Semantic Web

4. RDF

Gerd Gröner, Matthias Thimm
{groener,thimm}@uni-koblenz.de

Institute for Web Science and Technologies (WeST)
University of Koblenz-Landau

July 11, 2013
Recap

- XML as a modeling language
  - provides an easy (and standardized) means to represent (semi-)structured information
- XML syntax:
  - Elements
  - Attributes
  - Namespaces
- XML Schema and DTD
- Applications

**Home assignment:**
What is the XML Schema for the following DTD?

```xml
<!DOCTYPE a [
  <!ELEMENT a (b | c)+>
  <!ELEMENT b (a)>+
  <!ELEMENT c (#PCDATA)>
]>```
Some limitations of XML

- What is the relation between “Professor” and “Employee”
Some limitations of XML

- What is the relation between “Professor” and “Employee”
- Is “Professor” node in “Lecture” the same as “Professor” in “Seminar”, and does “Professor” has to be in the list of Professors?
Some limitations of XML

- What is the relation between “Professor” and “Employee”
- Is “Professor” node in “Lecture” the same as “Professor” in “Seminar”, and does “Professor” has to be in the list of Professors?
- How can I refer to “Kant” on the Web?
Some limitations of XML

- What is the relation between “Professor” and “Employee”
- Is “Professor” node in “Lecture” the same as “Professor” in “Seminar”, and does “Professor” has to be in the list of Professors?
- How can I refer to “Kant” on the Web?
- How do I link to external or internal data in/from the document?
Some limitations of XML

- What is the relation between “Professor” and “Employee”
- Is “Professor” node in “Lecture” the same as “Professor” in “Seminar”, and does “Professor” has to be in the list of Professors?
- How can I refer to “Kant” on the Web?
- How do I link to external or internal data in/from the document?
- Is it possible to order information and what meaning it has?
<bank>
  ...
</bank>

- a financial institution
- the shoreline of a lake
- a city in Iran
XML is a format for describing semi-structured information
XML is a format for describing semi-structured information
it has no requirements to
- use a specific vocabulary
- preserve the meaning of primitives
XML and semantics 2/2

- XML is a format for describing semi-structured information
- it has no requirements to
  - use a specific vocabulary
  - preserve the meaning of primitives
- needs standardization to be exchanged
XML and semantics 2/2

- XML is a format for describing semi-structured information
- it has no requirements to
  - use a specific vocabulary
  - preserve the meaning of primitives
- needs standardization to be exchanged
- only feasible for closed collaboration
  - people in a small community
  - web pages on a small intranet
Outline

1. The RDF model
2. Complex Statements
3. RDF Encodings
4. Summary and Exercises
The RDF model

Complex Statements

RDF Encodings

Summary and Exercises
RDF is a data model

- it represents semantic relationships in the form of a directed labelled graph
- the model is domain-neutral, application-neutral, and ready for internationalization
RDF is a data model
- it represents semantic relationships in the form of a directed labelled graph
- the model is domain-neutral, application-neutral, and ready for internationalization

The RDF data model is a conceptual layer that is, in general, independent of its representation using XML
- XML is a representation language for RDF
- RDF might be represented in other languages (Turtle, DB-specific syntax, . . .)
The vocabulary for RDF comprises

- a set $U$ of URIs,
- a set $L$ of literals,
- a set $B$ of blank nodes

Definition (RDF statement, RDF graph)

An RDF statement $S \in (U \cup B) \times U \times (U \cup L \cup B)$

An RDF graph $G$ is a finite set of RDF statements.
The vocabulary for RDF comprises

- a set $U$ of URIs,
- a set $L$ of literals,
- a set $B$ of blank nodes

Definition (RDF statement, RDF graph)

An *RDF statement* is a triple $S \in (U \cup B) \times U \times (U \cup L \cup B)$.

An *RDF graph* $G$ is a finite set of RDF statements.
The vocabulary for RDF comprises

- a set $U$ of URIs,
- a set $L$ of literals,
- a set $B$ of blank nodes

**Definition (RDF statement, RDF graph)**

An *RDF statement* is a triple $S \in (U \cup B) \times U \times (U \cup L \cup B)$. An *RDF graph* $G$ is a finite set of RDF statements.

For an RDF statement $S = (s, p, o)$ the element $s$ is called *subject*, $p$ is called *predicate*, and $o$ is called *object*. 
An RDF statement is a triple connecting entities (subject, object) with a predicate (relationship)
An RDF statement is a triple connecting entities (subject, object) with a predicate (relationship)

- Abstracts from simple sentences like *Bob knows Mary*.
- Smallest unit of information
The vocabulary of RDF comprises URIs, literals, and blank nodes. More specifically:

- A set $U$ of URIs:
  - also called *resources*
  - A URI uniquely identifies an entity (class, individual, relationship)
  - Everything is a resource: living and non-living objects, attributes, abstract concepts, ...
The vocabulary of RDF comprises URIs, literals, and blank nodes. More specifically:

- A set $U$ of URIs:
  - also called resources
  - A URI uniquely identifies an entity (class, individual, relationship)
  - Everything is a resource: living and non-living objects, attributes, abstract concepts, . . .

- A set $L$ of literals
  - strings that denote fixed resources
  - Used for names, labels, numbers, . . .
  - can only be used as objects
The vocabulary of RDF comprises URIs, literals, and blank nodes. More specifically:

- **A set $U$ of URIs:**
  - also called *resources*
  - A URI uniquely identifies an entity (class, individual, relationship)
  - Everything is a resource: living and non-living objects, attributes, abstract concepts, …

- **A set $L$ of literals**
  - strings that denote *fixed resources*
  - Used for names, labels, numbers, …
  - can only be used as objects

- **A set $B$ of blank nodes**
  - placeholder resources with anonymous label
  - used when the resource shall not be named
  - used for abstract concepts and for *reification* (later)
  - can be used for subjects and objects
(re)Use already known URIs:
- Search engines: Swoogle, Okkam

Use URLs as basis
- Good: http://west.uni-koblenz.de/#instituteWeST
- Bad: http://ThisSiteDoesNotExist/#instituteWeST

Use known standards/conventions for specific types of URIs:
- Phone number, ISSN, etc.

Do not use URLs as URIs for people or organizations
- Bad: http://west.uni-koblenz.de for WeST group
- Better: http://west.uni-koblenz.de/#groupWeST

Derive new URIs from the websites (addresses) you can control
- Good: http://west.uni-koblenz.de/#new for me
- Bad: http://west.uni-koblenz.de/#new for you
Example

▶ (http://example.org/#joe,
   http://xmlns.com/foaf/0.1/mbox,
   “joe.smith@example.org”)

▶ (http://example.org/#joe,
   http://xmlns.com/foaf/0.1/givenname,
   “John”)

▶ (http://example.org/#joe,
   http://xmlns.com/foaf/0.1/knows,
   http://example.org/#carl)
http://example.org/#carl
  ↗
http://xmlns.com/foaf/0.1/knows
  ↗
http://example.org/#joe
  ↓
http://xmlns.com/foaf/0.1/givenname
  "John"
  ↓
http://xmlns.com/foaf/0.1/mbox
  "joe.smith@example.org"
Outline

1 The RDF model
2 Complex Statements
3 RDF Encodings
4 Summary and Exercises
We use namespaces in the same way as for XML

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

Other standard namespaces:
- FOAF: `xmlns:foaf="http://www.xmlns.com/foaf/0.1"`
- For examples: `xmlns:ex="http://www.example.org"

Standard RDF vocabulary terms:
- `rdf:XMLLiteral`: XML literal values
- `rdf:Property`: class of properties
- `rdf:Statement`: class of RDF statements
- `rdf:Alt`, `rdf:Bag`, `rdf:Seq`: containers
- `rdf:List`: class of RDF Lists
- `rdf:nil`: the empty list
Standard RDF vocabulary terms (cont’d):

- **rdf:type**: type of an instance
- **rdf:first**: first item in a list
- **rdf:rest**: rest of a list
- **rdf:value**: for structured values
- **rdf:subject**: subject of a statement
- **rdf: predicate**: predicate of a statement
- **rdf:object**: object of a statement
Typed Literals

- \((ex:.thisLecture, ex:title, "Semantic Web")\) (untyped)

Other types:
- `xs:decimal`
- `xs:integer`
- `xs:float`
- `xs:boolean`
- `xs:date`
- `xs:time`
Typed Literals

- `(ex:thisLecture,ex:title, "Semantic Web")`
  (untyped)

- `(ex:thisLecture,ex:title, "Semantic Web"@en)`
  (untyped, but assigned “English” (en) language)
Typed Literals

- `(ex:thisLecture,ex:title, "Semantic Web")` (untyped)
- `(ex:thisLecture,ex:title, "Semantic Web"@en)` (untyped, but assigned “English” (en) language)
- `(ex:thisLecture,ex:title, "Semantic Web"^^xsd:string)` (explicit type string)
Typed Literals

- `(ex:thisLecture, ex:title, "Semantic Web")` (untyped)
- `(ex:thisLecture, ex:title, "Semantic Web"@en)` (untyped, but assigned “English” (en) language)
- `(ex:thisLecture, ex:title, "Semantic Web"^^xsd:string)` (explicit type string)

→ three different literals
Typed Literals

- \((\text{ex:thisLecture}, \text{ex:title}, "\text{Semantic Web}")\) (untyped)
- \((\text{ex:thisLecture}, \text{ex:title}, "\text{Semantic Web}@en\)\) (untyped, but assigned “English” (en) language)
- \((\text{ex:thisLecture}, \text{ex:title}, "\text{Semantic Web}^{\text{xsd:string}}\)\) (explicit type string)

→ three different literals

Other types:

- \(\text{xs:decimal}\)
- \(\text{xs:integer}\)
- \(\text{xs:float}\)
- \(\text{xs:boolean}\)
- \(\text{xs:date}\)
- \(\text{xs:time}\)
Containers 1/3

- `rdf:Seq` represent sequences using RDF statements
- order is important, elements may occur more than once
- uses blank nodes

```
(s:Kant, s:promotedPhDs, _b1)
(_, _b1, rdf:type, rdf:Seq)
(_, _b1, rdf:_1, s:Jonas)
...
```
- `rdf:Bag` represent bags (multi-sets) using RDF statements
- order is not important, elements may occur more than once
- uses blank nodes
- \texttt{rdf:Alt} represents a selection
- order is not important
- uses blank nodes

\texttt{http://uni-koblenz.de/#mensaFood}

\begin{itemize}
\item \texttt{rdf:Alt}
\item \texttt{S:Fish}
\item \texttt{S:Spaghetti Bolognese}
\item \texttt{S:Pommes}
\end{itemize}
(s:workGroups, s:groups, _b1)
(_b1, rdf:rest, _b2)
(_b2, rdf:first, s:group1)
(_b1, rdf:first, s:group2)
(_b2, rdf:first, s:group3)
...
Linked List 2/2

meaningless representation
Problem: How to represent the statement

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

Solution: Reification

Refers to situation in natural language where statement is transformed so actions and events in it become quantifiable

Types of reification:

Ad-hoc Reification

RDF Reification

Named Graphs

Reification using other Design Patterns
Problem: How to represent the statement

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

needs more arguments than a triple has

Solution: Reification

Refers to situation in natural language where statement is transformed so actions and events in it become quantifiable.

Types of reification:
- Ad-hoc Reification
- RDF Reification
- Named Graphs
- Reification using other Design Patterns
Problem: How to represent the statement

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

needs more arguments than a triple has

How to represent n-ary predicates?
Problem: How to represent the statement

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

needs more arguments than a triple has

How to represent n-ary predicates?

Solution: Reification
Problem: How to represent the statement

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

- needs more arguments than a triple has
- How to represent n-ary predicates?

Solution: Reification

- Refers to situation in natural language where statement is transformed so actions and events in it become quantifiable
Reification 1/4

Problem: How to represent the statement

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

needs more arguments than a triple has
How to represent n-ary predicates?

Solution: 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
Problem: How to represent the statement

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

needs more arguments than a triple has

How to represent n-ary predicates?

Solution: 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

Types of reification:

- Ad-hoc Reification
- RDF Reification
- Named Graphs
- Reification using other Design Patterns
“Kant” examined “Jonas” in class “Introduction to CS” and gave him grade “1.0”
“Kant” examined “Jonas” in class “Introduction to CS” and gave him grade “1.0”
“Kant” examined “Jonas” in class “Introduction to CS” and gave him grade “1.0”
Outline

1. The RDF model
2. Complex Statements
3. RDF Encodings
4. Summary and Exercises
RDF Encodings

- The RDF model just describes a labeled directed graph (set of triples)
The RDF model just describes a labeled directed graph (set of triples)

It does not prescribe the format how a set of triples is stored
The RDF model just describes a labeled directed graph (set of triples).

It does not prescribe the format how a set of triples is stored.

Possibilities:

- Graphical notation (informal)
- Triple-based notation (informal): (subject, predicate, object)
- Turtle (formal)
- RDF/XML (formal)
- N3 notation (formal)
Example of RDF in Turtle:

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

:mary :worksFor :ElJardinHaus.
:mary :name "Dalileh Jones"@en.
_:john :worksFor :ElJardinHaus.
_:john :idNumber "54321"^^xsd:integer.
<?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:ex="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)"
        ex:editor=""">
        <rdf:Description ex:fullName="Dave Beckett">
            <ex:homePage rdf:resource="http://purl.org/net/dajobe/" />
        </ rdf:Description>
    </ex:editor>
</rdf:Description>
</rdf:RDF>
Outline

1. The RDF model
2. Complex Statements
3. RDF Encodings
4. Summary and Exercises
Summary

- The RDF Model
  - URIs, literals, blank nodes
  - statements, graphs
- Complex statements
  - Typed literals
  - Containers: sequences, bags, alternatives
  - Linked lists
  - Reification
- RDF encodings: Turtle, RDF/XML
Pointers to further reading

▶ http://rdfabout.com (tutorials, guides, ...)
▶ RDF/XML Syntax Specification:
  http://www.w3.org/TR/REC-rdf-syntax/
Represent the following statement in RDF:

\[\text{On July 11th the temperature in Koblenz was 29}^\circ\text{C}\]
Exercises

▶ Represent the following statement in RDF:

On July 11th the temperature in Koblenz was 29°C

▶ Represent the following statement in RDF:

Matthias said “On July 11th the temperature in Koblenz was 29°C”
Exercises

- Represent the following statement in RDF:
  \[\text{On July 11th the temperature in Koblenz was 29°C}\]

- Represent the following statement in RDF:
  \[\text{Matthias said “On July 11th the temperature in Koblenz was 29°C”}\]

- Represent the following statements in RDF (Home assignment):
  
  Jack Smith studies at Koblenz University
  Koblenz University has the web site http://www.uni-koblenz.de
  Jack is a friend of Anna
  Anna studies at Boston University
  Boston University has the website http://www.bu.edu
  Anna has a dog called “Snoopy”