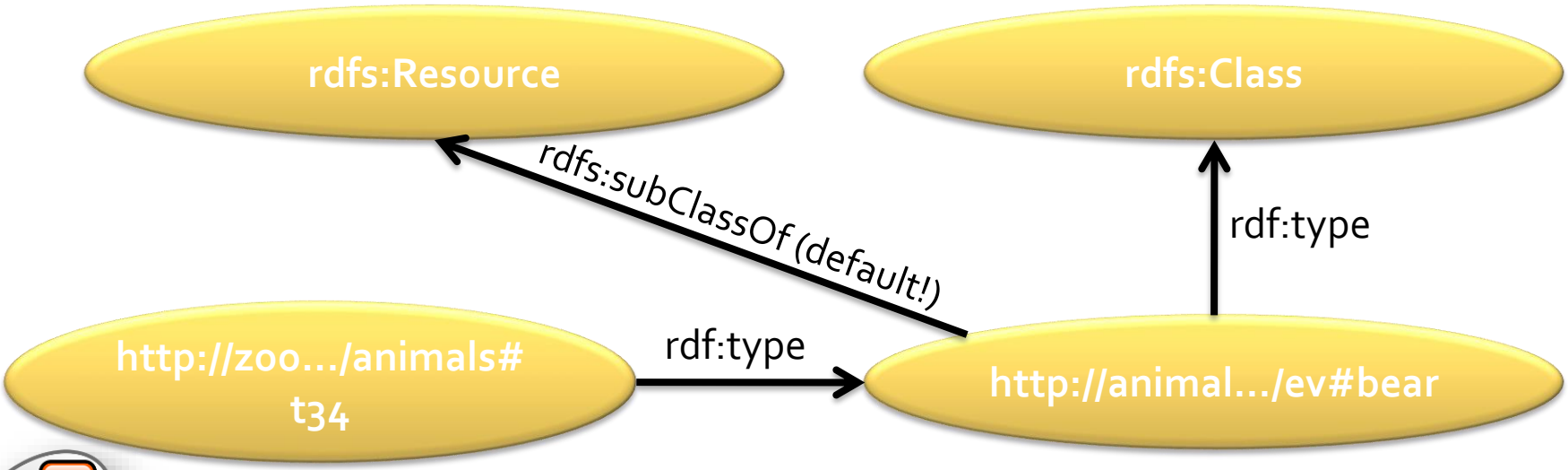
RDFS – Classes (1)

- RDF Schema Classes & Co.
  - Resources can be divided into groups, which are referred to as Classes.
  - Members of such a class are instances, which are typically identified by RDF-URI references and described by properties
  - rdf:type can indicate whether a resource is an instance of a class

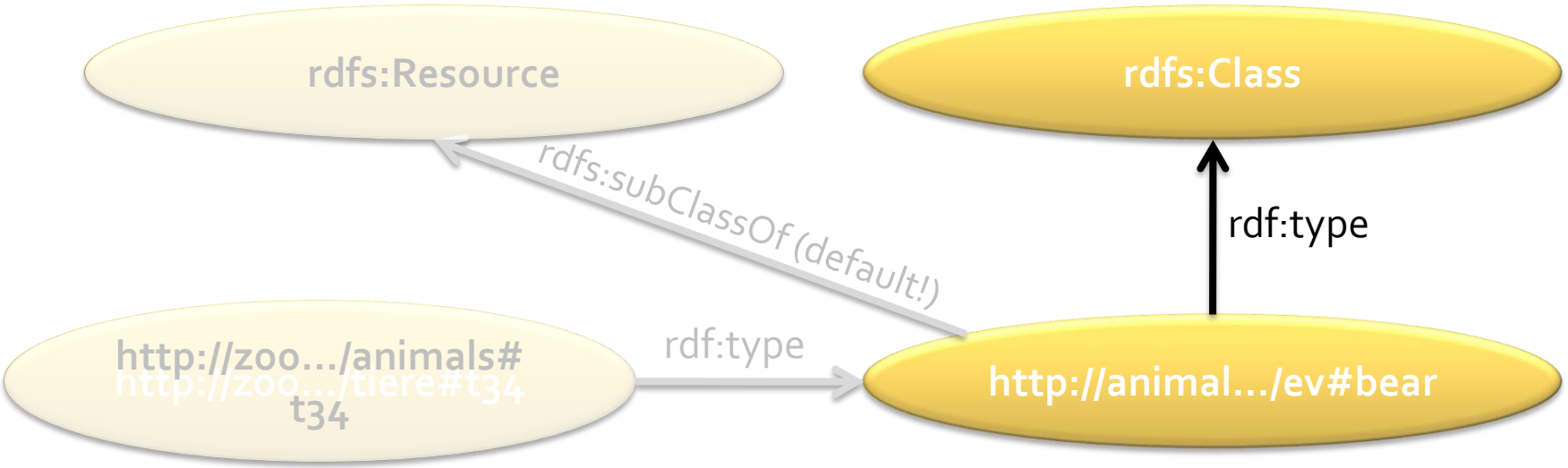
[Translated (twice) excerpt from the RDF Schema]

RDFS – Classes (2)

- So: Relationships between resources and classes
  - Typing: Individual belongs to a class (Knut and Teddy belong to class Bears)
  - Subclassing: Class is an instance of another one (Bears belong to class Mammals)
  - Note: Differentiate **rdf:** and **rdfs:**



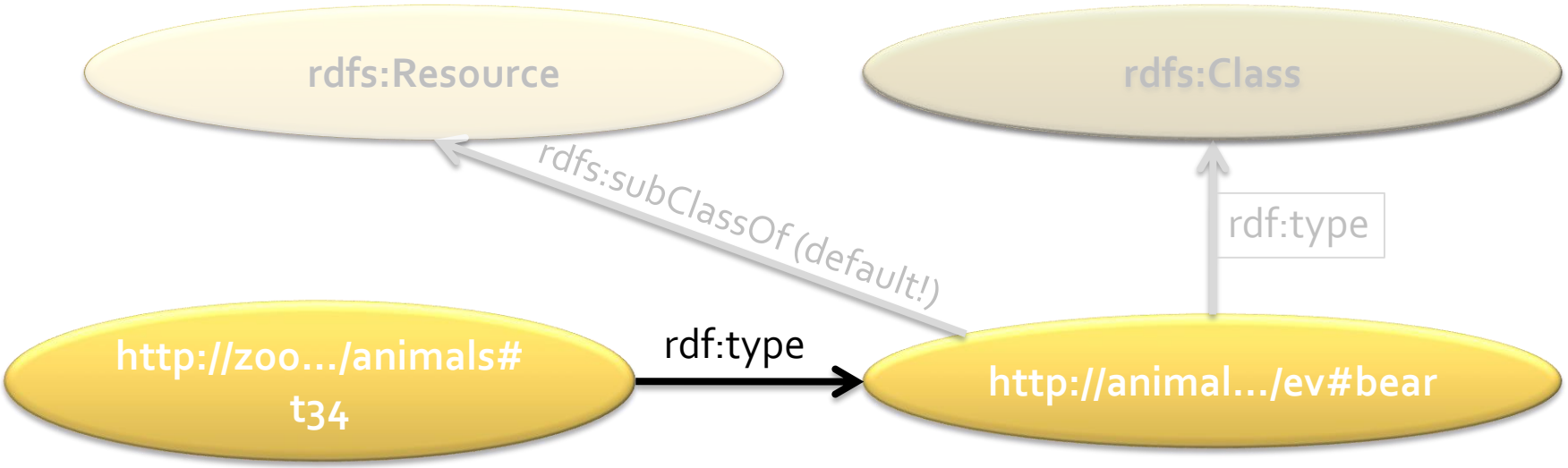
RDFS – Classes (3)



- Data types of an application
  - Describes terminological knowledge

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description rdf:ID="baer">
        <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  </rdf:Description>
</rdf:RDF>
```

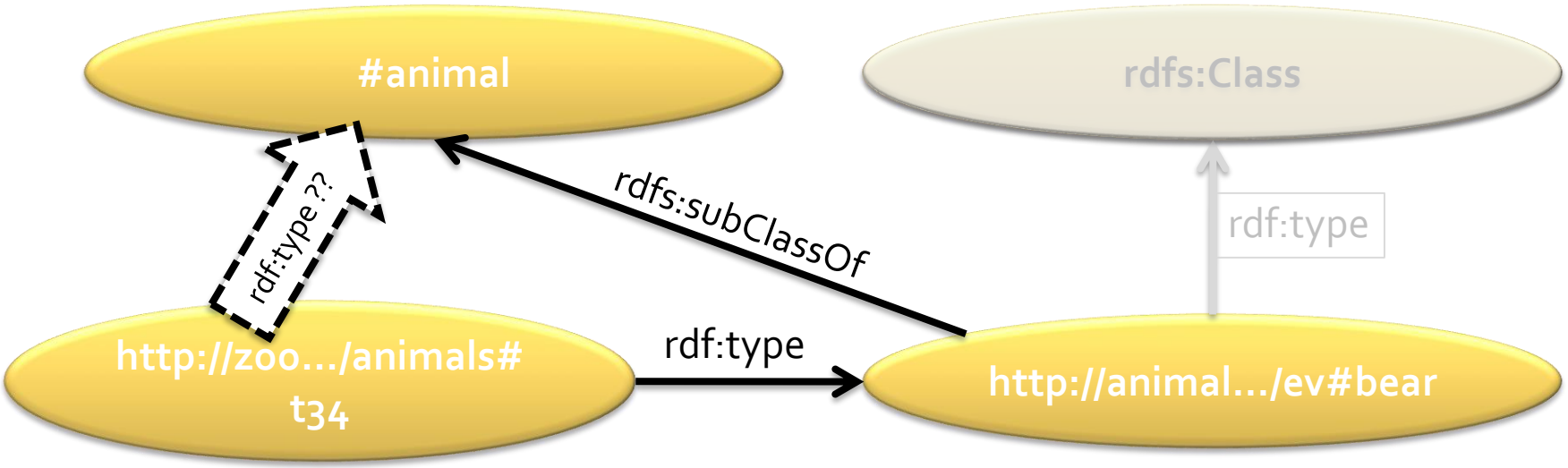
RDFS – Classes (4)



- Instances of an application
  - Separation of terminological and assertional knowledge

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:zt="http://example.org/evkonzept#">
  <rdf:Description rdf:about="#t34">
    <rdf:type rdf:resource="http://example.org/evkonzept#Baer"/>
  </rdf:Description>
</rdf:RDF>
```

RDFS – Classes (5)



- Is <#t34> <rdf:type> <#animal>?
  - Is not contained in RDF data, but can be concluded (inference)
  - Good RDF systems deliver such information upon any request to subject <#t34>
  - rdf:type is, therefore, a frequently used property
  - Note: rdf:type is a property, i.e. a subject can be described by multiple rdf:type properties.

RDFS & Inference (1)

- RDF Semantics
  - W3C Recommendation 10 February 2004
  - Specification of the six RDF specifications
  - Defines semantics and reasoning system, i.e. inference rules, for RDF and RDFS, by means of model theory
  - Describes rules for derivation of statements about subjects, property and values
  - Discusses the problem of inference w.r.t. literals
- RDF is an assertional language
  - (i.e. for allegations/statements about the world)
  - Especially if using the vocabulary defined with RDFS
  - Basis for description of further assertional languages
  - Meaning is not fixed
  - RDF Semantics describes the meaning of relationships in the context of RDF and RDFS, and provides rules to gain knowledge in specific situations (meaning) on certain elements.

RDFS & Inference (2)

- So: RDF + Vocabularies + RDF Semantic enables interpretation / inference
  - Since vocabularies can be added as desired, there can exist different interpretations of RDF data
  - Specifically, one always talks about RDF interpretations
- Sample rule:
  - IF (**<e>**<rdfs:subClassOf>**<m>**) and (**<k>**<rdf:type>**<e>**)
  - → addTriple (**<k>**<rdf:type>**<m>**)

Properties (1)

- Property is a concept connecting a subject to an object (see RDF Concepts and Abstract Syntax)
- rdf:Property
  - Property is also a resource identified by a URI
  - Property is bound to range and domains
  - Property can be split into subproperties (rdfs:subproperty)
- Statements about properties are possible – as well as properties of properties (URI of a URI)

Properties (2)

- Initial situation (<S><P><O>)
- Example:
  - (<P> <rdfs:range> <C>) means:
  - P is a property (in the range-statement, however, the Subject)
  - C is a class
  - → If P is used, O has to be an instance of C
- Confusing? You'll get used to it ;-)