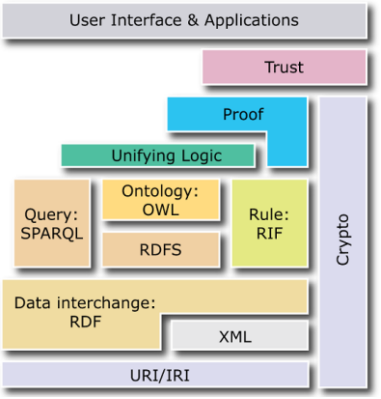
COMP3220:
Document Processing and the Semantic Web
Resource Description Framework (RDF)

Rolf Schwitter
Rolf.Schwitter@mq.edu.au

Today's Agenda

- RDF
- RDF Schema
- RML Mapping

The Semantic Web Architecture



- The Semantic Web architecture is based on the Resource Description Frameworks (RDF)

  which relies on
  - XML for syntax (but there are alternatives)
  - URI/IRIs for naming
  - Unicode for representation of text.
- RDF and RDF Schema form the lowest "semantic layer" of the Semantic Web architecture.
- Query, ontology and rule languages are layered on top.

RDF

- RDF stands for Resource Description Framework:

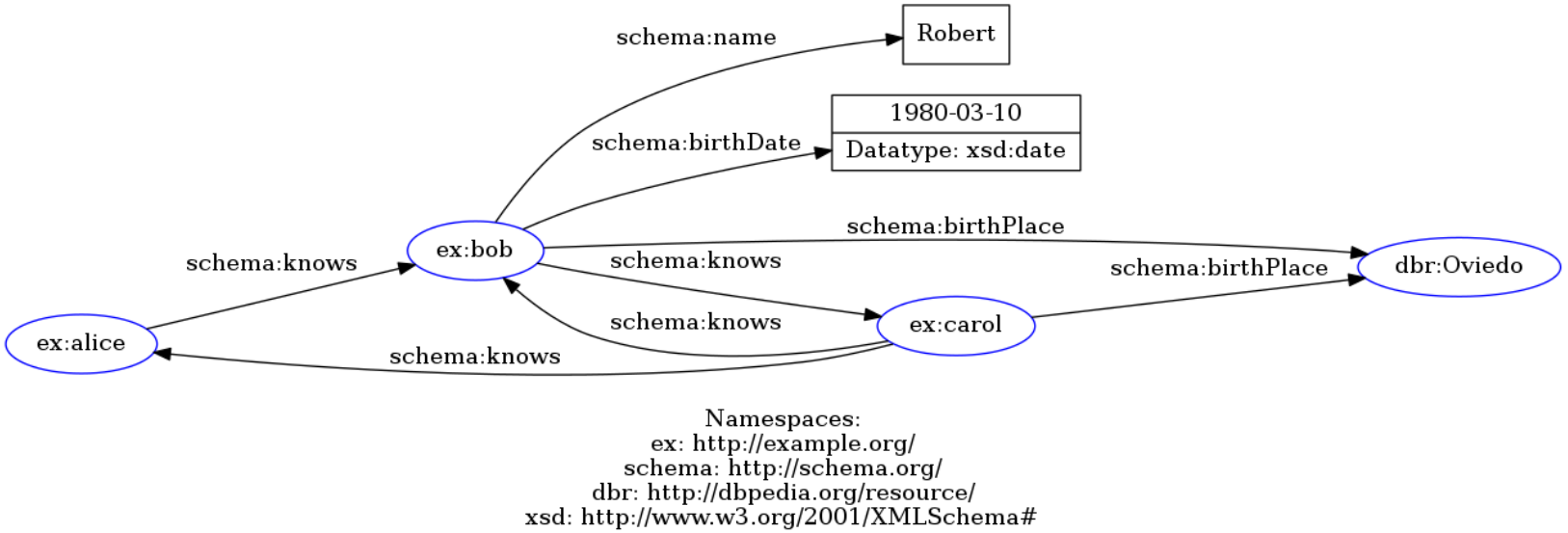
  `https://www.w3.org/TR/rdf11-primer/`
- RDF is a framework for describing resources on the web.
- RDF allows use to make statements about resources.

  `<subject> <predicate> <object>`
- The subject and object represent two related resources.
- The predicate represents the nature of the relationship.
- This relationship is also called an RDF property.
- RDF statements are also called triples.

RDF Graph

- An RDF graph is a set of RDF triples.
- An RDF triple consists of three components:
  - the subject, which is a IRI or a blank node
  - the predicate which is a IRI
  - the object which is a IRI, a literal or a blank node.
- Literals are used for values such as strings, numbers and dates.
- Blank nodes represent anonymous resources for which a IRI or literal is not given.

RDF Graph



RDF/XML Notation

```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:schema="http://schema.org/">

  <rdf:Description rdf:about="http://example.org/alice">
    <schema:knows rdf:resource="http://example.org/bob"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://example.org/bob">
    <schema:knows>
      <rdf:Description rdf:about="http://example.org/carol">
        <schema:knows rdf:resource="http://example.org/bob"/>
        <schema:knows rdf:resource="http://example.org/alice"/>
        <schema:birthPlace rdf:resource="http://dbpedia.org/resource/Oviedo"/>
      </rdf:Description>
    </schema:knows>

    <schema:name>Robert</schema:name>
    <schema:birthDate rdf:datatype="http://www.w3.org/2001/XMLSchema#date">1980-03-10</schema:birthDate>
    <schema:birthPlace rdf:resource="http://dbpedia.org/resource/Oviedo"/>
  </rdf:Description>

</rdf:RDF>
```

RDF/XML is an Exchange Syntax

- Remember: RDF's conceptual model is a graph.
- RDF provides an XML syntax for exchanging RDF graphs (RDF/XML).
- RDF/XML is not designed for being displayed to people.
- RDF/XML is designed to be read by computers.
- N-Triples is a simple line-based text format.
- Turtle is a human-readable format.

N-Triple Notation

```
<http://example.org/alice> <http://schema.org/knows> <http://example.org/bob> .
<http://example.org/bob> <http://schema.org/knows> <http://example.org/carol> .
<http://example.org/bob> <http://schema.org/name> "Robert" .
<http://example.org/bob> <http://schema.org/birthDate> "1980-03-10"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://example.org/bob> <http://schema.org/birthPlace> <http://dbpedia.org/resource/Oviedo> .
<http://example.org/carol> <http://schema.org/knows> <http://example.org/bob> .
<http://example.org/carol> <http://schema.org/knows> <http://example.org/alice> .
<http://example.org/carol> <http://schema.org/birthPlace> <http://dbpedia.org/resource/Oviedo> .
```

Turtle Notation

```
@prefix ex:     <http://example.org/> .
@prefix schema: <http://schema.org/> .
@prefix dbr:    <http://dbpedia.org/resource/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

ex:alice schema:knows      ex:bob .
ex:bob   schema:knows      ex:carol .
ex:bob   schema:name       "Robert" .
ex:bob   schema:birthDate  "1980-03-10"^^xsd:date .
ex:bob   schema:birthPlace dbr:Oviedo .
ex:carol schema:knows      ex:bob .
ex:carol schema:knows      ex:alice .
ex:carol schema:birthPlace dbr:Oviedo .
```

RDF in Turtle Notation (Simplified)

```
@prefix schema: <http://schema.org/> .
@prefix ex:     <http://example.org/> .
@prefix dbr:    <http://dbpedia.org/resource/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

ex:alice schema:knows  ex:bob .

ex:bob schema:name         "Robert" ;
       schema:birthDate    "1980-03-10"^^xsd:date ;
       schema:birthPlace   dbr:Oviedo ;
       schema:knows        ex:carol .

ex:carol schema:birthPlace  dbr:Oviedo ;
         schema:knows       ex:bob , ex:alice .
```

Blank Nodes

- A resource without a global identifier can be represented in RDF by a blank node.
- A blank node is like a simple variable in algebra.
- A blank node represents something without saying what its value is.
- A blank node can appear in the subject and object position of a triple.
- A blank node can be used to denote resources without explicitly naming them with an IRI.

Blank Nodes

```
@prefix schema: <http://schema.org/> .
@prefix ex:     <http://example.org/> .
@prefix dbr:    <http://dbpedia.org/resource/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

ex:alice schema:knows _:x .
_:x      schema:knows ex:dave .

ex:carol schema:knows _:y .

_:y      schema:birthPlace _:z ;
         schema:age "23"^^xsd:integer .

ex:dave  schema:birthPlace _:z .
```

RDF Containers

- RDF provides a container vocabulary.
- A container is a resource that contains things.
- The contained things are called members.
- Members may be resources (incl. blank nodes) or literals.
- RDF defines three types of containers:
  - **`rdf:Bag`**
  - **`rdf:Seq`**
  - **`rdf:Alt`**

RDF Containers

- `rdf:Bag` contains an unordered list of elements.
- `rdf:Seq` contains an ordered list of elements.
- `rdf:Alt` contains a list of alternative elements.
- Technically, the resource is given an `rdf:type` whose value is a container type:

```
ex:agmCommittee a rdf:Bag .
```

- Container membership properties have names of the form `rdf:_n`.
- Here *n* is a decimal integer greater than zero:

```
ex:agmCommittee rdf:_1 ex:Charles .
ex:agmCommittee rdf:_2 ex:Mark .
```

RDF Containers

- For example, the statement:

  The resolution was approved by the AGM Committee (as a whole) which has (as far as we know) the members Charles, Mark, and Roger.

  can be represented as a bag:

```
ex:resolution exterms:approvedBy ex:agmCommittee .
ex:agmCommittee a rdf:Bag .
ex:agmCommittee rdf:_1 ex:Charles .
ex:agmCommittee rdf:_2 ex:Mark .
ex:agmCommittee rdf:_3 ex:Roger .
```

RDF Collections

- Containers only say that certain identified resources are members.
- Containers do not say that other members do not exist.
- Collections describe groups containing only the specified members.

Collections

```
@prefix :       <http://example.org/> .
@prefix schema: <http://schema.org/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

:results schema:name "Marathon Results" ;
                :results _:1 .

_:1 rdf:first :dave .
_:1 rdf:rest  _:2 .
_:2 rdf:first :alice .
_:2 rdf:rest  _:3 .
_:3 rdf:first :bob .
_:3 rdf:rest  rdf:nil .
```

Collections (Simplified)

```
@prefix :       <http://example.org/> .
@prefix schema: <http://schema.org/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

:results schema:name "Marathon Results" ;
         :results (:dave :alice :bob) .
```

RDF Schema (Turtle Notation)

```
@prefix :       <http://example.org/> .
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix schema: <http://schema.org/> .

schema:Person  a rdfs:Class .

:Teacher a rdfs:Class ; rdfs:subClassOf schema:Person .

:teaches a rdfs:Property ;
         rdfs:domain :Teacher ; rdfs:range  :Course .

# RDF triples
:alice a        :Person .
:bob   a        :Teacher .
:carol :teaches :algebra .
```

Python and RDFLib

- RDFLib is a Python library for working with RDF:
  - **`https://pypi.python.org/pypi/rdflib`**
- The library includes parsers and serializers for:
  - RDF/XML, N-Triples, Turtle, RDFa, etc.
- The library includes a SPARQL 1.1 engine.
- The library includes also a wrapper for remote SPARQL endpoints.
- Installation: pip install rdflib

Input Document (Turtle Notation)

```
@prefix schema: <http://schema.org/> .
@prefix ex:     <http://example.org/> .
@prefix dbr:    <http://dbpedia.org/resource/> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

ex:alice schema:knows _:x .
_:x      schema:knows ex:dave .

ex:carol schema:knows _:y .

_:y      schema:birthPlace _:z ;
         schema:age "23"^^xsd:integer .

ex:dave  schema:birthPlace _:z .
```

Python and RDFLib

```
import rdflib

g = rdflib.Graph()
res = g.parse("example.rdf", format="turtle")

print("The graph has %s statements." % len(res))
print()

for subj, pred, obj in res:
   print(subj)
   print(pred)
   print(obj)
   print()

output = res.serialize(format='pretty-xml').decode("utf-8")
print(output)
```

Output

```
The graph has 6 statements.

http://example.org/carol
http://schema.org/knows
f76d657504d524d9b982719c7d2d67faeb2

f76d657504d524d9b982719c7d2d67faeb1
http://schema.org/knows
http://example.org/dave

http://example.org/dave
http://schema.org/birthPlace
f76d657504d524d9b982719c7d2d67faeb3

http://example.org/alice
http://schema.org/knows
f76d657504d524d9b982719c7d2d67faeb1

f76d657504d524d9b982719c7d2d67faeb2
http://schema.org/age
23
```

Output: RDF/XML

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:schema="http://schema.org/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
>
  <rdf:Description rdf:about="http://example.org/carol">
    <schema:knows>
      <rdf:Description rdf:nodeID="f76d657504d524d9b982719c7d2d67faeb2">
        <schema:birthPlace rdf:nodeID="f76d657504d524d9b982719c7d2d67faeb3"/>
        <schema:age rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">23</schema:age>
      </rdf:Description>
    </schema:knows>
  </rdf:Description>
  <rdf:Description rdf:about="http://example.org/alice">
    <schema:knows>
      <rdf:Description rdf:nodeID="f76d657504d524d9b982719c7d2d67faeb1">
        <schema:knows rdf:resource="http://example.org/dave"/>
      </rdf:Description>
    </schema:knows>
  </rdf:Description>
  <rdf:Description rdf:about="http://example.org/dave">
    <schema:birthPlace rdf:nodeID="f76d657504d524d9b982719c7d2d67faeb3"/>
  </rdf:Description>
</rdf:RDF>
```

From Data to RDF via RML

- But how do we generate RDF triples?
- We can use a mapping rule language to transform structured data into an RDF knowledge graph.
- The RDF Mapping Language (RML) is defined for structured input formats (CSV, JSON, and XML):

  `https://rml.io/specs/rml/`
- SDM-RDFizer is mapping rule interpreter for RML:

  `https://github.com/SDM-TIB/SDM-RDFizer`

RDF Mapping Language (RML)

Example input data

```
id,stop,latitude,longitude
6523,25,50.901389,4.484444
```

Example RML mapping

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix rml: <http://semweb.mmlab.be/ns/rml#>.
@prefix ql: <http://semweb.mmlab.be/ns/ql#>.
@prefix transit: <http://vocab.org/transit/terms/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>.
@base <http://example.com/ns#>.

<#AirportMapping> a rr:TriplesMap;
  rml:logicalSource [
    rml:source "Airport.csv" ;
    rml:referenceFormulation ql:CSV
  ];
  rr:subjectMap [
    rr:template "http://airport.example.com/{id}";
    rr:class transit:Stop
  ];

  rr:predicateObjectMap [
    rr:predicate transit:route;
    rr:objectMap [
      rml:reference "stop";
      rr:datatype xsd:int
      ]
    ];

  rr:predicateObjectMap [
    rr:predicate wgs84_pos:lat;
    rr:objectMap [
      rml:reference "latitude"
    ]
  ];

  rr:predicateObjectMap [
    rr:predicate wgs84_pos:long;
    rr:objectMap [
      rml:reference "longitude"
    ]
  ].
```

Example output data

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix transit: <http://vocab.org/transit/terms/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>.

<http://airport.example.com/6523> rdf:type transit:Stop.
<http://airport.example.com/6523> transit:route "25"^^xsd:int.
<http://airport.example.com/6523> wgs84_pos:lat "50.901389".
<http://airport.example.com/6523> wgs84_pos:long "4.484444".
```


https://rml.io/specs/rml/

Take-Home Messages

- RDF is a graph-based framework for describing resources.
- Resources can be anything, including documents, people, physical objects, and abstract concepts.
- An RDF statement consists of a subject, predicate, object.
- RDF uses IRIs and literals, (and blank nodes) to express descriptions of resources.
- There exist different serialization formats for RDF.
- RDF Schema provides a data-modelling vocabulary for RDF.
- RML is a mapping language from structured data to RDF.