Semantic Web Technologies
Resource Description Framework (RDF)

Heiko Paulheim
here be dragons...

Semantic Web Technologies (This lecture)

Technical Foundations

Berners-Lee (2009): Semantic Web and Linked Data
Overview

• A brief history of RDF
• Encodings of RDF
• Semantics and principles of RDF
• Embedding RDF in HTML – RDFa, Microdata, Microformats
• RDF Tools
• Examples for RDF in the wild
History: Metadata on the Web

• Goal: more effective rating and ranking of web contents, e.g., by search engines

• Who has created this page?
• When has it been changed the last time?
• What is its topic?
• Which is the content's license?
• How does it relate to other pages?
Metadata on the Web: Dublin Core

• Developed in 1995 at a workshop in Dublin, Ohio
• 15 predefined tags
• A widely accepted standard (ISO 15836:2009)

• May be embedded into HTML:

```html
<html>
<head profile="http://dublincore.org/documents/2008/08/04/dc-html/">
    <title>Semantic Web</title>
    <link rel="schema.DC" href="http://purl.org/dc/elements/1.1/" />
    <meta name="DC.publisher" content="University of Mannheim" />
    <meta name="DC.subject" content="Semantic Web" />
    <meta name="DC.creator" content="Heiko Paulheim" />
    <meta name="DC.relation" content="http://www.w3.org/2001/sw/" />
    ...
</head>
<body>
    ...
</body>
</html>
```
Metadata on the Web: Dublin Core

- Identifier
- Format
- Type
- Language
- Date
- Title
- Subject
- Coverage
- Description
- Creator
- Publisher
- Contributor
- Rights
- Source
- Relation
What is RDF?

• „Resource Description Framework“
• A W3C standard since 2004
• Description of arbitrary things

• View 1: Sentences in the form <subject, predicate, object>
  „Heiko works for University of Mannheim.“

• View 2: Directed graphs with labeled edges

![Diagram of RDF with nodes labeled Heiko and Uni Mannheim connected by an edge labeled "works for".]
Basic Building Blocks of RDF

- **Resources**
  - denote things
  - are identified by a URI
  - can have one or multiple types

- **Literals**
  - are values like strings or integers
  - can only be objects, not subjects or predicates (graph view: they can only have ingoing edges)
  - can have a datatype or a language tag (but not both)

- **Properties (Predicates)**
  - Link resources to other resources and to literals
Types

- All resources (not literals) can have a type
- Types can be arbitrarily defined
- The predefined predicate `rdf:type` defines the type of a resource

- Semantic Web is a lecture

* <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
Resource vs. Literal

• A literal is an atomic value
  – can only be object
  – i.e., a literal terminates always a graph

• A resource can be a subject itself

http://www.dws.informatik.uni-mannheim.de/sw

dc:creator

"Heiko Paulheim"

http://www.dws.informatik.uni-mannheim.de/sw

dc:creator

http://www.dws.informatik.uni-mannheim.de/heiko
Datatypes for Literals

- (Almost) all XML Schema datatypes may be used

- Exception:
  - XML specific types
  - The underspecified type "duration"
  - sequence types
Language Tags for Literals

• Literals may be defined in different natural languages
  – "München"@de
  – "Munich"@en

• Those can be marked

• Note: the Semantic Web is multilingual!

• Language codes according to ISO 963
  – if both are defined, ISO 963-1 has to be used!

# Language Tags for Literals

<table>
<thead>
<tr>
<th>ISO 639-2 Code</th>
<th>ISO 639-1 Code</th>
<th>English name of Language</th>
<th>French name of Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>aar</td>
<td>aa</td>
<td>Afar</td>
<td>afar</td>
</tr>
<tr>
<td>abk</td>
<td>ab</td>
<td>Abkhazian</td>
<td>abkhaze</td>
</tr>
<tr>
<td>aca</td>
<td>aca</td>
<td>Achiene</td>
<td>acah</td>
</tr>
<tr>
<td>ach</td>
<td>ach</td>
<td>Acoli</td>
<td>acoli</td>
</tr>
<tr>
<td>ada</td>
<td>ada</td>
<td>Adangme</td>
<td>adangme</td>
</tr>
<tr>
<td>ady</td>
<td>ady</td>
<td>Adyge; Adygei</td>
<td>adyghé</td>
</tr>
<tr>
<td>afl</td>
<td>afl</td>
<td>Afro-Asiatic languages</td>
<td>afro-asiatiques, langues</td>
</tr>
<tr>
<td>afh</td>
<td>afh</td>
<td>Afrihili</td>
<td>afrihili</td>
</tr>
<tr>
<td>afr</td>
<td>afr</td>
<td>Afrikaans</td>
<td>afrikaans</td>
</tr>
<tr>
<td>ain</td>
<td>ain</td>
<td>Aini</td>
<td>ainou</td>
</tr>
<tr>
<td>aka</td>
<td>aka</td>
<td>Akan</td>
<td>akon</td>
</tr>
<tr>
<td>akk</td>
<td>akk</td>
<td>AkaKadian</td>
<td>akkadian</td>
</tr>
<tr>
<td>alb (B)</td>
<td>sq</td>
<td>Albanian</td>
<td>albanais</td>
</tr>
<tr>
<td>sq (T)</td>
<td>sq</td>
<td>Alsatian</td>
<td>alsatien</td>
</tr>
<tr>
<td>ale</td>
<td>ale</td>
<td>Aleut</td>
<td>aleut</td>
</tr>
<tr>
<td>alg</td>
<td>alg</td>
<td>Algonquian languages</td>
<td>algonquines, langues</td>
</tr>
<tr>
<td>alt</td>
<td>alt</td>
<td>Southern Alta</td>
<td>alta du Sud</td>
</tr>
<tr>
<td>amh</td>
<td>amh</td>
<td>Amharic</td>
<td>amharique</td>
</tr>
<tr>
<td>ang</td>
<td>ang</td>
<td>English, Old (ca.450-1100)</td>
<td>anglo-saxon (ca.450-1100)</td>
</tr>
<tr>
<td>anp</td>
<td>anp</td>
<td>Angika</td>
<td>angika</td>
</tr>
<tr>
<td>apa</td>
<td>apa</td>
<td>Apache languages</td>
<td>apache, langues</td>
</tr>
</tbody>
</table>
Datatypes in RDF

• Examples:
  • :Munich :hasName "München"@de .
  • :Munich :hasName "Munich"@en .
  • :Munich :hasPopulation "1356594 "^^xsd:integer .
  • :Munich :hasFoundingYear "1158-01-01"^^xsd:date .

• Note: there are no default datatypes (not even “string”!)
• These are three different literals:
  – "München"
  – "München"@de
  – "München"^^xsd:string .
Triple Notation

- Triples consist of a subject, predicate, and object
- An RDF document is an *unordered* set of triples

**Simple triple:**

```xml
<http://www.dws.informatik.uni-mannheim.de/teaching/semantic-web>
<http://purl.org/dc/elements/1.1/relation>
<http://www.w3.org/2001/sw/> .
```

**Literal with language tag:**

```xml
<http://www.dws.informatik.uni-mannheim.de/teaching/semantic-web>
<http://purl.org/dc/elements/1.1/subject>
"Semantic Web"@en .
```

**Type literal:**

```xml
<http://www.dws.informatik.uni-mannheim.de/teaching/semantic-web>
<http://www.uni-mannheim.de/mhb/creditpoints>
"6"^^<http://www.w3.org/2001/XMLSchema#integer> .
```
Turtle Notation

• A simplified triple notation

• Central definition of namespaces (and a default namespace):

```turtle
@prefix dc: <http://purl.org/dc/elements/1.1/>
@prefix : <http://www.dws.informatik.uni-mannheim.de/teaching/>
```

• Triples sharing the same subject or subject+predicate:

```turtle
:semantic-web dc:subject "Semantisches Web"@de ,
    "Semantic Web"@en ;
    dc:creator "Heiko Paulheim".
```

• Shorthand notation for `rdf:type`:

```turtle
:semantic-web a :lecture .
```
Notation RDF/XML

- A W3C standard since 2004
- Encodes RDF in XML
- Suitable for machine processing (plenty of XML tools!)

- Defining resources:
  
  ```xml
  <rdf:Description rdf:about="http://dws.informatik.uni-mannheim.de/teaching/sw">
    <dc:creator>Heiko Paulheim</dc:creator>
  </rdf:Description>
  ```

- Defining typed resources:
  
  ```xml
  <rdf:Description rdf:about="http://dws.informatik.uni-mannheim.de/teaching/sw">
    <rdf:type rdf:resource="http://www.uni-mannheim.de/mhb/Lecture"/>
  </rdf:Description>
  ```

- Alternative representation:
  
  ```xml
  <mhb:Lecture rdf:about="http://dws.informatik.uni-mannheim.de/teaching/sw"
  xmlns:mhb="http://www.uni-mannheim.de/mhb/" />
  ```
Notation RDF/XML

• Relations between resources by nesting

```xml
<mhb:Lecture rdf:about="http://dws.informatik.uni-mannheim.de/teaching/sw">
  <mhb:givenBy>
    <mhb:Lecturer rdf:about="http://dws.informatik.uni-mannheim.de/heiko"/>
  </mhb:givenBy>
</mhb:Lecture>
```

• Relations between resources by explicit links

```xml
<mhb:Lecturer rdf:about="http://dws.informatik.uni-mannheim.de/heiko"/>

<mhb:Lecture rdf:about="http://dws.informatik.uni-mannheim.de/teaching/sw">
  <mhb:givenBy rdf:resource="http://dws.informatik.uni-mannheim.de/heiko"/>
</mhb:Lecture>
```
Notation RDF/XML

• An RDF graph may contain cycles
• XML may not → explicit links are necessary

```
<mhb:University rdf:about="http://www.uni-mannheim.de">
  <mhb:hasEmployee>
    <mhb:UniversityMember rdf:about="http://www.heikopaulheim.com/">
      <mhb:worksFor rdf:resource="http://www.uni-mannheim.de"/>
    </mhb:UniversityMember>
  </mhb:hasEmployee>
</mhb:University>
```
JSON-LD Notation

- JSON is popular in script programming
- JSON-LD: Standard for serializing RDF in JSON

```json
{
    "@id": "http://www.heikopaulheim.com/",
    "http://dws.informatik.uni-mannheim.de/name": "Heiko Paulheim",
    "http://dws.informatik.uni-mannheim.de/teaches": {
        "http://http://www.w3.org/1999/02/22-rdf-syntax-ns#type": "http://dws.informatik.uni-mannheim.de/Lecture",
        "http://dws.informatik.uni-mannheim.de/title": "Semantic Web"
    }
}
```
Blank Nodes

- Information that is not or cannot be specified
  - "Dieter Fensel has written a book about the Semantic Web"

![Diagram](image-url)
Blank Nodes

• Information that is not or cannot be specified
  – "Dieter Fensel has written something about the Semantic Web."

Dieter Fensel \( \xrightarrow{\text{dc:creator}} \) "Semantic Web"
Blank Nodes in Turtle

• Variant 1: explicitly named with an underscore

```turtle
:Dieter_Fensel dc:creator _:x .
_:x a :Book ;
   dc:subject "Semantic Web" .
```

• Variant 2: unnamed with square brackets

```turtle
:Dieter_Fensel dc:creator
[ a :Book;
   dc:subject "Semantic Web" ].
```

• Notes:
  – both are equivalent
  – changing blank node names does not change the semantics!
Application of Blank Nodes: n-ary Predicates

• RDF predicates always connect a subject and an object
  – i.e., in the sense of predicate logic, they are binary predicates
    
    \[
    \text{:Heiko :works\_for :UniMannheim .} \\
    \iff \text{works\_for(Heiko, UniMannheim) .}
    \]

• Sometimes, n-ary predicates are needed
  – has\_ingredient(Recipe, Sugar, 100g)
Application of Blank Nodes: n-ary Predicates

:recipe :hasIngredient [ :ingredient :Sugar; :amount "100g" ] .

Recipe

has ingredient

Sugar

ingredient

amount

"100g"
Application of Blank Nodes: n-ary Predicates


How does this differ from the version on the previous slide?
Semantic Principles of RDF

• On the Web, "anybody can say anything about anything"
  – This is called the AAA principle (Allemang & Hendler)

• This principle also holds for the Semantic Web
Semantic Principles of RDF

• One thing can have multiple names

  :Munich :capitalOf :Bavaria .

• On the semantic web, there is not just one name for each thing
  – this is called the Non-unique name assumption

• This means: Just that two things have different names does not mean that they are different!
RDF: Intuition and Actual Semantics

• Let us consider the following example:

  :Peter :fatherOf :Julia ,
  :Mary .

• How many children does Peter have?

• Intuitively, we assume that Julia and Mary are two different persons
• However, this is not trivial for a machine
  – (and the assumption may even be wrong)
Semantic Principles of RDF

• We (probably) do not know all the contents of the Semantic Web
• Therefore, there may be more information on a resource than what we have

• This principle is called "Open World Assumption"
Let us consider this example again:

```
:Peter :fatherOf :Julia ,
       :Mary .
```

How many children does Peter have?

Intuition says: two children

However, he could also have three or more
(oder also just one, as we have learned just a minute ago)
RDF: Intuition and Actual Semantics

• Both
  – Non-unique Name Assumption and
  – Open World Assumption

will re-occur quite a few times in this lecture

• Hint: consider those two whenever something seems weird when interpreting RDF data
RDF and HTML

- The Semantic Web uses RDF
- The “classic” Web uses HTML

- Does that mean that each information has to be encoded twice?
  - once for humans, once for machines

```
<html>
  ...
  <b>Dr. Mark Smith</b>
  <i>Physician</i>
  Main St. 14
  Smalltown
  Mon-Fri 9-11 am
  Wed 3-6 pm
  ...
</html>
```

```
:p a :Physician .
:p :hasDegree "Dr." .
:p :hasName "Mark Smith" .
:p :hasAddress :a .
:a :street "Main Street" .
:a :number "14"^^xsd:int .
:a :city "Smalltown" .
p :hasOpeningHours [ a rdf:Bag ;
  [ :day :Monday;
    :from "9"^^xsd:int;
    :to "11"^^xsd:int;
  ]
  ...
```
Using RDF and HTML Together – Variant 1

- Explicit reference to a RDF version
  - an agent stumbling on the HTML page can download the RDF data file

```html
<html>
<head>
    <link rel="meta" type="application/rdf+xml" title="DC" href="dc.rdf" />
</head>
<body>
...
</body>
</html>

<?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/">
    <rdf:Description rdf:about="http://www.dws.informatik.uni-mannheim.de/mhb/sw">
        <dc:publisher>University of Mannheim</dc:publisher>
        <dc:subject>Semantic Web</dc:subject>
        <dc:creator>Heiko Paulheim</dc:creator>
        <dc:relation rdf:resource="http://www.w3.org/2001/sw/" />
    </rdf:Description>
</rdf:RDF>
```
Using RDF and HTML Together – Variant 2

- Content Negotiation

Intelligent Agent

Browser

Server

give me RDF

RDF

give me HTML

HTML
Content Negotiation in Detail

HTTP/1.1 303 See Other
Location: http://www.mannheim.de/index.html

GET / HTTP/1.1
Host: www.mannheim.de
Accept: text/html

Browser

Server
Content Negotiation in Detail

HTTP GET

HTTP/1.1 303 See Other
Location: http://www.mannheim.de/data.rdf

GET / HTTP/1.1
Host: www.mannheim.de
Accept: application/rdf+xml
Content Negotiation: MIME Types

- MIME: Multipurpose Internet Mail Extensions
- Original purpose: classifying e-mail attachments
  - Text, PDF, ..
- First version: 1996
- Administrated by IANA

- Important MIME types for the Semantic Web
  - application/rdf+xml
  - text/turtle
  - text/n3
  - application/json
  - application/sparql-query
  - application/sparql-results+xml
Using RDF and HTML Together

• Link to RDF Document
  – Can be done with a simple HTML editor
  – No special server configuration needed

• Content Negotiation
  – Requires particular server setup
  – *One* URI can be used for different representations

• Both cases require
  – two different representations
  – “double bookkeeping”

→ Potential source of inconsistencies!
RDF in Attributes (RDFa)

- Idea of RDFa
  - Why not encode HTML and RDF in one document
  - The essential information only has to be encoded once

- RDFa combines XHTML with RDF

```
<html>
  ...
  <body about="http://www.marcsmith.com/MarcSmith">
    <b><span property="doc:name">Dr. Mark Smith</span></b>
    <i><span property="doc:profession">Physician</span></i>
    <span rel="doc:address" href="http://www.marcsmith.com/Address">
      <span property="doc:street">Main Street</span>
      <span property="doc:number">14</span>
      <span property="doc:city">Smalltown</span>
    </span>
  </body>
  ...
</html>
```
RDFa Language Constructs

- **about="http://foo.bar/aSubject"**
  - Defines the subject of a page or section

- **property = "http://foo.bar/aProperty"**
  - Defines a relation
  - Contents of the tag are interpreted as a literal

- **rel = "http://foo.bar/aRelation"**
  - Defines a relation to another resource

- **href = "http://foo.bar/aResource"**
  - Defines a relation's object
  - can be the subject of a resource again

- **typeof = "http://foo.bar/aType"**
  - defines a resource's type
RDF in Attributes (RDFa)

```html
<html>
...<body about="http://www.marcsmith.com/MarcSmith">
  <b><span property="doc:name">Dr. Mark Smith</span></b>
  <i><span property="doc:profession">Physician</span></i>
  <span rel="doc:address" href="http://www.marcsmith.com/Address">
    <span property="doc:street">Main Street</span>
    <span property="doc:number">14</span>
    <span property="doc:city">Smalltown</span>
  </span>
...</body>
</html>
```

http://www.marcsmith.com/MarcSmith

```
doc:name
"Dr. Marc Smith"
```
RDF in Attributes (RDFa)

```html
<html>
  ...
  <body about="http://www.marcsmith.com/MarcSmith">
    <b><span property="doc:name">Dr. Mark Smith</span></b>
    <i><span property="doc:profession">Physician</span></i>
    <span rel="doc:address" href="http://www.marcsmith.com/Address">
      <span property="doc:street">Main Street</span>
      <span property="doc:number">14</span>
      <span property="doc:city">Smalltown</span>
    </span>
  </body>
</html>
```
RDF in Attributes (RDFa)

```
<html>
  ...
  <body about="http://www.marcsmith.com/MarcSmith"> 
    <b><span property="doc:name">Dr. Mark Smith</span></b><span property="doc:profession">Physician</span>
    <span rel="doc:address"> 
      <span property="doc:street">Main Street</span><br>
      <span property="doc:number">14</span><br>
      <span property="doc:city">Smalltown</span>
    </span>
  </body>
</html>
```

Relations without "href" become blank nodes!
Alternative to RDFa: Microdata

- Adding structured information to web pages
  - By marking up contents
  - Arbitrary vocabularies are possible
  - Introduced with HTML5
- Similar to RDFa

```html
<div itemscope itemtype="http://schema.org/PostalAddress">
  <span itemprop="name">Data and Web Science Group</span>,
  <span itemprop="addressLocality">Mannheim</span>,
  <span itemprop="postalCode">68131</span>,
  <span itemprop="addressCountry">Germany</span>
</div>
```
Alternative to RDFa: Microdata

• Markup can be extracted to RDF
  – See W3C Interest Group Note: Microdata to RDF [1]

Alternative to RDFa: Microdata

• Commonalities
  – Arbitrary classes/predicates are possible
  – Although Microdata is mainly used with schema.org

• Differences
  – Microdata is slightly less expressive
  – No URIs, only blank nodes
  – No cycles in the RDF graph
  – No reification (see later)
RDFa, MicroFormats, and Microdata

- **MicroFormats**: fixed vocabularies for persons, addresses, etc.

- **WebDataCommons**: Large-Scale Extraction of RDFa, MicroFormats, and Microdata from the Web

http://webdatacommons.org/structureddata/2018-12/stats/stats.html
RDF Tools: Storage

• RDF is often stored in relational databases
• Different storage strategies
  – single triple table
  – one table per class
  – one table per property
• Strategies differ
  – by requirements of (disk) space
  – by query response time for different query types
• Examples: Virtuoso, Sesame, ...
RDF Tools: Visualization

• Mostly graph-based visualization tools
RDF Tools: Validation

• W3C RDF Validator:
  – W3C RDF Validator: http://www.w3.org/RDF/Validator/
  – Output of RDF/XML and graphs

• EasyRDF Validator:
  – http://www.easyrdf.org/converter
  – Understands and converts a variety of notations
RDF Tools: Programming, Reasoning

• Programming Frameworks
  – for developing RDF-based applications
  – e.g., JENA, RDFReactor, ...

• Reasoners
  – can draw logical conclusions from RDF graphs
  – can answer queries on RDF graphs

• Both will be covered in separate lectures
Metadata for RDF

• Recap: Dublin Core was designed as Metadata for the Web
• On the Semantic Web, we may have metadata as well

• Most prominently: *provenance*
  – Where does the data come from?
  – Who created it?
  – When was it created?
  – What was the process creating it?
  – …

https://www.w3.org/TR/prov-o/
Reification

- Latin *res* ("Thing"), *facere* ("make")
  - an Explication
  - making a statement, an opinion etc. the subject of a statement
- In RDF: Statements about statements

- "Peter says that Rome is the capital of Spain."

Implementation:
  - RDF Statements are considered resources themselves
  - Can be subject or object of other statements
- Reification can have multiple levels
  - “Peter says that Wikipedia states that Rome is the capital of Spain.”
Reification in RDF

Peter says that Rome is the capital of Spain.
Implementing Reification as Standard RDF

Peter says that Rome is capital of Spain:

```
<rdf:Statement rdf:type="rdf:Statement">
  <rdf:subject rdf:resource="http://example.com/rome"/>
  <rdf:predicate rdf:resource="http://example.com/is-capital-of"/>
  <rdf:object rdf:resource="http://example.com/spain"/>
</rdf:Statement>
```
Encoding Reification in Turtle

- **Variant 1: Named Statement (with URI)**

  ```turtle
  :triple1 rdf:type rdf:Statement ;
  rdf:subject :Rome ;
  rdf:predicate :isCapitalOf ;
  rdf:object :Spain .
  :Peter :says :triple1 .
  ```

- **Variant 2: Unnamed Statement (Blank Node)**

  ```turtle
  :Peter :says [ a rdf:Statement ;
  rdf:subject :Rome ;
  rdf:predicate :isCapitalOf ;
  rdf:object :Spain .
  ] .
  ```
Reification in the Wild

- Example: the WebIsALOD dataset

http://webisa.webdatacommons.org/

http://webisa.webdatacommons.org/
Design Decisions: Provenance in WebIsALOD

- Variant 1: RDF Reification
• Variant 2: Singleton Property
Design Decisions: Provenance in WebIsALOD

- Variant 3: n-ary relations

```
isa:Web_Service_
isa:_GMail_
isa:broader_1
isa:broader_2

_:3214353
prov:wasDerivedFrom

_:4327432
prov:wasGeneratedBy

_:0924372
```
Design Decisions: Provenance in WebIsALOD

• Variant 4: NdFluents
Design Decisions: Provenance in WebIsALOD

• Variant 5: RDF Graphs
Design Decisions: Provenance in WebIsALOD

• Challenge 1: Verbosity
  – WebIsALOD has 400M core statements

• Challenge 2: Usability
  – Querying should not be overly complex
  – (see later)

• Challenge 3: Understandability
  – We should keep the learning curve low

• Challenge 4: Scalability
  – Support by RDF store should be considered

RDF Graphs are least verbose
NDFluents and reification lead to longer queries
Reification and RDF Graphs are standardized
Virtuoso recommends RDF Graphs
Wrap Up

• RDF is a language for describing arbitrary things
  – interpretation: set of statements or directed graph
  – Notations: RDF/XML, Turtle

• Special language constructs
  – Blank nodes
  – Reification and its variants

• Semantics
  – Non-unique name assumption
  – Open world assumption

• Embedding in HTML is possible

• Large set of tools is available
A Critical Look in the Rear View Mirror

• Is RDF more powerful than XML?

• XML is a markup language for information

• In XML, arbitrary elements and attributes can be defined

• XML tag names are meaningless for a computer

• RDF is a markup language for information

• In RDF, arbitrary classes and predicates can be defined

• RDF class and predicate names are meaningless for a computer
A Critical Look in the Rear View Mirror

• So, why did we spend an entire lecture on RDF?
A Critical Look in the Rear View Mirror

How standards proliferate:
(See: A/C chargers, character encodings, instant messaging, etc.)

Situation: There are 14 competing standards.

14?! Ridiculous! We need to develop one universal standard that covers everyone's use cases. Yeah!

Soon:

Situation: There are 15 competing standards.

http://xkcd.com/927/
Semantic Web – Architecture

here be dragons...

Semantic Web Technologies
(This lecture)

Technical Foundations

Berners-Lee (2009): Semantic Web and Linked Data
Questions?