

# Some Other Useful Features of RDF



# Acknowledgement

- This presentation is based on the excellent RDF primer by the W3C available at <http://www.w3.org/TR/rdf-primer/> This presentation is based on the excellent RDF primer by the W3C available at <http://www.w3.org/TR/rdf-primer/> and <http://www.w3.org/2007/02/turtle/primer/> .
- Much of the material in this presentation is verbatim from the above Web site.

# Presentation Outline

- Lists (containers and collections)
- Reification

# Lists in RDF

- Often we want to **relate a subject to a groups of things that play a similar role**: for example, to say that a book was created by several authors, or to list the students in a course, or the software modules in a package.
- RDF provides two kinds of lists for this:
  - **Containers (open lists)**
  - **Collections (closed lists)**

# Containers

- A **container** is a resource that contains things. The contained things are resources called **members**.
- RDF defines three types of containers:
  - **bags**
  - **sequences**
  - **alternatives**
- The following **classes** are also defined in RDF for these kinds of containers:
  - `rdf:Bag`
  - `rdf:Seq`
  - `rdf:Alt`

# Containers (cont'd)

- A **bag** represents a group of **resources or literals**, possibly including **duplicate** members, where there is no significance in the order of the members.
- A **sequence** represents a group of **resources or literals**, possibly including **duplicate** members, where the **order** of the members is significant.
- An **alternative** represents a group of **resources or literals** that are alternatives (typically for a single value of a property).
- `rdf:Bag`, `rdf:Seq` **and** `rdf:Alt` are **subclasses** of the RDFS class `rdfs:Container`.

# RDF Containers (cont'd)

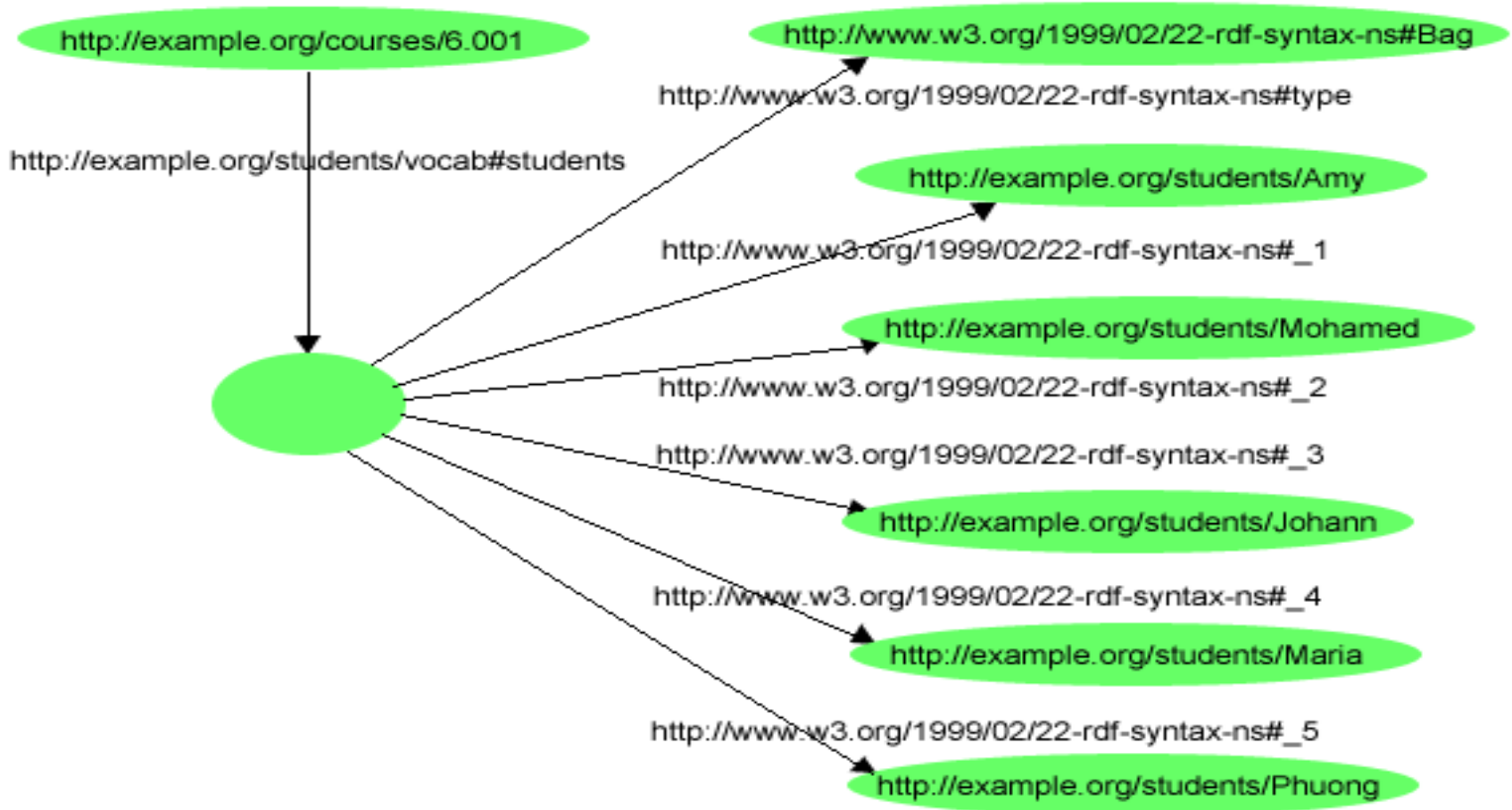
- To describe a resource as being one of these types of containers, the resource is given an `rdf:type` property whose value is one of the predefined resources `rdf:Bag`, `rdf:Seq`, or `rdf:Alt`.
- The **container resource** (which may either be a blank node or a resource with a `URIref`) denotes the group as a whole.
- The **members** of the container can be described by defining a **container membership property** for each member, with the container resource as its subject and the member as its object.
- There are **built-in container membership properties** with names of the form `rdf:_n`, where `n` is a decimal integer greater than zero, with no leading zeros, e.g., `rdf:_1`, `rdf:_2`, `rdf:_3`, and so on, and are used specifically for describing the members of containers.
- Container resources may also have **other properties** that describe the container, in addition to the container membership properties and the `rdf:type` property.

# Example of Bag

- Consider the sentence "Course 6.001 has the students Amy, Mohamed, Johann, Maria, and Phuong" .



# Example of Bag (cont'd)



# Example of Bag (cont'd)

- In Turtle notation:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix s: <http://example.org/students/vocab#>.
<http://example.org/courses/6.001>
  s:students [
    a rdf:Bag;
    rdf:_1 <http://example.org/students/Amy>;
    rdf:_2 <http://example.org/students/Mohamed>;
    rdf:_3 <http://example.org/students/Johann>;
    rdf:_4 <http://example.org/students/Maria>;
    rdf:_5 <http://example.org/students/Phuong>.
  ].
```

# Container Membership Properties

- The **class** `rdfs:ContainerMembershipProperty` has as instances all built-in container membership properties.
- The following triples are true for these:

```
rdfs:ContainerMembershipProperty
    rdfs:subClassOf rdf:Property .

rdf:_1 rdf:type rdfs:ContainerMembershipProperty .
rdf:_1 rdfs:domain rdfs:Resource .
rdf:_1 rdfs:range rdfs:Resource .
...
```

# Container Membership Properties (cont'd)

- One can define his own container membership properties, for example `ex:hasIngredient` by asserting:

```
ex:hasIngredient rdf:type  
                  rdfs:ContainerMembershipProperty
```

- **Example:**

```
ex:GreekSalad ex:hasIngredient ex:Tomato  
ex:GreekSalad ex:hasIngredient ex:Cucumber  
ex:GreekSalad ex:hasIngredient ex:Feta
```

# Container Membership Properties (cont'd)

- There is also the **property** `rdfs:member`

```
rdfs:member rdf:type rdf:Property .  
rdfs:member rdfs:domain rdfs:Resource .  
rdfs:member rdfs:range rdfs:Resource .
```

- All container membership properties are subproperties of `rdfs:member`

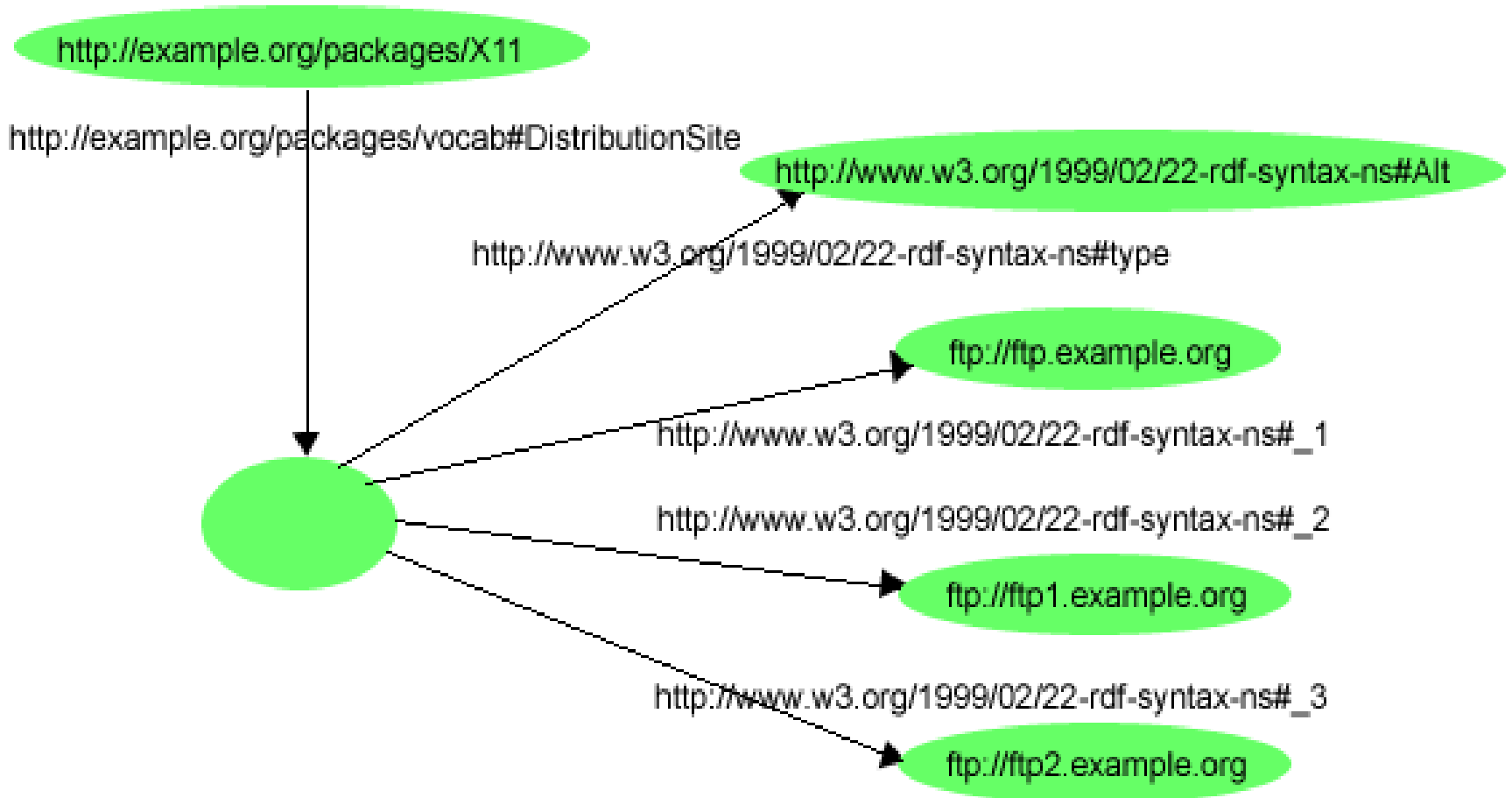
```
rdf:_1 rdfs:subPropertyOf rdfs:member  
...  
ex:hasIngredient rdfs:subPropertyOf rdfs:member
```

- `rdfs:member` is useful when we want to **query for all members of a container** (we cannot use `rdf:_1`, `rdf:_2`, ...)

# Example of Alternative

- Consider the sentence:
  - "The source code for X11 may be found at ftp.example.org, ftp1.example.org, or ftp2.example.org" .

# Example of Alternative (cont'd)



# Example of Alternative (cont'd)

- In Turtle notation:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
```

```
@prefix s: <http://example.org/packages/vocab#>.
```

```
<http://example.org/packages/X11>
```

```
  s:DistributionSite [
```

```
    a rdf:Alt;
```

```
    rdf:_1 <ftp://ftp.example.org>;
```

```
    rdf:_2 <ftp://ftp1.example.org>;
```

```
    rdf:_3 <ftp://ftp2.example.org>.
```

```
  ].
```



# Containers vs. simple triples

- Consider the statement:
  - Sue has written "Anthology of Time", "Zoological Reasoning", and "Gravitational Reflections".
- The above statement can be expressed in RDF using **three triples**:
  - `exstaff:Sue exterms:publication ex:AnthologyOfTime .`
  - `exstaff:Sue exterms:publication ex:ZoologicalReasoning .`
  - `exstaff:Sue exterms:publication ex:GravitationalReflections .`

# Containers vs. simple triples (cont'd)

- Alternatively, a bag can be used:

```
exstaff:Sue exterms:publication _:z .  
_:z rdf:type rdf:Bag .  
_:z rdf:_1 ex:AnthologyOfTime .  
_:z rdf:_2 ex:ZoologicalReasoning .  
_:z rdf:_3 ex:GravitationalReflections .
```

# Containers vs. simple triples (cont'd)

- However, there are cases where using a container is the most prominent modeling option.
- Consider the statement:
  - The resolution was approved by the Rules Committee, having members Fred, Wilma, and Dino.
- The statement says that the committee **as a whole** approved the resolution; it does not necessarily state that each committee member **individually** voted in favor of the resolution.

# Containers vs. simple triples (cont'd)

```
ex:resolution exterm:approvedBy ex:rulesCommittee .
```

```
ex:rulesCommittee rdf:type rdf:Bag .
```

```
ex:rulesCommittee rdf:_1 ex:Fred .
```

```
ex:rulesCommittee rdf:_2 ex:Wilma .
```

```
ex:rulesCommittee rdf:_3 ex:Dino .
```

# Containers Should be Used Carefully

- **Example:**

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix s: <http://example.org/packages/vocab#>.
<http://example.org/packages/X11>
  s:DistributionSite [
    a rdf:Alt;
    a rdf:Bag;
    rdf:_2 <ftp://ftp.example.org>;
    rdf:_2 <ftp://ftp1.example.org>;
    rdf:_5 <ftp://ftp2.example.org>
  ].
```

- The above example is a well-formed RDF description although the resource has been defined as an instance of both `rdf:Alt` and `rdf:Bag`. Also, the property `rdf:_2` has two values.
- **RDF does not enforce any “well-formedness constraint” for containers.** RDF applications that require containers to be "well-formed" should be written to check that the container vocabulary is being used appropriately, in order to be fully robust.

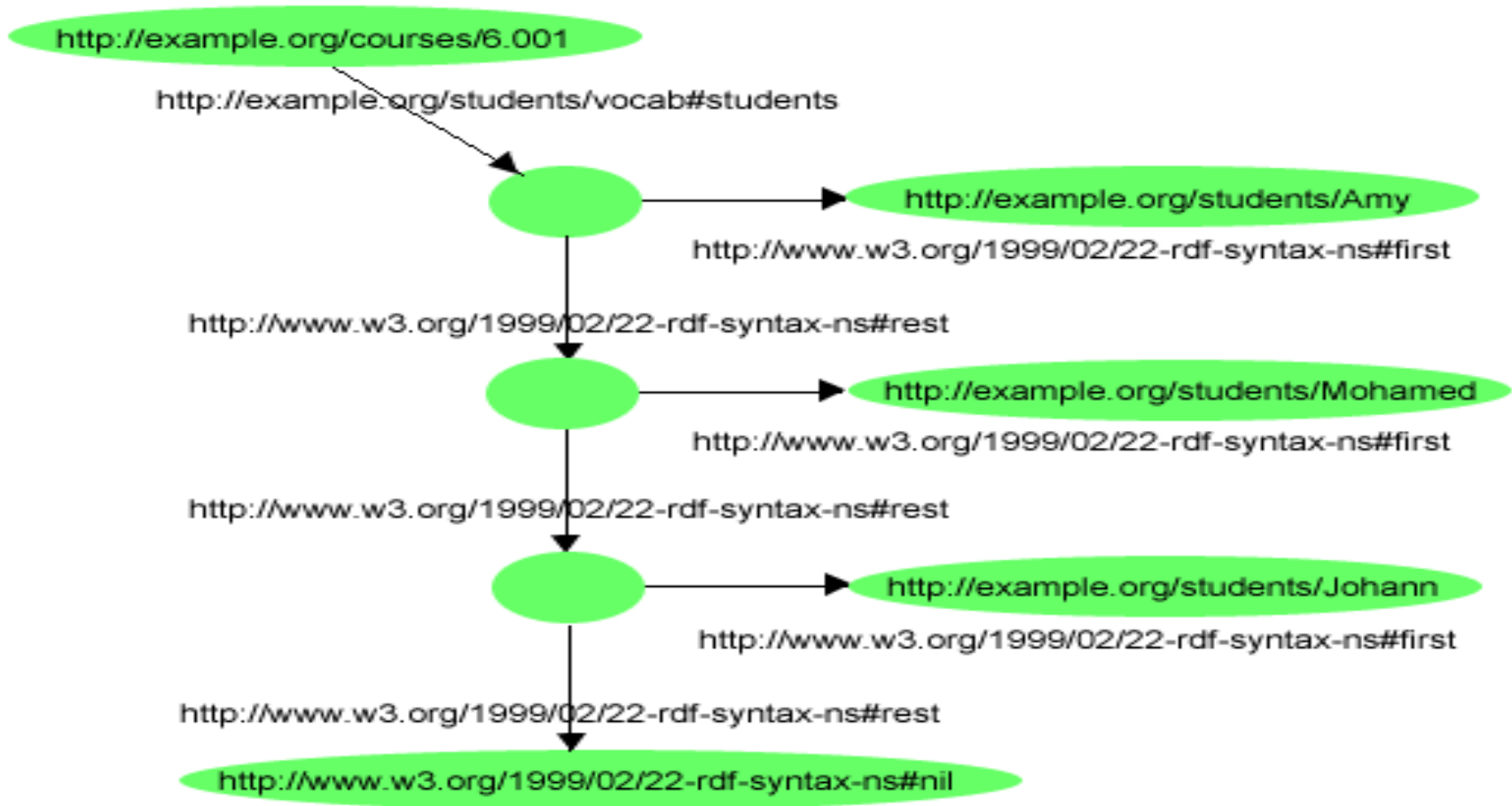
# RDF Collections

- A limitation of the containers is that there is no way to **close** them, i.e., to say "these are all the members of the container". A container only says that certain identified resources are members; it does not say that other members do not exist.
- RDF provides support for describing groups containing **only the specified members**, in the form of **RDF collections**.
- An **RDF collection** is a group of things represented as a **list structure** in the RDF graph. This list structure is constructed using a predefined **collection vocabulary** consisting of the predefined type `rdf:List`, the predefined properties `rdf:first` and `rdf:rest`, and the predefined resource `rdf:nil`.
- **RDF does not enforce well-formedness constraints for collections** too; this is left to applications.

# Example

- Consider the sentence “The students in course 6.001 are Amy, Mohamed and Johann” .

# Example as Graph





# Example in Turtle

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
```

```
@prefix s: <http://example.org/students/vocab#>.
```

```
<http://example.org/courses/6.001>
```

```
  s:students (
```

```
    <http://example.org/students/Amy>
```

```
    <http://example.org/students/Mohamed>
```

```
    <http://example.org/students/Johann>
```

```
  ).
```

# Presentation Outline

- Lists (containers and collections)
- **Reification**

# Reification

- RDF applications sometimes need to **describe other RDF statements** using RDF, for instance, to record information about when statements were made, who made them, or other similar information (this is sometimes referred to as “**provenance**” information).
- **Example:** The company `example.com` might want to record **who** made the statement

```
exproducts:item10245 exterms:weight "2.4"^^xsd:decimal .
```

# Reification (cont'd)

- RDF provides a **built-in vocabulary** intended for describing RDF statements. A description of a statement using this vocabulary is called a **reification** of the statement.
- The RDF reification vocabulary consists of the **class** `rdf:Statement`, and the properties `rdf:subject`, `rdf:predicate`, and `rdf:object`.

# Example (cont'd)

- `exproducts:item10245 exterms:weight "2.4"^^xsd:decimal .`
- Assign the statement about the tent's weight a URIref such as `exproducts:triple12345`.
- Now statements can be written describing the statement. For example:

```
exproducts:triple12345 rdf:type rdf:Statement .
exproducts:triple12345 rdf:subject exproducts:item10245 .
exproducts:triple12345 rdf:predicate exterms:weight .
exproducts:triple12345 rdf:object "2.4"^^xsd:decimal .
```
- Four statements like the above are usually called **“the reification quad”**.

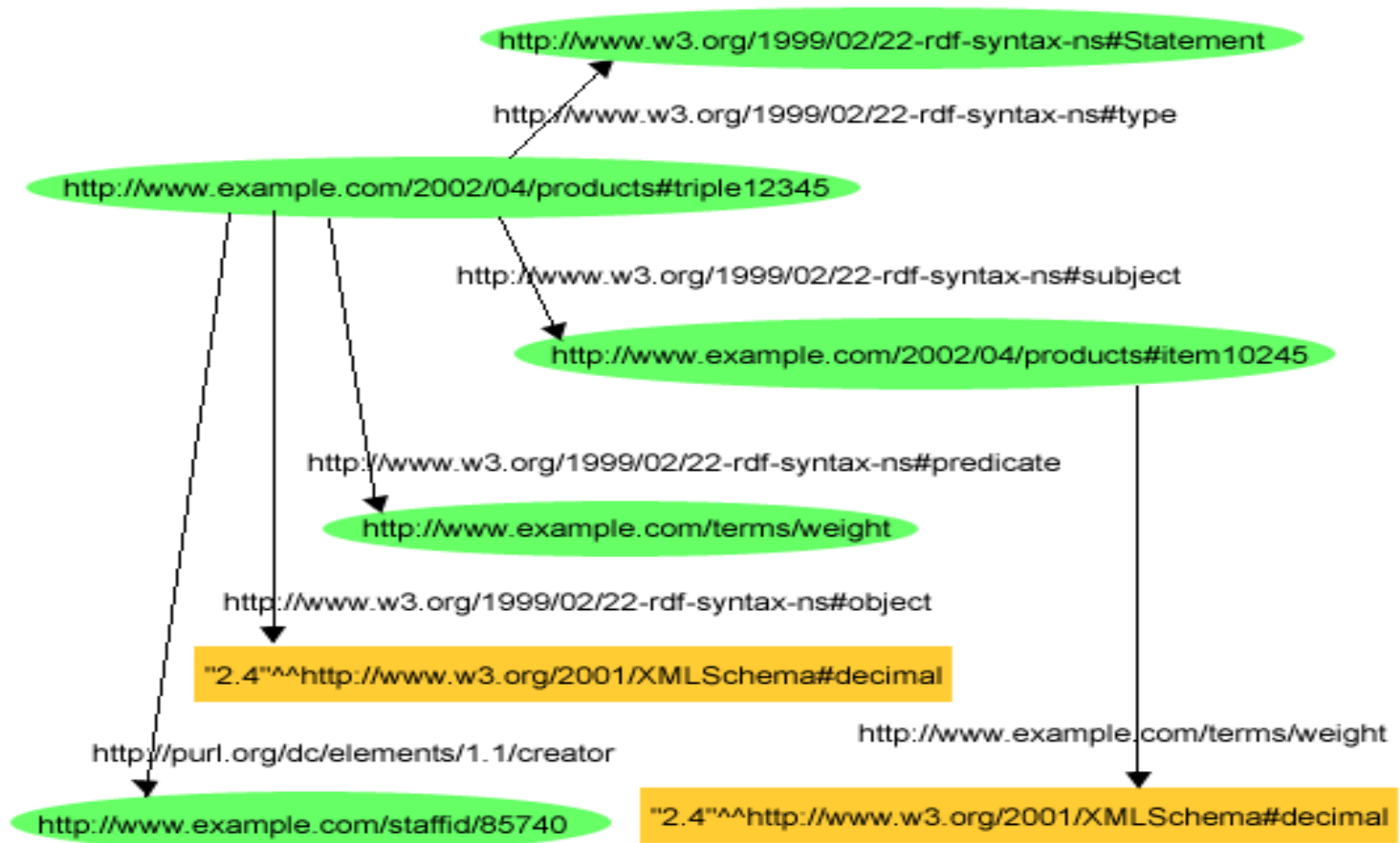
# Example (cont'd)

- Now we can add information about who made the statement:

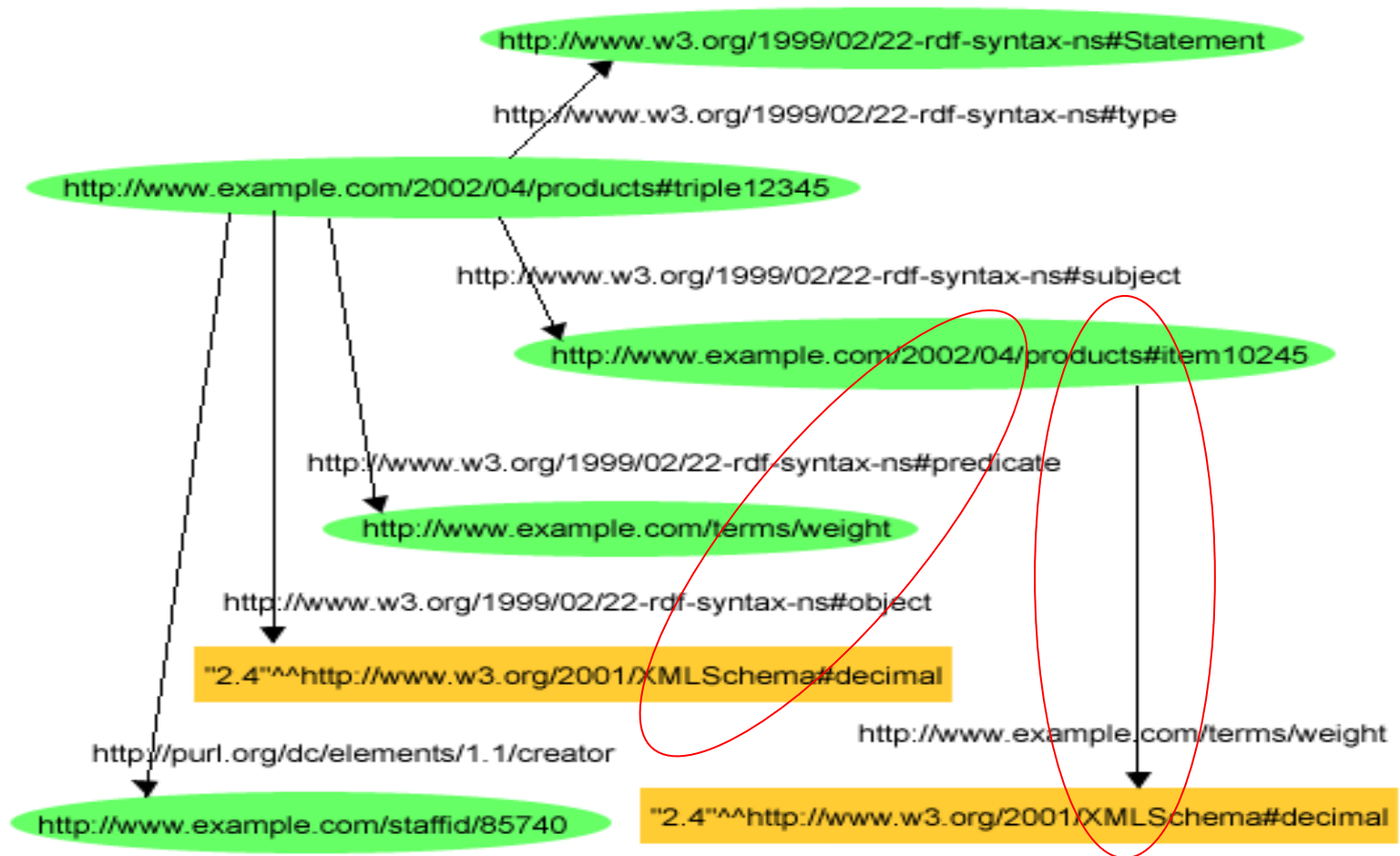
```
exproducts:triple12345 rdf:type rdf:Statement .
exproducts:triple12345 rdf:subject exproducts:item10245 .
exproducts:triple12345 rdf:predicate exterms:weight .
exproducts:triple12345 rdf:object "2.4"^^xsd:decimal .

exproducts:triple12345 dc:creator exstaff:85740 .
```

# The Statement, its Reification and its Attribution



# The Statement, its Reification and its Attribution





# Reification (cont'd)

- In the previous graph, nothing indicates that the original statement describing the tent's weight is the resource `exproducts:triple12345`, the resource that is the subject of the four reification statements and the statement that `exstaff:85740` created it.
- Asserting the reification is **not the same** as asserting the original statement, and neither implies the other:  
When someone says that John said something about the weight of a tent, they are not making a statement about the weight of a tent themselves, **they are making a statement about something John said**

# Reification (cont'd)

- The reification mechanism of RDF is **weak**.
- Applications are left to interpret and use the reification vocabulary in a proper manner.
- What RDF offers is **not enough to identify a particular triple** that is being reified and assert information about that triple (e.g., who wrote it, when etc.)
- SPARQL offers us the concept of **named graphs** that goes beyond reification: we are allowed to name an RDF graph using a URI and then assert information about this graph as a whole.

# Readings

- Chapter 2 of the book “Foundations of Semantic Web Technologies” or Chapters 2 and 3 of the Semantic Web Primer available from [http://www.csd.uoc.gr/~hy566/Chapter 2 of the book “Foundations of Semantic Web Technologies” or Chapters 2 and 3 of the Semantic Web Primer](http://www.csd.uoc.gr/~hy566/Chapter%20of%20the%20book%20%22Foundations%20of%20Semantic%20Web%20Technologies%22%20or%20Chapters%202%20and%203%20of%20the%20Semantic%20Web%20Primer) available from <http://www.csd.uoc.gr/~hy566/SWbook>Chapter 2 of the book “Foundations of Semantic Web Technologies” or Chapters 2 and 3 of the Semantic Web Primer available from [http://www.csd.uoc.gr/~hy566/SWbook\\_Chapter 2 of the book “Foundations of Semantic Web Technologies” or Chapters 2 and 3 of the Semantic Web Primer](http://www.csd.uoc.gr/~hy566/SWbook_Chapter%202%20of%20the%20book%20%22Foundations%20of%20Semantic%20Web%20Technologies%22%20or%20Chapters%202%20and%203%20of%20the%20Semantic%20Web%20Primer) available from <http://www.csd.uoc.gr/~hy566/SWbook.pdf> .
- The following material from the Semantic Web Activity Web page on RDF <http://www.w3.org/RDF/> :
  - RDF 1.1 Primer.
  - RDF 1.1: Concepts and Abstract Syntax
- Check out the linked data for the International Semantic Web Conference 2011 for some examples of using container membership functions: <http://data.semanticweb.org/conference/iswc/2011/complete>

# RDF-star

- Previously named RDF\*
- Extension of RDF 1.1
- More compact form of reification
- Treat a triple statement as a resource.

# Reification vs RDF-star

The creator of the triple statement:

```
exproducts:item10245 exterms:weight "2.4"^^xsd:decimal .  
is exstaff:85740 .
```

```
exproducts:triple12345 rdf:type rdf:Statement .  
exproducts:triple12345 rdf:subject exproducts:item10245 .  
exproducts:triple12345 rdf:predicate exterms:weight .  
exproducts:triple12345 rdf:object "2.4"^^xsd:decimal .  
  
exproducts:triple12345 dc:creator exstaff:85740 .
```

```
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>  
dc:creator exstaff:85740 .
```

- *5 triple statements vs 1 triple statement: reduced document size, hence:*
- *increased efficiency in data exchange.*

Reification

RDF Star

# Reification vs RDF-star

quoted triple

```
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>  
dc:creator exstaff:85740 .
```

RDF-star triple  
asserted triple

- Triples that include a triple as a subject or an object are known as [RDF-star triples](#).
- This triple statement **does not assert** that item10245 has weight 2.4
- It **only asserts** that exstaff:85740 has created this triple!
- The quoted triple can be either the subject or the object of an asserted triple
- An **RDF-star graph** is a set of [RDF-star triples](#).

# Reification vs RDF-star

```
@prefix exproducts : <http://example.org/products/vocab#>.
```

```
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>  
                                dc:creator exstaff:85740 .
```

- This dataset **does not assert** that item10245 has weight 2.4
- It **only asserts** that exstaff:85740 has created this triple!

# Reification vs RDF-star

```
@prefix exproducts : <http://example.org/products/vocab#>.


    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal.
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>
                                                dc:creator exstaff:85740 .
```

How many asserted triples do we have here?



# Reification vs RDF-star

Asserted &  
quoted triple




```
@prefix exproducts : <http://example.org/products/vocab#>.
    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal.
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>
    dc:creator exstaff:85740 .
```

- This dataset **asserts** that item10245 has weight 2.4
- **And asserts** that exstaff:85740 has created this triple

# Reification vs RDF-star

Asserted &  
quoted triple




```
@prefix exproducts : <http://example.org/products/vocab#>.
    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal.
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>
    dc:creator exstaff:85740 .
```

In a more compact way:

```
@prefix exproducts : <http://example.org/products/vocab#>.
    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal
    { | dc:creator exstaff:85740 } .
```

# Reification vs RDF-star

Asserted &  
quoted triple



```
@prefix exproducts : <http://example.org/products/vocab#>.
    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal.
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>
    dc:creator exstaff:85740 .
<<exproducts:item10245 exterms:weight "2.4"^^xsd:decimal>>
    dc:date "2021-07-07"^^xsd:date .
```

In a more compact way:

```
@prefix exproducts : <http://example.org/products/vocab#>.
    exproducts:item10245 exterms:weight "2.4"^^xsd:decimal
    { | dc:creator exstaff:85740; dc:date "2021-07-07"^^xsd:date } .
```

# Readings

- Reading on RDF star:
- [https://w3c.github.io/rdf-star/cg-spec/editors\\_draft.html#dfn-asserted](https://w3c.github.io/rdf-star/cg-spec/editors_draft.html#dfn-asserted)