RDF Standard and Technologies

Tutorial for NETTAB 2007

2007-06-12

Heiko Stoermer, University of Trento, Italy
Primer's Primer

<html>
<head>
<title>NETTAB2007</title>
</head>
<body>
This year, <a href="http://...">NETTAB</a> provides an RDF Tutorial by <a href="http://...">Heiko Stoermer</a> from the <a href="http://...">University of Trento</a>.
</body>
</html>

Query:

"Which talks will NETTAB feature in 2007?"

Search-Engine answer: ???
Primer's Primer

NETTAB2007

hasYear 2007
hasURL http://...

HeikoStoermer
givesTalk TutorialRDF
hasURL http://...

TutorialRDF

isA Tutorial
location NETTAB2007

Query:

"Which talks will NETTAB feature in 2007?"

Possible RDF-Answer:

RDFTutorial, given by HeikoStoermer
Tutorial Overview - Theory

- Introduction to the Semantic Web Vision
- Introduction to RDF
  - What is RDF (not)
  - Main RDF Ingredients
  - Composing, creating, storing and viewing RDF
- Advanced RDF
  - Defining RDF Vocabularies
  - Querying RDF
- Discussion Pro/Con RDF
Tutorial Overview - Practice

- Creating a model with IsaViz
- PHP + RDF with RAP
- A word on Java
- Further resources and readings
  - general
  - developer tools
  - advanced topics
The Semantic Web I

- The Web today: Documents for humans.
- Problem: hard (impossible) to machine-process on a semantic level.
- Evidence: keyword-based search engines.
- Example: search for "red wine" does not return "Teroldego" 😞
The Semantic Web II

- Vision: Make the information in the Web machine-processable, for intelligent services, better user interaction and autonomous agents

- Examples:
  - search engines which know that Teroldego is a type of red wine 😊
  - automatic (re-) classification/ordering of documents
  - faceted navigation and browsing
  - applications that are able to combine remote services dynamically to achieve tasks
The Semantic Web III

- Realization idea: Semantic annotation of objects + query and reasoning mechanisms

- Requirement:
  - machine-processable languages for annotation and representation
  - reasoning tools
  - a naming mechanism

- Related areas: Logics, Knowledge Representation, Automated Reasoning

(very little/no Statistics)
Semantic Web IV

- Current approach: abstract representation of the world (classes, relations) + statements about real-world objects that conform to this abstract representation.

Core Language: RDF
Think Graphs!

- RDF is much about graphs and less about syntax
RDF Vision: Distribute, Integrate

- UNITN affiliation
- University type
- Italy location
- Heiko Stoermer givesTalk
- NETTAB2007 location
- RDFTutorial hasURL
- http://...
RDF Vision: Distribute, Integrate
What is RDF?

- An abstract formalism
- A graph data model (directed)
  - terms used: "graph" or "model"
- A set of binary statements ("triples")
  - Subject Predicate Object
- A representation of a part of the world
What is RDF not?

- A relational database
- A (database) management system
- A query language
- A file
- A new version of HTML or XML
- Something to say negative things with...
RDF Elements

- Resources \( R \)
- Properties \( P \)
- Literal Values \( L \)
- Assertions "\( R P L \)" or "\( R P R \)"
- Namespaces
(Almost) Everything is a Resource

- RDF stores statements about "resources":
  - Tangible things of the real world
  - Electronic objects
  - Abstract ideas such as classes/topics/...

- Resources are identified by URIs
  - URIs are rigid designators in a global domain.
Properties create Statements

- **Resource Property Resource**:  
  - NETTAB location Pisa
  - Heiko givesTalk RDFTutorial

- **Resource Property Literal**:  
  - Heiko fullName "Heiko Stoermer"
  - NETTAB date "2007"

- **Literal Property Resource**:  
  - "2007" dateOf NETTAB
Literal Values are Data

- Untyped literals are just strings
- Typed literals borrow from XML Schema Datatypes:
  - string
  - date
  - float
  - ...

Assertions span the Graph

- **Assertion = Triple = Statement**
- A graph can be **empty**
- A graph **cannot** contain only resources
- A set of assertions creates a graph
- A graph can be a **lettuce**: Heiko type Researcher
  Paolo type Professor
My Language is mine!

- RDF knows Namespaces
- Used to separate vocabularies (see RDFS later today)
- A namespace is defined by a URI
- There syntactic methods to define abbreviations for these URIs and a default namespace for a graph.
Composing RDF

- With a text editor (textual serialization in a file)
- With a graphical "drawing" tool
  - IsaViz Demo
- Programmatically (in-memory), see examples later today
RDF is XML

```xml
<?xml version="1.0"?>
<rdf:RDF

  xmlns:gss="http://www.w3.org/2001/11/IsaViz/graphstylesheets#"
  xmlns:nettab="http://www.nettab.org/tutorial-ns#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <rdf:Description rdf:ID="NETTAB2007">
    <nettab:date rdf:datatype="http://www.w3.org/2001/XMLSchema#date" rdf:resource="2007-06-12"/>
  </rdf:Description>

  <rdf:Description rdf:ID="hst">
    <nettab:name>Heiko Stoermer</nettab:name>
    <nettab:givesTalk>
      <rdf:Description rdf:about="http://www.know-who.net/talks/nettab.ppt">
        <nettab:name>RDF Tutorial</nettab:name>
        <nettab:location rdf:resource="#NETTAB2007"/>
      </rdf:Description>
    </nettab:givesTalk>
  </rdf:Description>
</rdf:RDF>
```
RDF is not XML

@prefix : <#> .

:NETTAB2007

:hst <http://www.nettab.org/tutorial-ns#givesTalk>
  <http://www.know-who.net/talks/nettab.ppt> ;
  <http://www.nettab.org/tutorial-ns#name> "Heiko Stoermer" .

<http://www.know-who.net/talks/nettab.ppt>
  <http://www.nettab.org/tutorial-ns#location> :NETTAB2007 ;
  <http://www.nettab.org/tutorial-ns#name> "RDF Tutorial" .
Storing RDF

- RDF graphs can be serialized as files (see example later) and stored in the file system
- For more DBMS-like applications, there are RDF repositories that provide
  - Query functionality
  - Access control
  - Distribution
- Example:
  - Sesame
  - 3-Store
  - JENA
  - RDF-API for PHP
Viewing RDF

- RDF Gravity
- IsaViz
- dot
- Jambalaya
- W3C RDF Validator
Advanced RDF'ing

- Schemas
- Query languages
No life without schemas...

- RDF Schema (RDFS) is a vocabulary to create vocabularies...
  - Comparable to XML Schema or XML DTD
  - Used to standardize which "tags" the creator of a graph is allowed to use for annotating resources
- Introduces notions such as "Class" and "Subclass"
- Helps define which relations a resource of a certain type may have
Main RDFS Namespace Elements

- X rdf:type rdfs:Class
  - denotes that resource X is a class
- R rdf:type rdf:Property
  - denotes that resource R is a property
- R rdfs:domain X
  - denotes that the subject of R must be an X
- R rdfs:range Y
  - denotes that the object of R must be a Y
RDFS 2

```xml
<?xml version="1.0"?>
<rdf:RDF
   xmlns:gss="http://www.w3.org/2001/11/IsaViz/graphstylesheets#"
   xmlns:nettab="http://www.nettab.org/tutorial-ns#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

   <rdfs:class rdf:ID="person" />
   <rdfs:class rdf:ID="talk" />
   <rdfs:class rdf:ID="event" />

   <rdf:Property rdf:ID="givesTalk">
      <rdfs:domain rdf:resource="#person" />
      <rdfs:range rdf:resource="#talk" />
   </rdf:Property>

   <rdf:Property rdf:ID="location">
      <rdfs:domain rdf:resource="#talk" />
      <rdfs:domain rdf:resource="#event" />
   </rdf:Property>

</rdf:RDF>
```
RDFS 3

- Compatibility check of a graph to a schema is NOT automatically performed upon parsing
- This is a consistency check which is performed by an RDFS reasoner on demand
- RDF triples that are inconsistent can be added to a graph (e.g. programmatically) and are not detected unless a consistency check is performed
- to answer queries which involve properties from a superclass, the query engine must have reasoning capabilities
- more details are left for the OWL tutorial later today
Querying RDF

- Several query languages exist to retrieve resulting triples from RDF
  - RDQL
  - SERQL
  - SPARQL (upcoming W3C Standard)
- These languages use **triple patterns** as input and return **matching triples** as results
- Example today: SPARQL
SPARQL Example

PREFIX nettab
  <http://www.nettab.org/tutorial-ns#>
SELECT ?x ?y ?z
WHERE { ?x nettab:givesTalk ?z }

Matching triple:

Subject: http://www.nettab.org/tutorial-ns#hst
Predicate: http://www.nettab.org/tutorial-ns#givesTalk
Object: http://www.know-who.net/talks/nettab.ppt
SPARQL Features

- Can deliver triples in serialized form
  - XML output
  - RDF graph
- Knows value filters (e.g. 'age >= 24')
- Knows "optionals" to return information in case it is available
- Optionals and filters can be combined
- Knows other constructs as from SQL (order, distinct, offset, limit...)
RDF Discussion

- Strengths and weaknesses
- Further developments
- Semantic Web shortcomings
- State of the Art
RDF Pros

- Potential universal data format with enhanced capabilities:
  - reasoning on subclass relations and properties
  - query results can be serialized easily (as opposed to SQL results)
  - RDF+OKKAM provides information integration for free
RDF Cons

- Limited Semantics
- Maturity
- Context
- Addressing
Limited Semantics

- Subclass relations are “built in“, i.e. directly understood by an RDF reasoner
- Other important relations have no semantics to a reasoner, their names are only symbols that are (hopefully!) meaningful to a human who writes a query, e.g.:
  - part-of
  - causal relations (cause -> effect)
- This is not RDF’s „fault“, it is inherent to the underlying KR mechanisms
Maturity: RDF is young... and old!

- RDF is only a few years old
- related technologies such as SPARQL are not even fully standardized yet
- repositories promote "successes" to store billions of triples; but how long does it take to answer reasoning queries?
- research (and funding) has mostly ended

⇒ transition phase between research and product development
⇒ too much has been invested already, RDF will probably not disappear.
Knowledge is Contextual

1. KR theory says: statements depend on situations, viewpoints, opinions, etc.
2. the Semantic Web envisions all RDF statements that exist as one big knowledge base

1) and 2) can be incompatible
Knowledge is Contextual II

Expression in RDF

„human“ consistency check OK

„Prodi prime_minister Gov_Italy“ + in 2006

„human“ consistency check OK

„Berlusconi prime_minister Gov_Italy“ + in 2004
Knowledge is Contextual III

- RDF knows only triples, not n-tuples
- The sentence "Prodi is Prime Minister of Italy in 2006" cannot be directly modelled.
  → Preliminary solution approaches exist, but are in research prototype state.
Addressing is Crucial

- Especially in Bioinformatics, RDF is seen as a future standard for information integration:
  - Integrating data from different sources
  - Integrating and clustering information around resources
  - Example: medical records of different hospitals for the same person
Your Resources are Lost

The „global graph“ vision of the Semantic Web has an identity and reference problem:

- whoever creates an RDF graph is free to create the identifiers for the described resources
- there is no mechanism to ensure that in your graph(s) and my graph(s) e.g. the NETTAB conference gets described using the same identifier
- so even if we described the same objects, we would never find out about it
The OKKAM Vision

- An architecture and infrastructure in development to address the identity and reference problem

- Strategy:
  - issuing globally unique identifiers for resources
  - enabling you to find my resources, so we can finally talk about the same objects and integrate our information correctly

- More information:
  
  www.okkam.org
Practical Part

- IsaViz demo
- PHP RDF API Quickstart
<?php
ob_start();
define("RDFAPIINCLUDE_DIR", "C:/Programme/LAMP/Apache/Apache2
/htdocs/rap095/rdfapi-php/api/");
include RDFAPIINCLUDE_DIR . "RdfAPI.php";

// Filename of an RDF document
$base="ex1_simple.xml.rdf"

// Create a new MemModel
$model = ModelFactory::getDefaultModel();

// Load and parse document
$model->load($base);

// Visualize model
$model->writeAsHtmlTable();
?>
PHP API: SPARQL Query

```php
// Load and parse document
$model->load($base);

// create querystring
$querystring = 'PREFIX nettab <http://www.nettab.org/tutorial-ns#>
SELECT ?x ?y
WHERE { ?x nettab:givesTalk ?y }';

// execute query and display resulting triples with HTML default renderer
echo $base->sparqlQuery($querystring, 'HTML');
?>
```
## PHP API: SPARQL Result

<table>
<thead>
<tr>
<th>?x</th>
<th>?y</th>
</tr>
</thead>
<tbody>
<tr>
<td>hst</td>
<td><a href="http://www.know-who.net/talks/nettab.ppt">http://www.know-who.net/talks/nettab.ppt</a></td>
</tr>
</tbody>
</table>
A word on Java

- **Major toolkit: JENA Toolkit**
  - jena.sourceforge.net
  - Production-strength
  - tested
  - large user base
- Usage more complex
- Visualization more complex
- Includes storage plugin architecture
- Includes reasoning and query answering support
- Includes support for OWL
Not covered in this talk...

- Blank nodes
- Reification
- RDF Collections
- Named Graphs in SPARQL
- and a lot more...
Resources - General

- W3C RDF page
  http://www.w3.org/RDF/

- Dave Beckett's Resource Description Framework (RDF) Resource Guide
  http://planetrrdf.com/guide/
Resources - Developer

- Developers Guide to Semantic Web Toolkits for different Programming Languages (Bizer & Westphal)
  http://www.wiwiss.fu-berlin.de/suhl/bizer/toolkits/

- Jena Semantic Web Framework:
  http://jena.sourceforge.net/

- RAP Toolkit for PHP
  http://www.wiwiss.fu-berlin.de/suhl/bizer/rdfapi
Resources - SPARQL

- http://www.w3.org/2004/Talks/17Dec-sparql/
- http://www.w3.org/TR/rdf-sparql-query/ (Working draft!)
Resources - Repositories

- Jena and RAP toolkits
- http://esw.w3.org/topic/LargeTripleStores
- Sesame: http://www.openrdf.org/
- 3store: http://threestore.sourceforge.net/
Further Reading

- RDF and **Topic Maps**: http://www.w3.org/TR/rdftm-survey/
- RDF and **Context**: http://okkam.dit.unitn.it:8088/RDFContextManager/publications
- **OKKAM** and **ISO11179** (added by request): http://www.okkam.org/ -> Wiki -> OkkamRelatedWork
Thank you!