

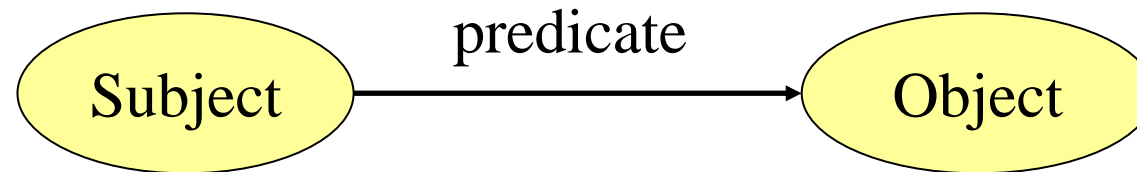
Semantic Web - RDF

Maciej Janik

Semantic Web
2009-07-01



- Resources (Subject, Object) connected by Predicates (relationships)



▪ Resources

- ♦ A resource is a referenced entity (Class, Individual, Relationship, ...)
- ♦ Resources *must* have
 - URIs – Uniform Resource Identifiers *or*
 - IRIs - Internationalized Resource Identifiers

- URI
 - ◆ <http://www.w3.org/Addressing/>
 - ◆ <http://www.ietf.org/rfc/rfc3987.txt>
- IRI (generalization of URI)
 - ◆ Normalized path: <http://a.b.c/d/../d> = <http://a.b.c/d>
 - ◆ Special characters: <http://häuser.und.bäume.de>
 - ◆ Right-to-left and left-to-right notation
 - Logical representation: <http://ab.CDE.FGH/ij/kl/mn/op.html>
 - Visual representation: <http://ab.HGF.EDC/ij/kl/mn/op.html>

- (re)Use already known URIs
 - ◆ Search engines: Swoogle, Okkam
- Have a document that URI points to
 - ◆ Good: <http://isweb.uni-koblenz.de/#groupISWeb>
 - ◆ Bad: <http://ThisSiteDoesNotExist/#groupISWeb>
- Use known standards/conventions for specific types of URIs:
 - ◆ Phone number, ISSN, etc.
- Do not use URLs as URIs for people or organizations.
<http://isweb.uni-koblenz.de> is just a website, not the ISWeb group identifier
 - ◆ Bad: <http://isweb.uni-koblenz.de> for ISWeb group
 - ◆ Better: <http://isweb.uni-koblenz.de/#groupISWeb>
- Derive new URIs from the websites (addresses) you can control :
 - ◆ Good: <http://isweb.uni-koblenz.de/#new> for me
 - ◆ Bad: <http://isweb.uni-koblenz.de/#new> for you

- **Resource**
 - ♦ Resource is a referenced entity (Class, Object, Entity, Relationship, ...)
 - ♦ Resource must have:
 - URIs – Uniform Resource Identifiers *or*
 - IRIs - Internationalized Resource Identifiers
- **Property (relationship)**
 - ♦ Similar to association in UML or relationship in database
 - ♦ Relationships between Resources and other Resources, or Resources to Literals
 - ♦ Property is also a Resource (have URI)
- **Literal**
 - ♦ Simple (atomic) data type (e.g String, int ...)
- **Statements**
 - ♦ “Resource has Property with Value”
 - ♦ Format: **Subject** –[Property]→ **Object**
 - ♦ Resources and/or literals are included in statement

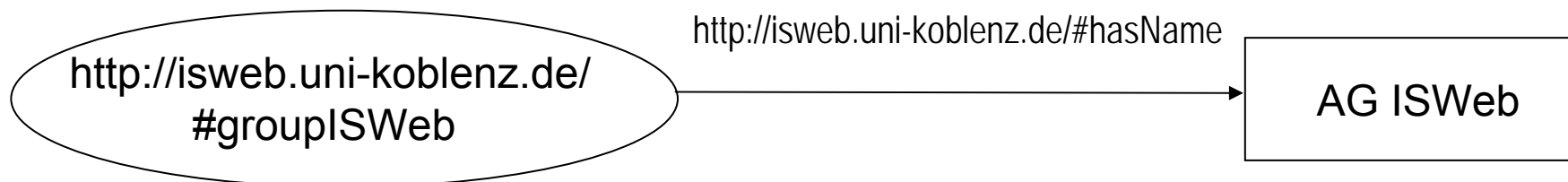
- **Statement**

- ♦ “Resource <http://isweb.uni-koblenz.de/#groupISWeb> has name AG ISWeb”

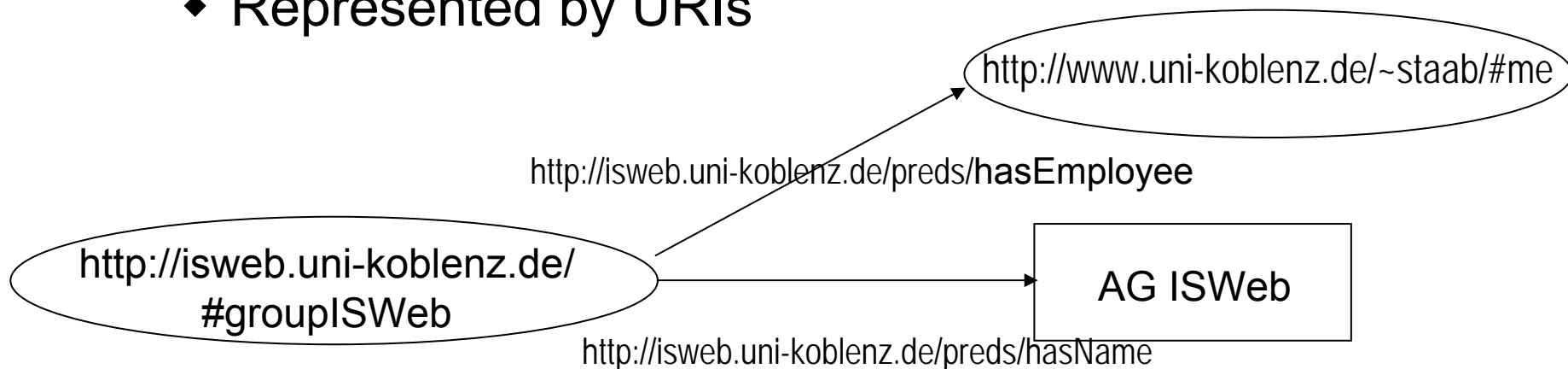
- **Structure**

- ♦ Resource (subject) <http://isweb.uni-koblenz.de/#groupISWeb>
- ♦ Property (predicate) <http://isweb.uni-koblenz.de/#hasName>
- ♦ Value (object) “AG ISWeb”
here: literal

- **Related Graph**



- **Nodes:**
 - ◆ Resources represented by URIs
 - ◆ Unnamed Resources (Blank Nodes)
 - ◆ Literals represented by Strings
- **Directed Edges:**
 - ◆ Represented by URIs



Example: Turtle notation

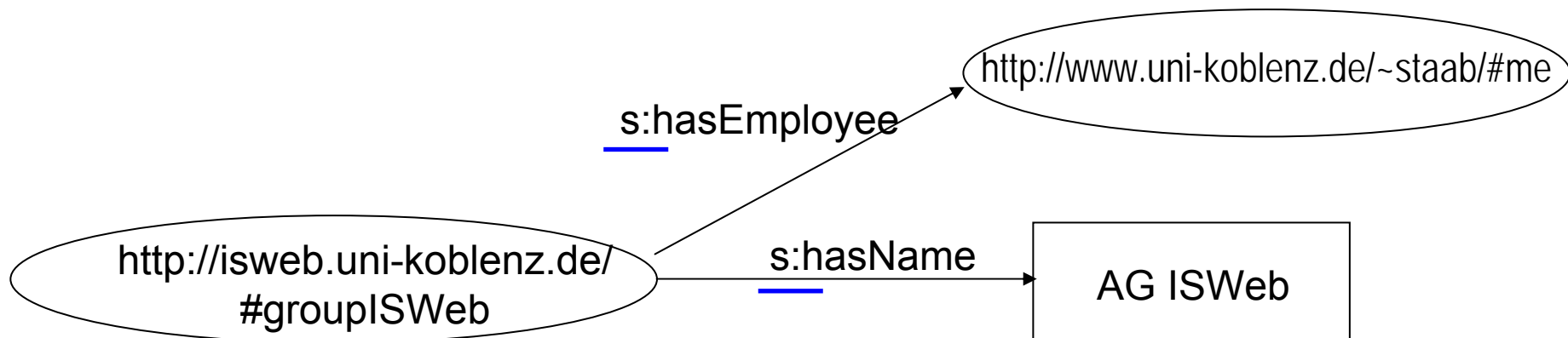
Turtle

```
<http://isweb.uni-koblenz.de/#groupISWeb>  
<http://isweb.uni-koblenz.de/preds/hasEmployee>  
<http://www.uni-koblenz.de/~staab/#me>
```

Turtle with Namespaces

```
@prefix s <http://isweb.uni-koblenz.de/preds/>
```

```
<http://isweb.uni-koblenz.de/#groupISWeb>  
s:hasEmployee <http://www.uni-koblenz.de/~staab/#me>
```



Example: Turtle notation (cont'd)

```
@prefix s <http://isweb.uni-koblenz.de/preds/>
<http://isweb.uni-koblenz.de/#groupISWeb> s:hasEmployee <http://www.uni-koblenz.de/~staab/#me> .
<http://isweb.uni-koblenz.de/#groupISWeb> s:hasEmployee <http://www.uni-koblenz.de/~sizov/#me> .
<http://isweb.uni-koblenz.de/#groupISWeb> s:hasEmployee <http://www.uni-koblenz.de/~janik/#me> .
<http://isweb.uni-koblenz.de/#groupISWeb> s:hasName "AG ISWeb"
```

Shorter version

```
@prefix s <http://isweb.uni-koblenz.de/preds>
@prefix u <http://www.uni-koblenz.de/>
<http://isweb.uni-koblenz.de/#groupISWeb> s:hasEmployee u:~staab/#me;
s:hasEmployee u:~sizov/#me;
s:hasEmployee u:janik/#me;
s:hasName "AG ISWeb".
```

Even shorter

```
@prefix s <http://isweb.uni-koblenz.de/preds>
@prefix u <http://www.uni-koblenz.de/>
<http://isweb.uni-koblenz.de/#groupISWeb>
s:hasEmployee u:~staab/#me, u:~sizov/#me, u:janik/#me;
s:hasName "ISWeb".
```

- RDF namespace

```
xmlns:rdf="http://www.w3.org/  
1999/02/22-rdf-syntax-ns#"
```

- rdf:XMLLiteral
- rdf:Property
- rdf:type

- rdf:Bag
- rdf:Seq
- rdf:Alt
- rdf:_1
- rdf:_2
- ...

- rdf:List
- rdf:first
- rdf:rest
- rdf:nil



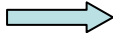
- rdf:Statement
- rdf:subject
- rdf:predicate
- rdf:object

- rdf:value

RDF Data types

s:thisLecture

title

- „Semantic Web“,  untyped
- „Semantic Web“@en,  untyped, but assigned „english“ (en) language
- „Semantic Web“^^xsd:string.  explicit type String

These are **three different** literals in the system.

Assigning type (e.g. String) cannot be combined with language

- Trend: all simple data types defined in XML
- Example

```
<xs:complexType name=„Name“>
```

```
  <xs:sequence>
```

```
    <xs:element name= „Title“           type=„String“, minOccurs=„0“/>
```

```
    <xs:element name= „FirstName“       type=„String“, minOccurs=„1“/>
```

```
    <xs:element name= „LastName“        minOccurs=„1“/>
```

```
  </xs:sequence>
```

```
  <xs:attribute name=„Birthday“ type="xs:date"/>
```

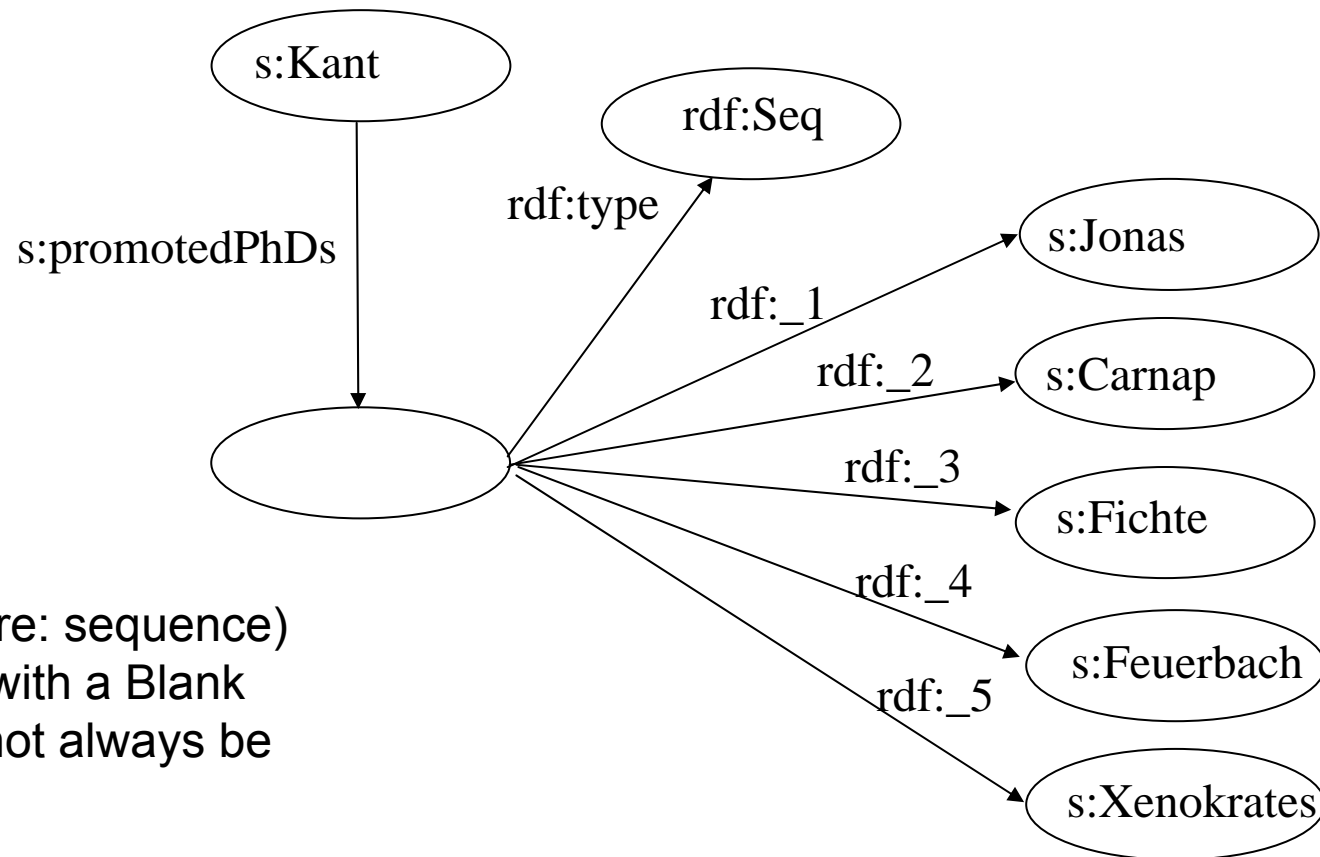
```
</xs:complexType>
```

Container

- Typed container
- Standard predicate names

List

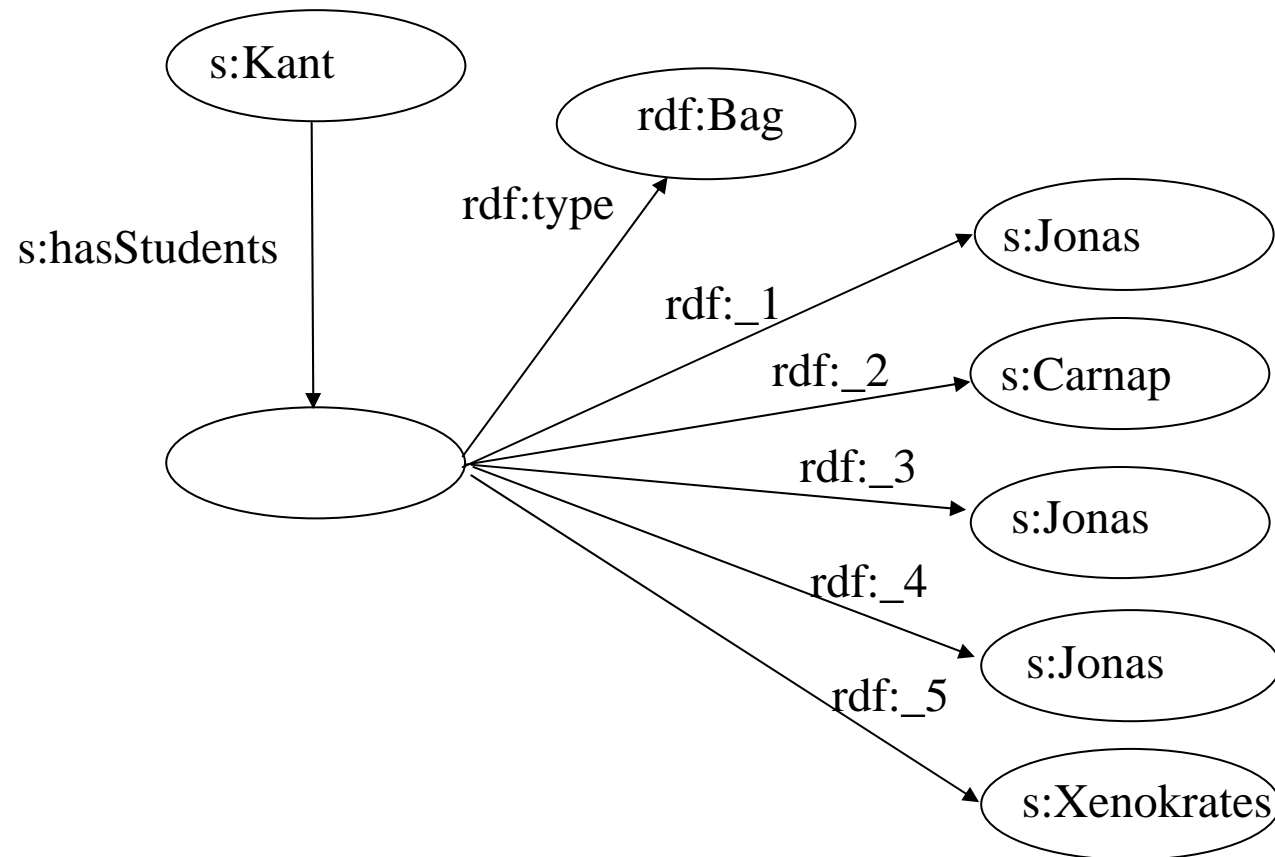
Number in the predicate is meaningful (used for indicating order); the same element can occur multiple times.



A container (here: sequence) is represented with a Blank Node – it may not always be necessary

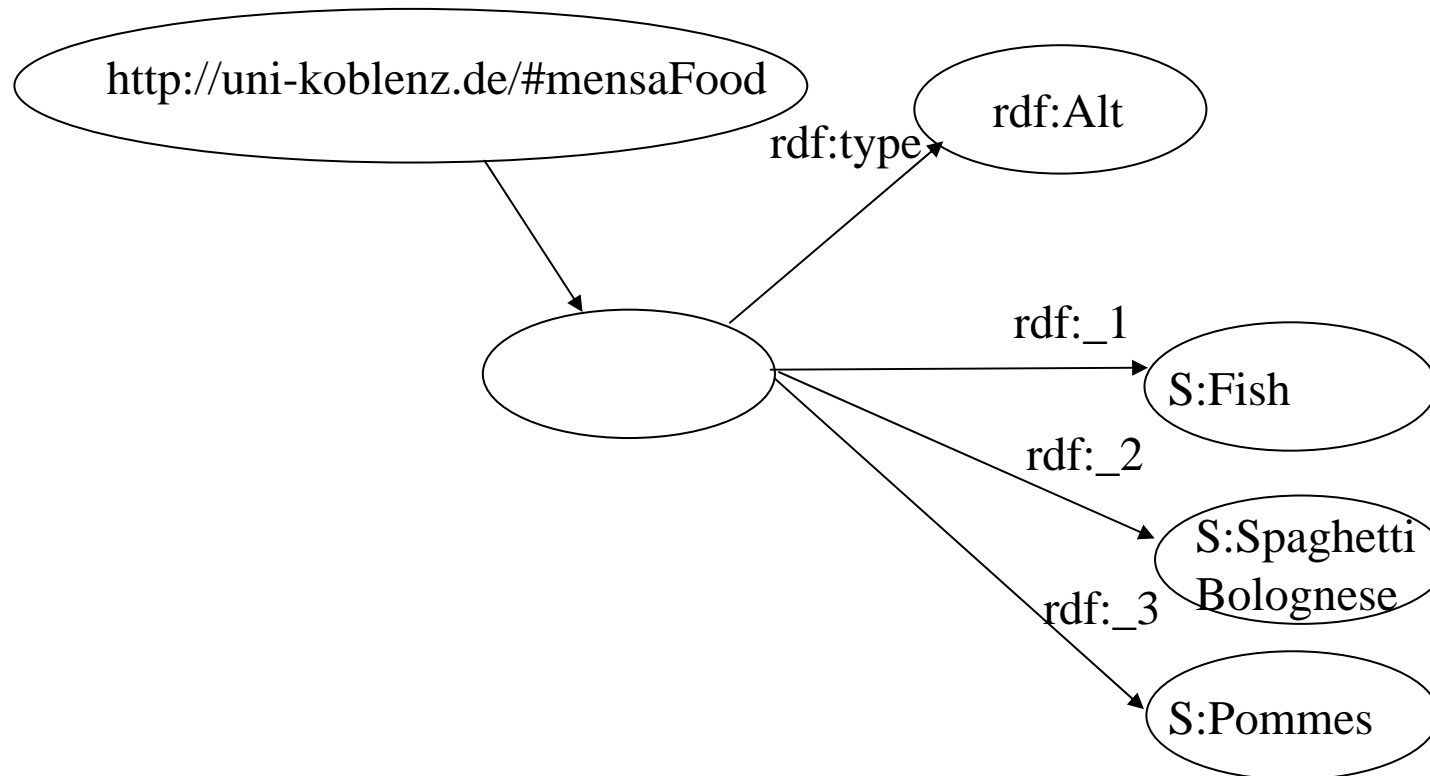
Multi-set

Numbers in predicates do not carry a specific meaning; this is a (multi) set of unordered resources; the same resource can appear more than once



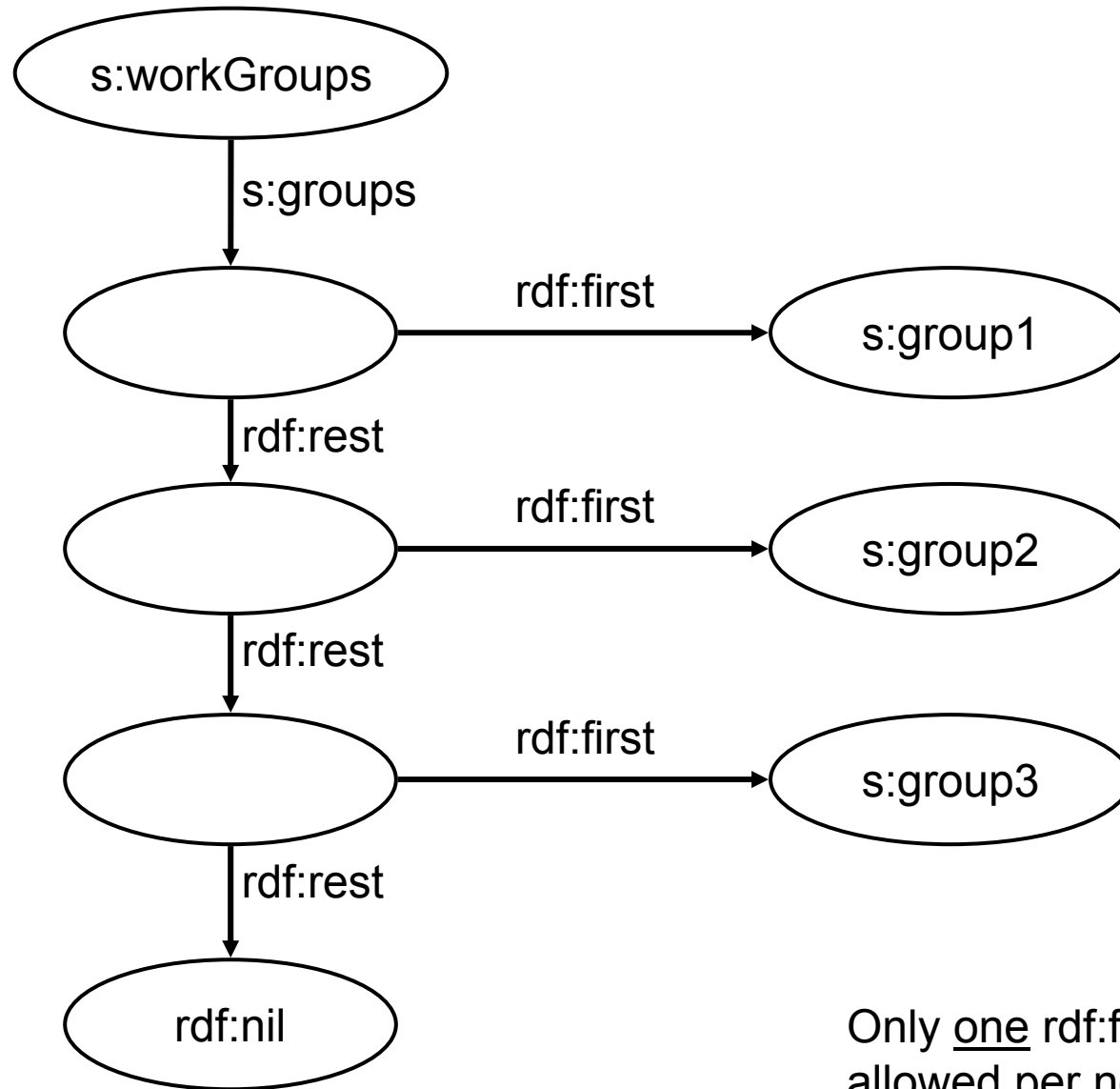
Example: Alternative

- Only one resource can be selected
- Numbers in predicates have no meaning



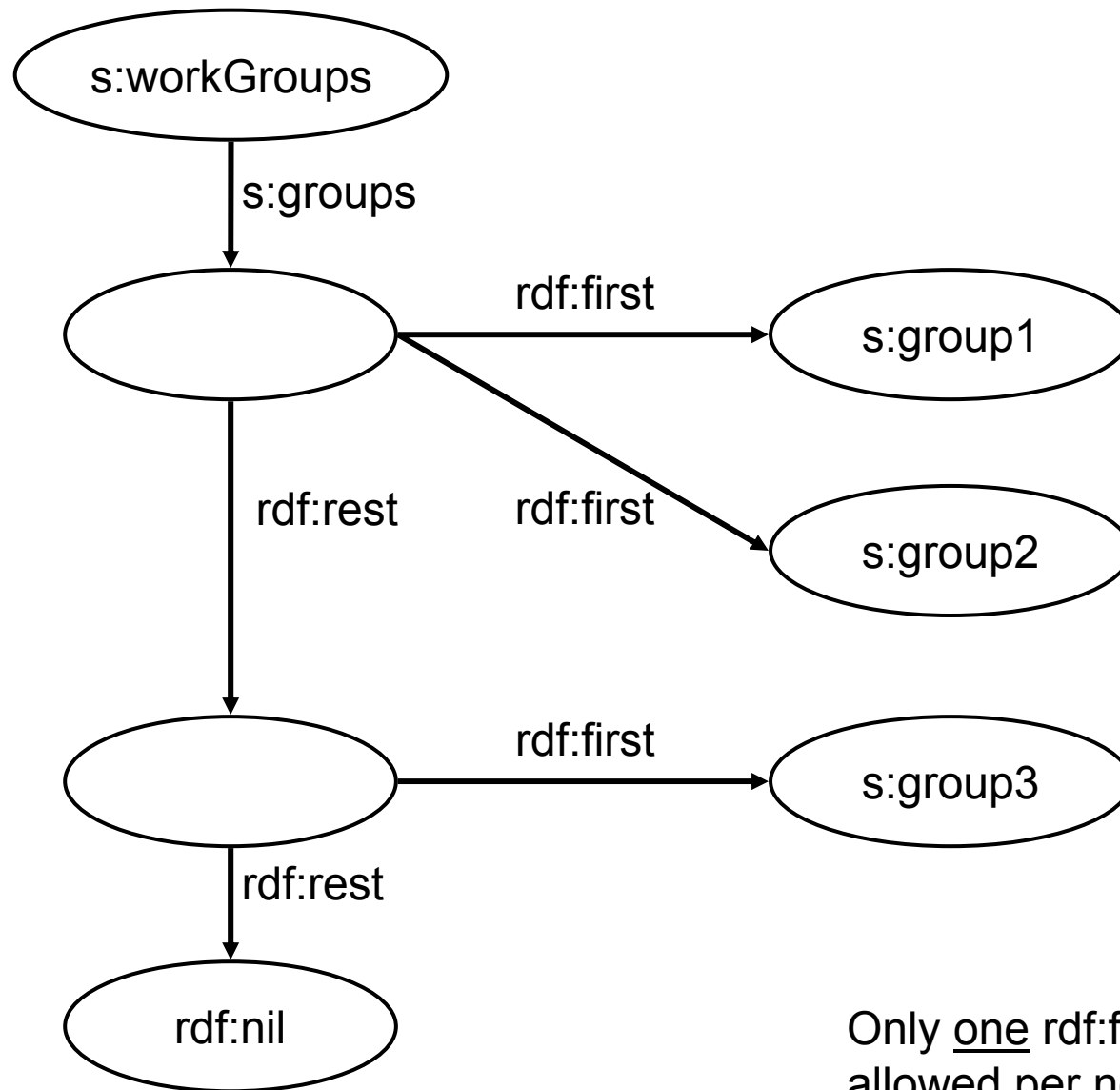
Collections

Way to represent multiple objects



Only one `rdf:first` and `rdf:rest` is allowed per node.

Linked list – what is the meaning of this?



Only one `rdf:first` and `rdf:rest` is allowed per node.

Reification

Statements about statements

How can I express following fact:

„Kant“ examined „Jonas“ in class „Introduction to CS“ and gave him grade „1.0“

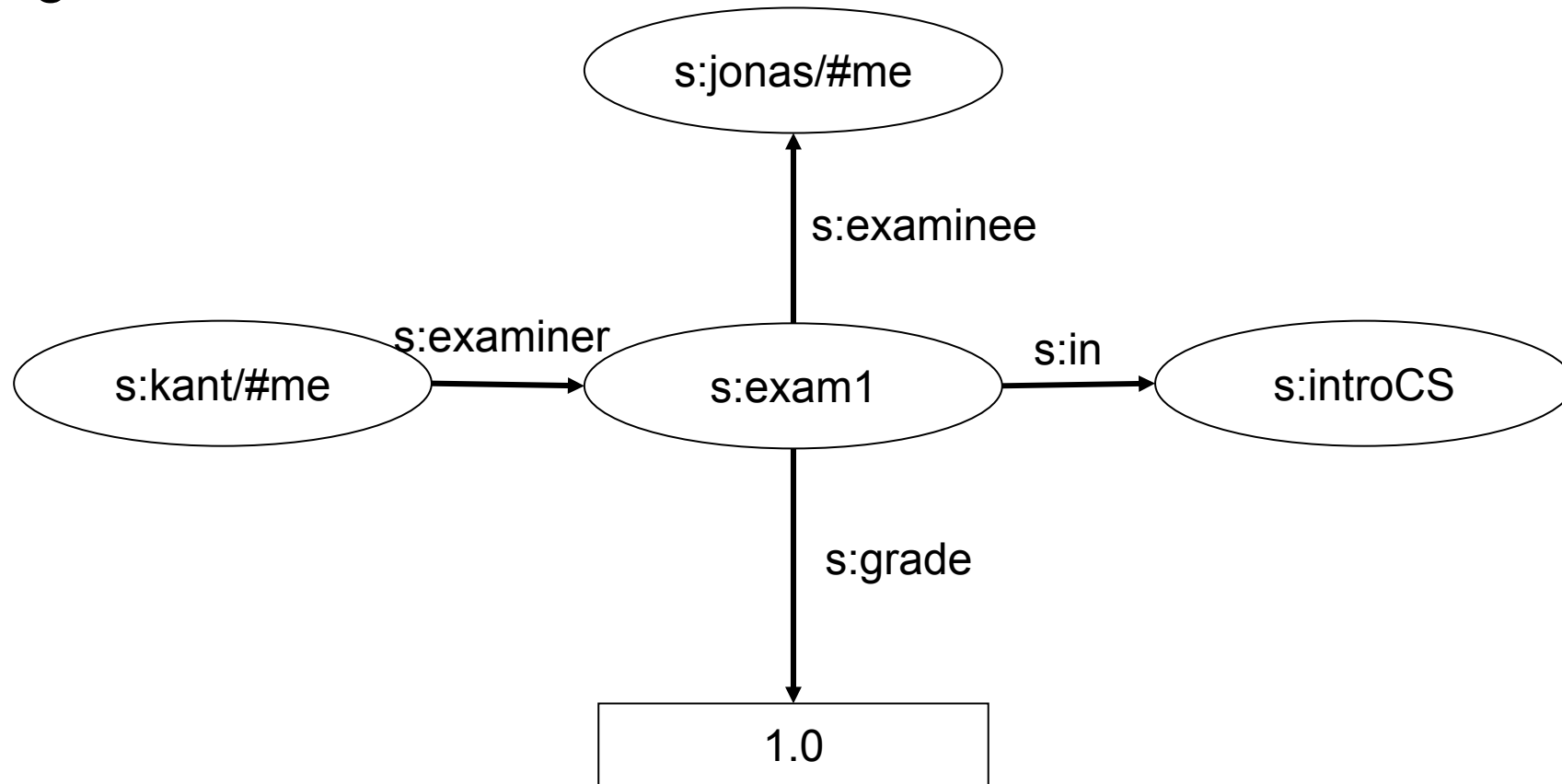
⇒ need for multi-relationship

Reification = refers to situation in natural language where statement is transformed so actions and events in it become quantifiable; here „Jonas exam“ becomes a described object

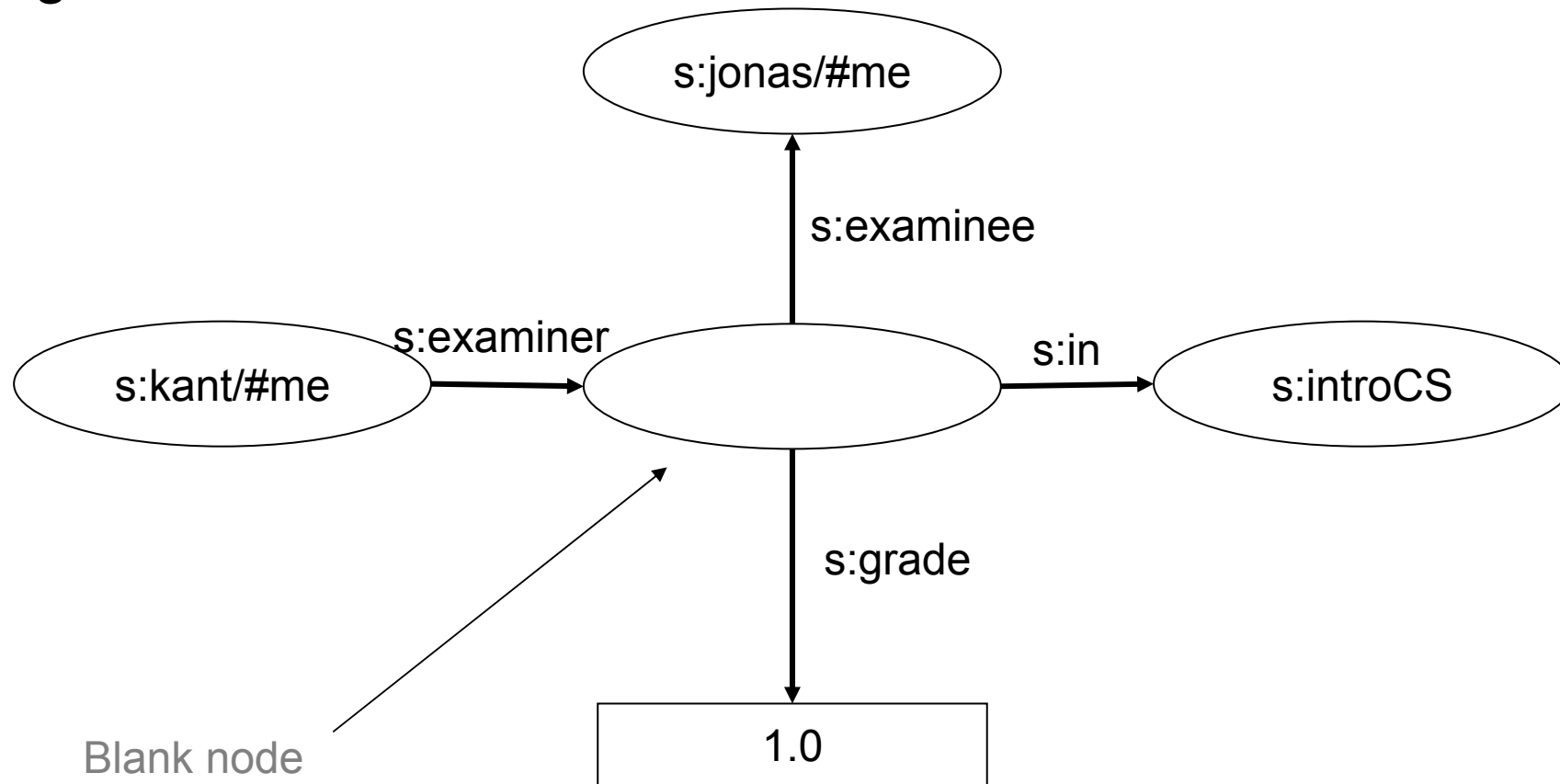
Forms of reification

- Ad hoc Reification
- RDF Reification
- Named Graphs
- Reification using other Design Patterns

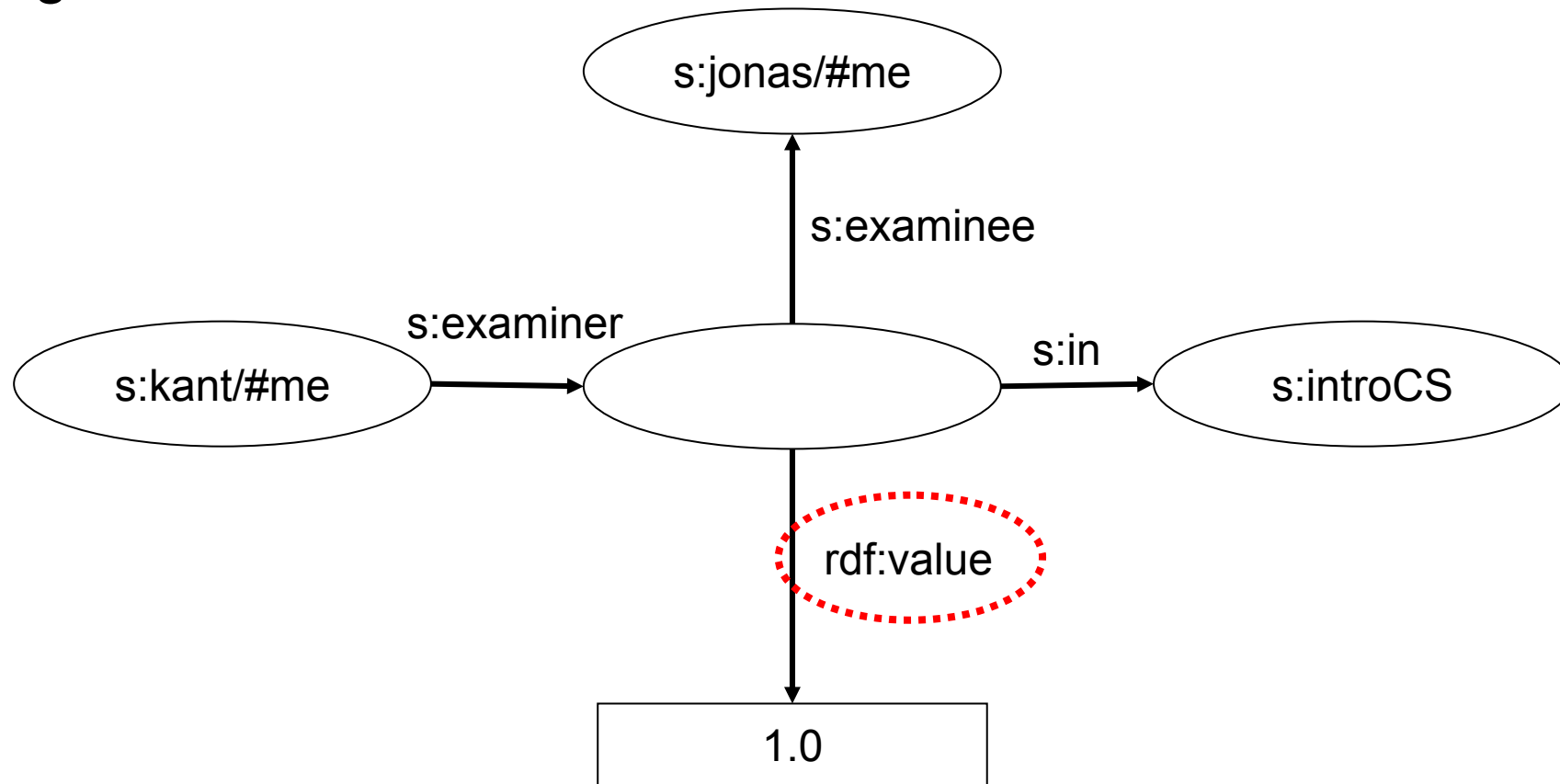
„Kant“ examined „Jonas“ in „Introduction to CS“ and gave him grade „1.0“



„Kant“ examined „Jonas“ in „Introduction to CS“ and gave him grade „1.0“

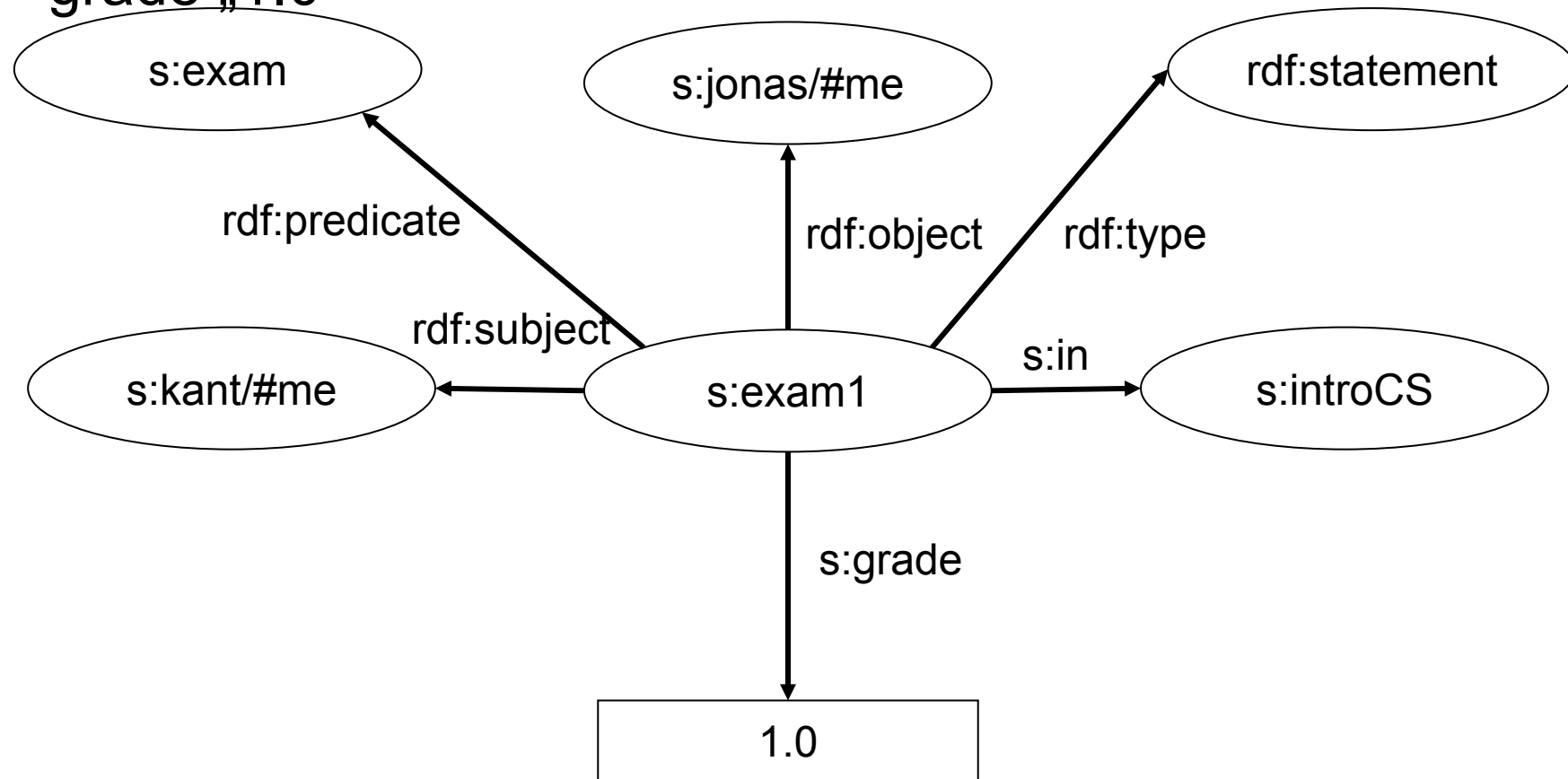


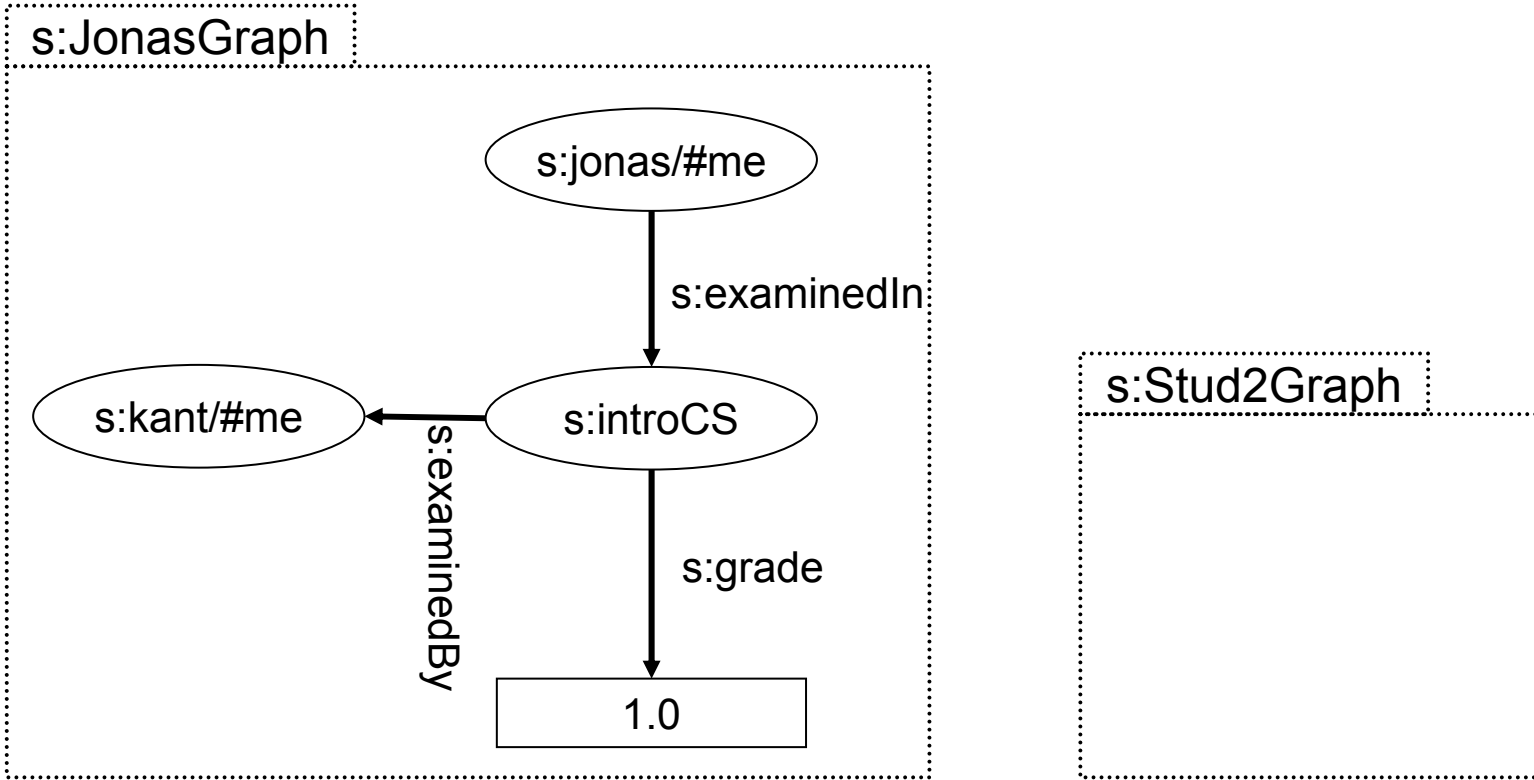
„Kant“ examined „Jonas“ in „Introduction to CS“ and gave him grade „1.0“



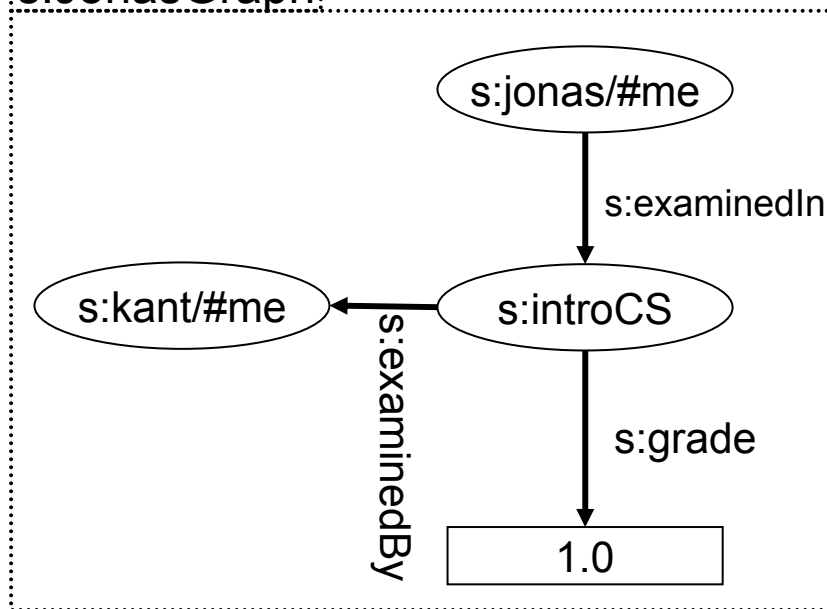
Best Practice

„Kant“ examined „Jonas“ in „Introduction to CS“ and gave him grade „1.0“





s:JonasGraph:



Based on Turtle notation

@prefix s <http://isweb.uni-koblenz.de/preds>

s:JonasGraph

{

s:jonas/#me s:examinedIn s:introCS.

s:introCS

s:grade "1.0" ;

s:examinedBy s:kan/#me .

}

- RDF is a standard syntax to represent (edge labeled) directed graphs in XML
- Uses resources with unique IRIs / URIs
- Describes named relationships between resources
- Has a limited vocabulary and semantics
- Supports
 - ◆ Collections (bag, alternative, set)
 - ◆ Lists
 - ◆ Reification (!)
- Good to describe ground facts, but not to describe simple model / schema → **RDFS (RDF Schema)**