Semantic Web

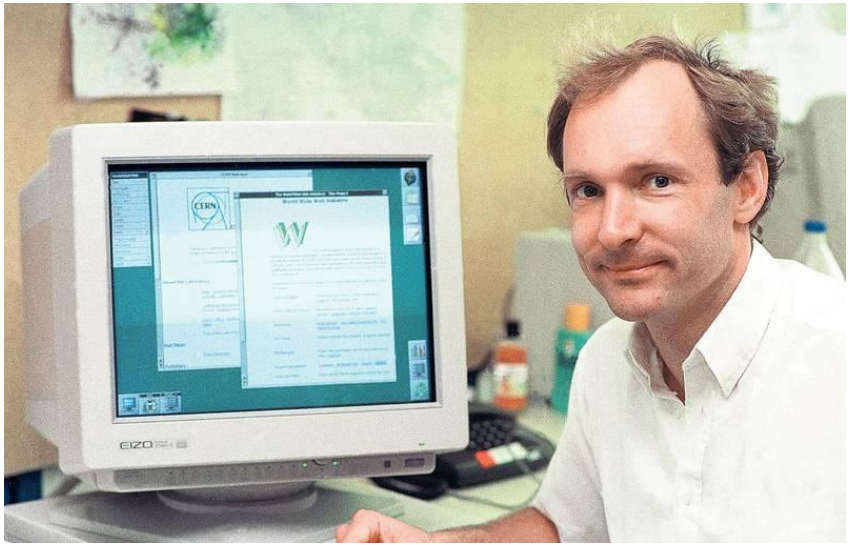
(KEN3140)

Lecture 2: Resource Description Framework (RDF)

Michel Dumontier and Kody Moodley

Maastricht University

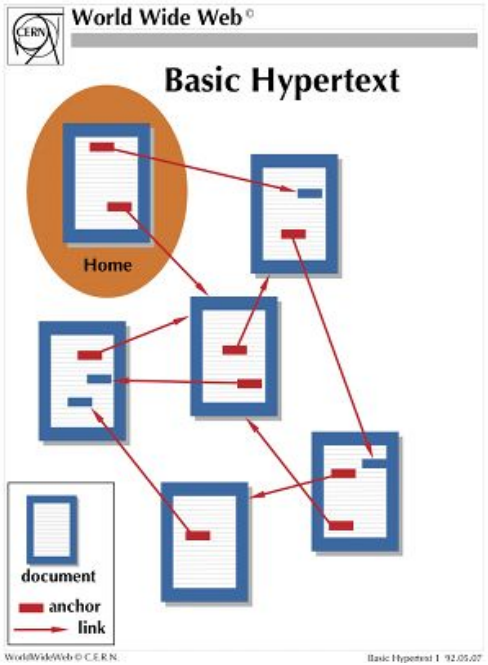
Institute of Data Science

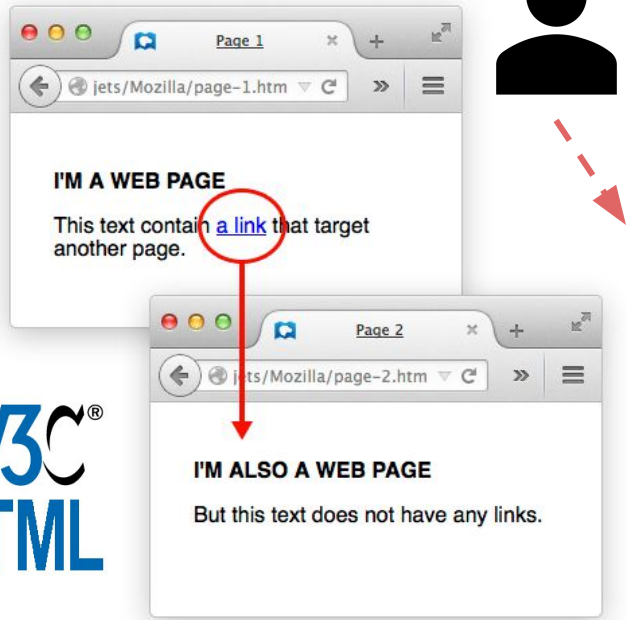


Tim Berners-Lee

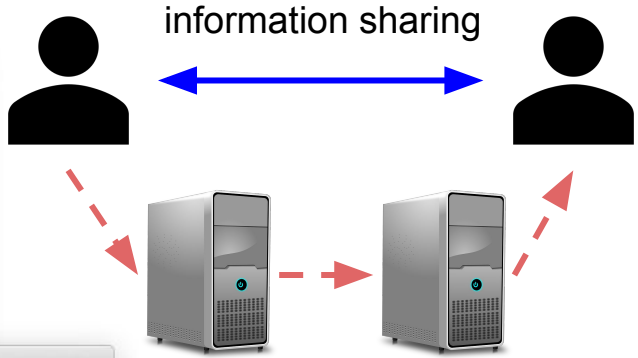
Inventor of the World Wide Web

World Wide Web





W3C® HTML



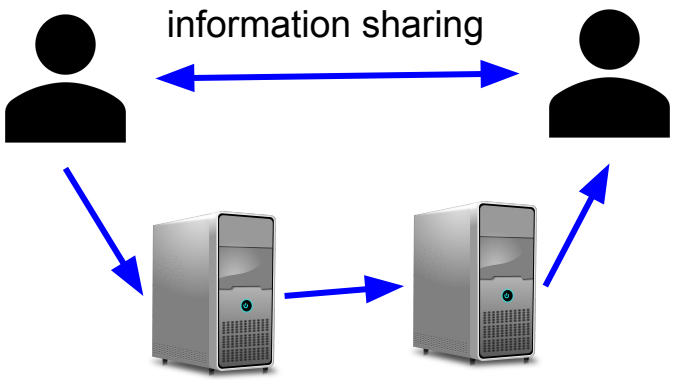
Humans have to make sense of (extract **knowledge** from) the information content

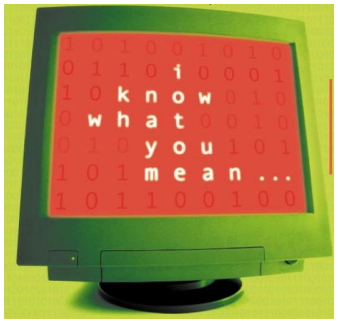
Semantic Web

Tim Berners-Lee, 2001

"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling **computers and people** to work in cooperation."

Scientific American, May 2001



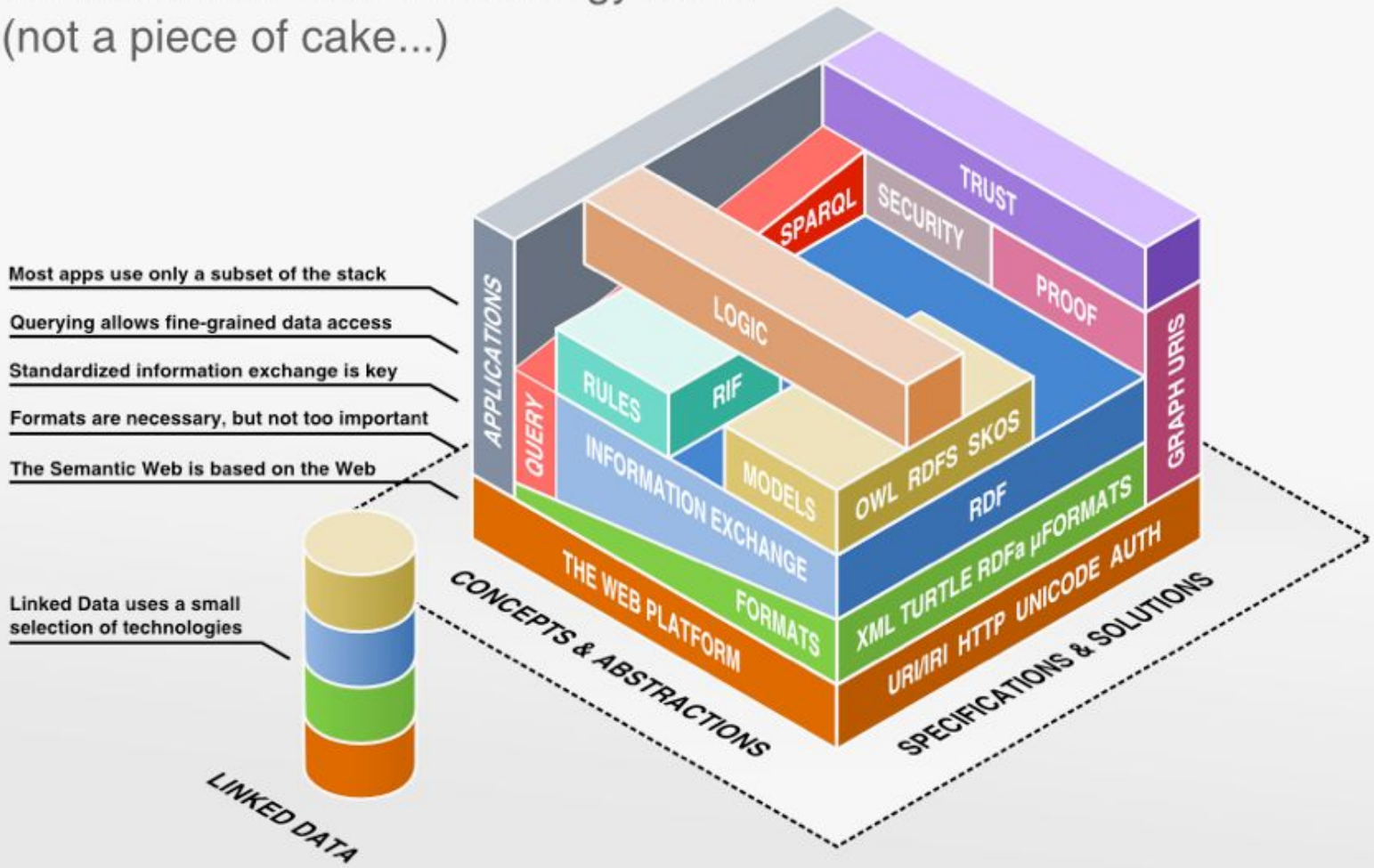


Humans & computers alike can interpret the information and share a common understanding of its meaning

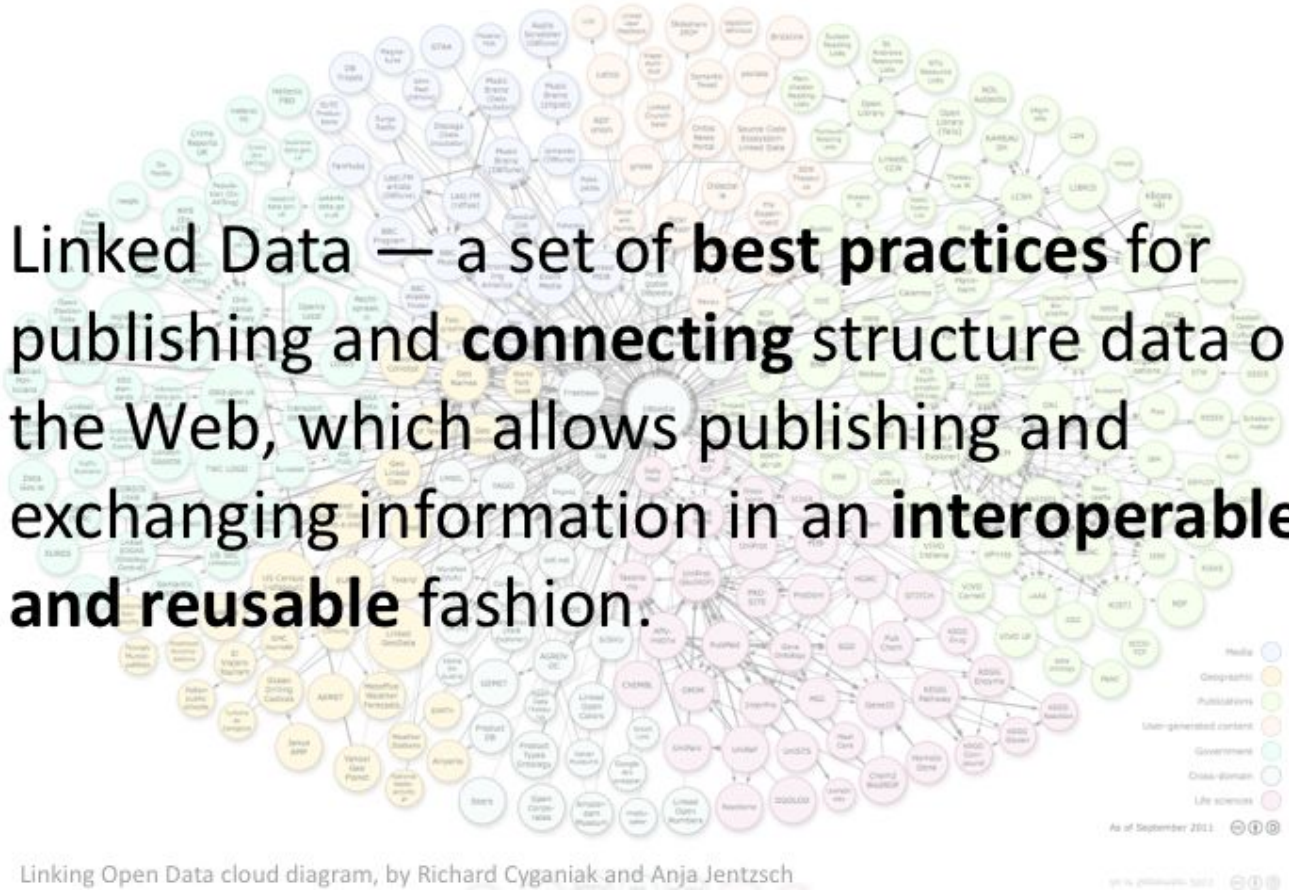
World Wide Web Consortium (W3C)

- Working Group for developing and maintaining Web Standards
- These are the people that defined HTML, XML, CSS etc.
- **“Okay great, Tim Berners-Lee! Semantic Web is a nice vision, but how are we going to make this happen?” We need to devise new ways of**:
- Representing and exchanging information between humans, between humans and machines, and between machines. New data formats (RDF, RDFS)
- Capturing the meaning (semantics) of information. Formal languages with mathematically defined meaning for representing information (OWL)
- Retrieving or querying information to answer questions (new query languages) - SPARQL
- Automatically inferring implicit facts from the information (reasoning algorithms) - RDFS and OWL reasoning algorithms
- …
- All of these new standards should be compatible with / build on top of existing Web standards (e.g. HTML, XML etc.)

The Semantic Web Technology Stack
(not a piece of cake...)



Network of Linked Data

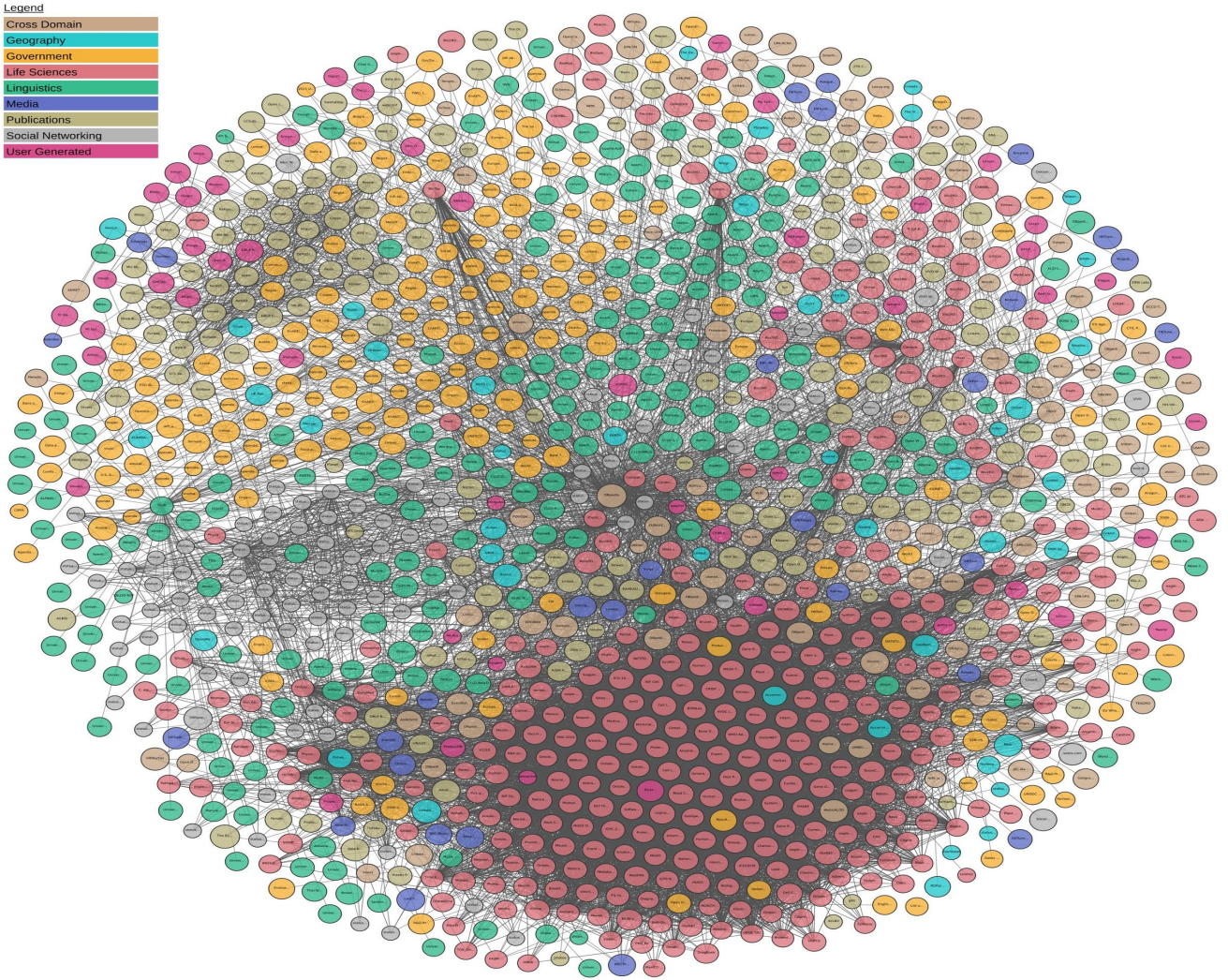


Linked Data — a set of **best practices** for publishing and **connecting** structure data on the Web, which allows publishing and exchanging information in an **interoperable and reusable** fashion.

X Linking Open Data cloud diagram, by Richard Cyganiak and Anja Jentzsch

The Linked Open Data Graph

https://lod-cloud.net/



The Linked Open Data Cloud from lod-cloud.net

Linked Data Principles

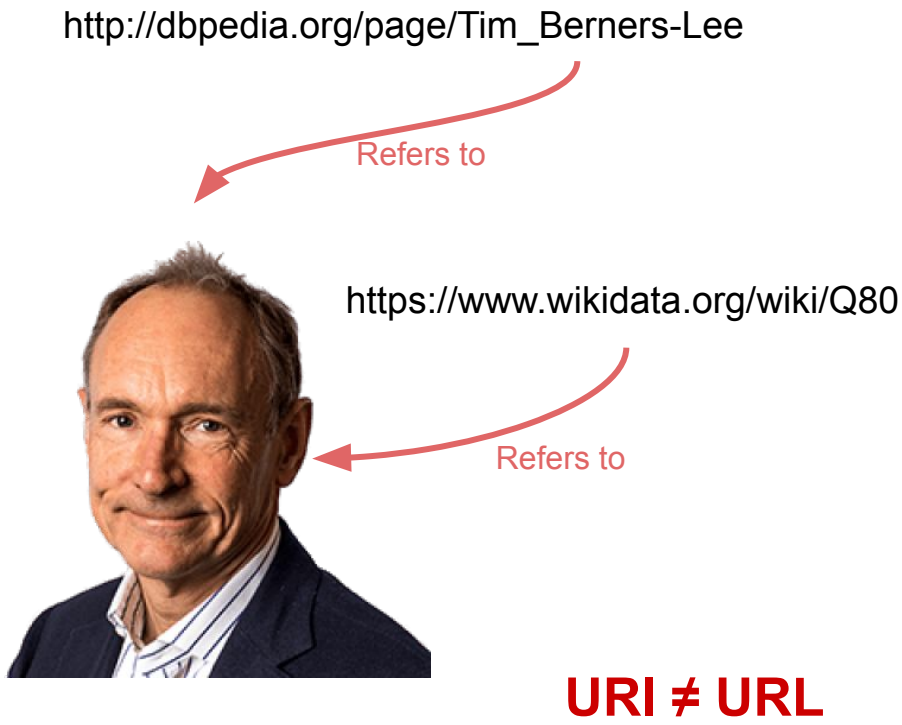
1. Use **Uniform Resource Identifiers (URIs)** as names for things
2. Use **HTTP URIs**, so that people can look up those names
3. When someone looks up a URI, provide **useful information**, using Semantic Web standards
4. Include **links** to other URIs, so that they can discover more things

URI as names for things (1)

Uniform Resource Identifier: “string of characters that unambiguously (uniquely) identifies a particular resource…”

“... a **name** given to describe a thing. This thing can be anything in the world (**either physical or abstract**).”

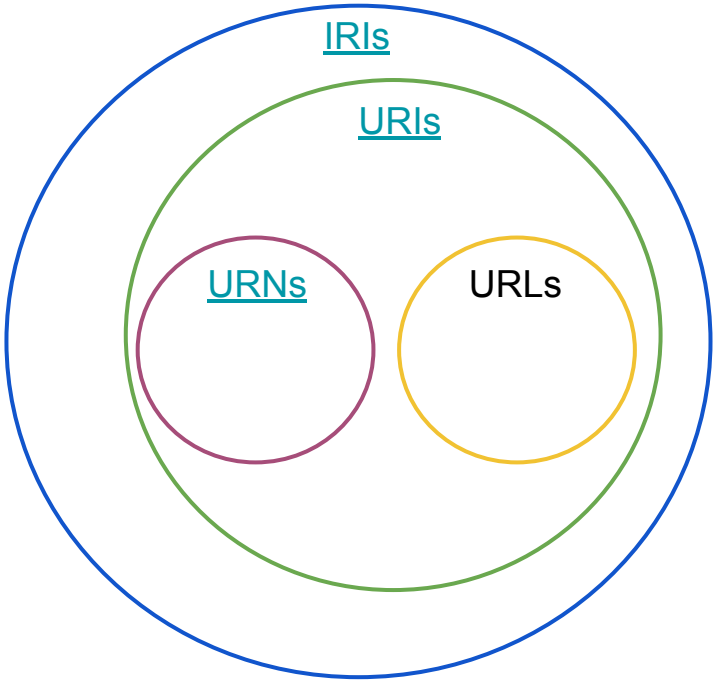
Things (also called resources or entities) can have **multiple** URIs



Namespaces, IRIs, URIs and URLs

- This URI has a “Slash” namespace (last character preceding the resource name is a “/” character).
- Can also have “Hash” namespaces in URIs e.g. https://www.w3.org/1999/02/22-rdf-syntax-ns#type
- Other identifiers in this family: Internationalized Resource Identifier (IRI), Uniform Resource Locator (URL), Uniform Resource Name (URN)
- **NB:** “resources” are the “things” Tim Berners-Lee refers to in the Linked Data Principles

IRIs, URIs and URLs

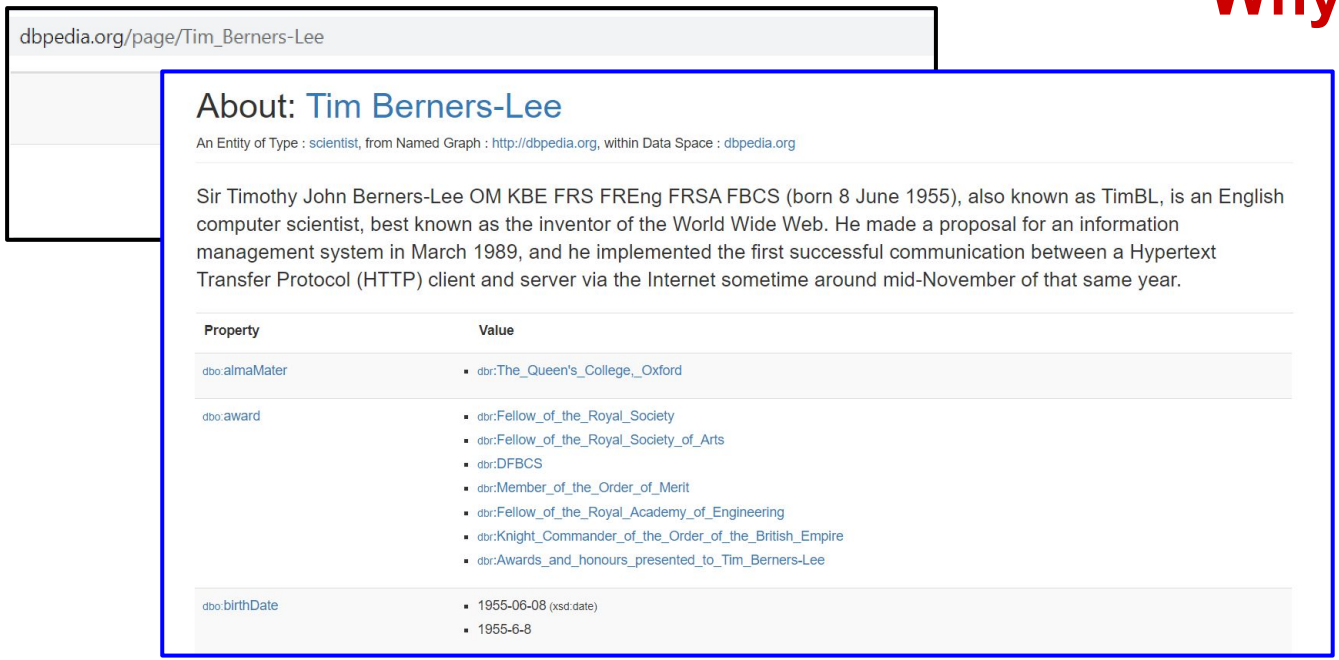


- URI: name of thing (or reference to it)
- URL: location of thing
- URN: Uniform Resource Name - a globally unique and persistent URI (even after a resource ceases to exist this identifier will remain). Protocol of URN is always “urn:” - **rare in practice**
- URI vs IRI? Small difference: URI uses only the US-ASCII character set. IRIs can use Universal Coded Character Set (UCS) characters which extends US-ASCII
- Further reading on differences: link
- Sometimes URI = URL (this is a recommended practice for Linked Data but is not always so easy to implement in practice)
- URI best practices reading here and here

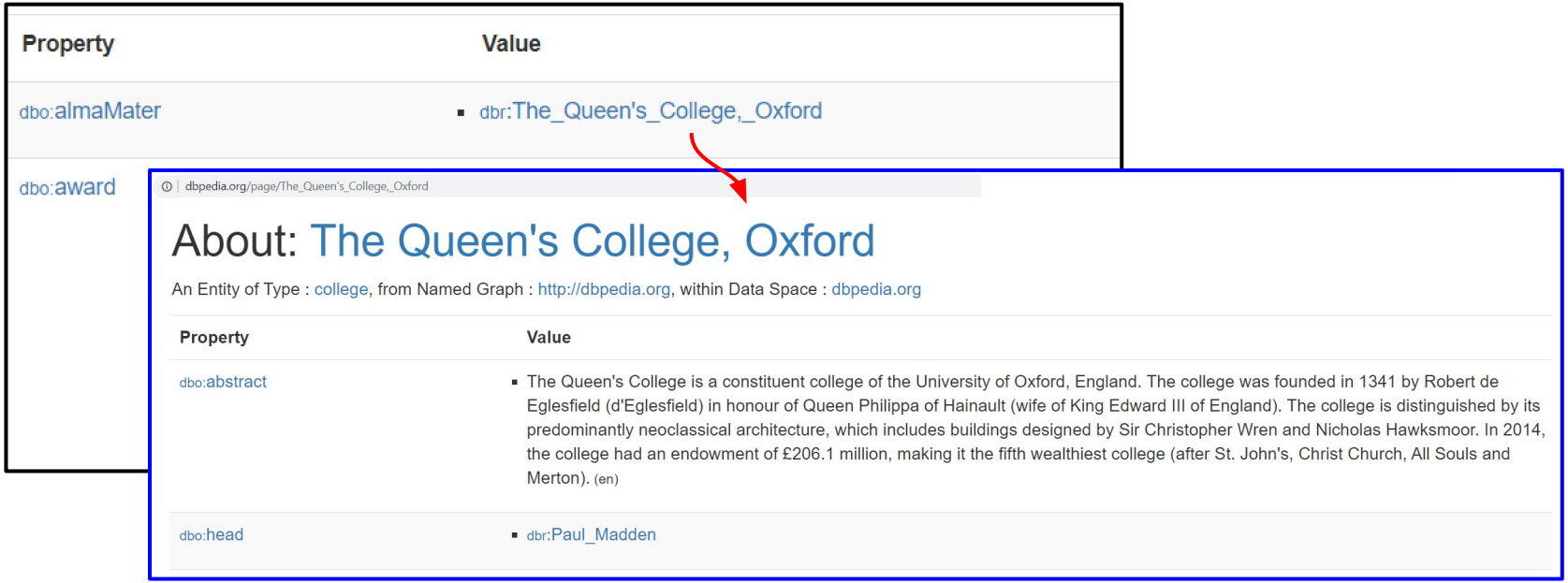
Use HTTP URIs (2)...and provide useful info (3)

http://dbpedia.org/page/Tim_Berners-Lee

Why?



Provide links to related entities (4)



Poll point 1

Long history of Knowledge Representation (KR)

- Existential Graphs (1896; Peirce)
- Semantic Networks “semnets” (Introduced mid-1960s, hype 1980s),
- Conceptual Graphs
- Cognitive Semantic Networks
- Structured Inheritance Networks
- Multilayered Extended Semantic Networks (MultiNets)
- Basic Conceptual Graphs
- Full Conceptual Graphs
- Hierarchical Semantic Form
- **Resource Description Framework (RDF)**
- Property Graph

Resource Description Framework (RDF)

RDF is a framework to represent information on the Web

RDF was developed by the W3C and first became a W3C recommendation in 1999. This was superseded by the RDF 1.0 specification in 2004, and the RDF 1.1 specification in 2014. The full RDF 1.1 specification consists of:

- An abstract data model with a set of implementing syntaxes for storing and exchanging information represented in RDF
- a formal semantics
- An accompanying RDF query language - SPARQL
- the RDF Schema vocabulary with extends RDF with additional features

RDF is highly versatile

RDF can be used to describe things in a machine readable manner. This can include **digital objects** such as documents, **real world objects** such as people, and **concepts** such as Person or Organization.

RDF is a graph formalism in that the intention is to describe information about interconnected entities that can be depicted in a directed graph.

RDF spec (and a word on W3C specs in general)

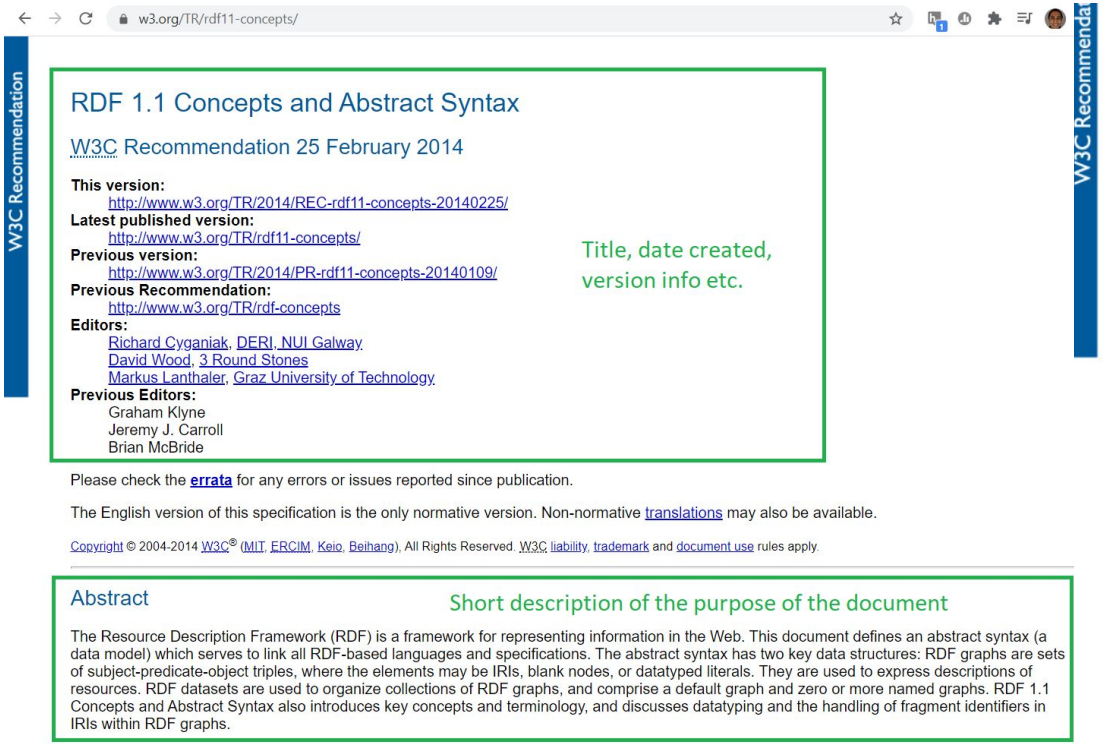


Table of Contents

RDF spec (and a word on W3C specs in general)

1.1 Graph-based Data Model

The core structure of the abstract syntax is a set of triples, each consisting of a subject, a predicate and an object. A set of such triples is called an RDF graph. An RDF graph can be visualized as a node and directed-arc diagram, in which each triple is represented as a node-arc-node link.

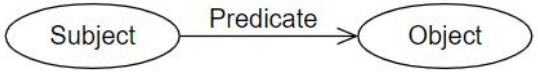


Fig. 1 An RDF graph with two nodes (Subject and Object) and a triple connecting them (Predicate)

There can be three kinds of nodes in an RDF graph: IRIs, literals, and blank nodes.

EXAMPLE 3

```
_:alice <http://xmlns.com/foaf/0.1/knows> _:bob <http://example.org/graphs/john> .
_:bob <http://xmlns.com/foaf/0.1/knows> _:alice <http://example.org/graphs/james> .
```

NOTE

N-Quads documents do not provide a way of serializing empty graphs that may be part of an RDF dataset.

RDF spec (and a word on W3C specs in general)

B. References

B.1 Normative references

[BCP47]
A. Phillips; M. Davis. *Tags for Identifying Languages*. September 2009. IETF Best Current Practice. URL: http://tools.ietf.org/html/bcp47

[NFC]
M. Davis, Ken Whistler. *TR15, Unicode Normalization Forms.*. 17 September 2010, URL: http://www.unicode.org/reports/tr15/

[RFC2119]
S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels.* March 1997. Internet RFC 2119. URL: http://www.ietf.org/rfc/rfc2119.txt

[RFC3987]
M. Dürst; M. Suignard. *Internationalized Resource Identifiers (IRIs)*. January 2005. RFC. URL: http://www.ietf.org/rfc/rfc3987.txt

[UNICODE]
The Unicode Standard. URL: http://www.unicode.org/versions/latest/

[XMLSCHEMA11-2]
David Peterson; Sandy Gao; Ashok Malhotra; Michael Sperberg-McQueen; Henry Thompson; Paul V. Biron et al. *W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes*. 5 April 2012. W3C Recommendation. URL: http://www.w3.org/TR/xmlschema11-2/

B.2 Informative references

[COOLURIS]
Leo Sauermann; Richard Cyganiak. *Cool URIs for the Semantic Web*. 3 December 2008. W3C Note. URL: http://www.w3.org/TR/cooluris

[DOM4]
Anne van Kesteren; Aryeh Gregor; Ms2ger; Alex Russell; Robin Berjon. *W3C DOM4*. 4 February 2014. W3C Last Call Working Draft. URL: http://www.w3.org/TR/dom/

[HTML-RDFA]
Manu Sporny. *HTML+RDFa 1.1*. 22 August 2013. W3C Recommendation. URL: http://www.w3.org/TR/html-rdfa/

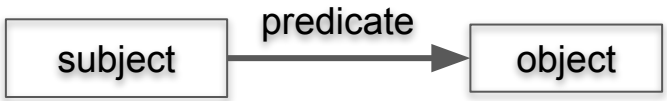
[HTML5]
Robin Berjon; Steve Faulkner; Travis Leithead; Erika Doyle Navara; Theresa O'Connor; Silvia Pfeiffer. *HTML5*. 4 February 2014. W3C Candidate Recommendation. URL: http://www.w3.org/TR/html5/

[JSON-LD]
Manu Sporny, Gregg Kellogg, Markus Lanthaler, Editors. *JSON-LD 1.0*. 16 January 2014. W3C Recommendation. URL: http://www.w3.org/TR/json-ld/

The RDF Data Model

The fundamental unit of an RDF graph is a statement called a **triple**. An RDF graph is essentially a set of triples.

Each triple consists of three (hence the name triple) components: a **subject**, a **predicate,** and an **object**. The triple expresses a relationship between the subject and the object. The subject and object represent nodes in the RDF graph and the predicate represents an edge / relation between these nodes



Subjects, Predicates, Objects

...and this is where the connection with Linked Data comes in:

RDF Subjects and Predicates are denoted by IRIs

RDF Objects are denoted by either an IRI or a “literal value”

Starry night (June 30, 1889) by Vincent van Gogh

Subject	**Predicate**	**Object (IRI)**
<http://somenamespace.com/**starrynight**>	<http://somenamespace.com/**createdBy**>	<http://somenamespace.com/**Vincent_van_Gogh**> .

Subject	**Predicate**	**Object (Literal)**
<http://somenamespace.com/**starrynight**>	<http://somenamespace.com/**dateCreated**>	“1889-06-30”^^xsd:date .

RDF Nodes

There are three kinds of nodes in an RDF graph: IRIs, **literals**, and blank nodes.

A **literal** has up to 3 parts:

1) a **lexical form** encoded as a Unicode string in quotes (“...”)
2) a **datatype IRI**, that determines how the lexical form maps to a literal value.
3) and a **language tag only** when the datatype IRI is a rdf:langString

Observe that RDF uses ^^ to associate the datatype of a literal with is lexical form:
“1.01”^^<https://www.w3.org/2001/XMLSchema#decimal>
“1.01”^^xsd:decimal
“Maastricht University”@en
“Universiteit Maastricht”@nl

IRIs can be abbreviated (prefixes)

Full IRI:

```
http://www.w3.org/1999/02/22-rdf-syntax-ns#langString
```

Define an abbreviation for your namespace (called a prefix):

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

Shortened IRI:

```
rdf:langString
```

Data types of literals

The use of **XML Schema** data types is recommended (they are predefined and widely used)

Strings:	"Maastricht University"
Integers:	"1"^^<http://www.w3.org/2001/XMLSchema#integer>
Integers:	"1"^^xsd:integer
Decimals:	"1.23"^^xsd:decimal
Date:	"2014-9-11"^^xsd:date
Time:	"11:05:45"^^xsd:time
Date with time:	"2014-9-11T11:05:45"^^xsd:dateTime

RDF Nodes

There are three kinds of nodes in an RDF graph: IRIs, literals, and **blank nodes**.

Blank nodes are nodes that do not have a persistent identifier. They are convenient for complex data structures, and where there is no desire to assign a permanent identifier. They used in some RDF syntaxes and some RDF store implementations.

It is now generally believed to be a lazy/bad practice, so you should avoid them, and assign IRIs for all nodes instead!

RDF Predicates (data vs. object properties)

A predicate denotes a relationship between the subject and the object.
A predicate is specified with an IRI

Property is another term for predicate. Generally, when the object in a triple is a Literal value, we call the predicate a data property, if the object is an IRI sometimes call the predicate an object property

Subject **Predicate** Object property **Object (IRI)**

<http://somenamespace.com/**starrynight**> <http://somenamespace.com/**createdBy**> <http://somenamespace.com/**Vincent_van_Gogh**> **.**

Subject **Predicate** **Object (Literal)**

<http://somenamespace.com/**starrynight**> <http://somenamespace.com/**dateCreated**> “1889-06-30”^^xsd:date .

Data property

RDF Predicates (built-in)

RDF has some predicates built into its vocabulary that the specification developers decided would be needed and widely reused, independent of the type of information captured.

E.g. the IRI `http://www.w3.org/2000/01/rdf-schema#label` is a predicate defined by the RDFS vocabulary which can be used to attach a string label or name to an entity in the graph.

<http://somenamespace.com/**starrynight**> <http://www.w3.org/2000/01/rdf-schema**#label**> “The Starry Night”@en .

The IRI `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` is a special predicate defined by RDF which is used to specify a category (called a concept, class or type) to which an entity belongs (the “is a” relation). The category / type entities are usually defined in an ontology (learn more about this in the Ontologies and OWL sections of the course).

<http://somenamespace.com/**starrynight**> <http://www.w3.org/1999/02/22-rdf-syntax-ns**#type**> <http://somenamespace.com/**Painting**> .

RDF Predicates (external)

There are other predicates outside of RDF defined in other vocabularies on the Web. There are many of these vocabularies. Some domain-specific e.g. biomedicine and some domain-agnostic. There is an online Linked Data tool to browse for some of these: https://lov.linkeddata.es/dataset/lov/

E.g. the IRI `https://schema.org/employee` is a predicate defined by the Schema.org vocabulary (a vocabulary defined by a partnership between the world's major search engines).

<http://somenamespace.com/**maastricht_university**> <https://schema.org/**employee**> <http://somenamespace.com/**kody**> .

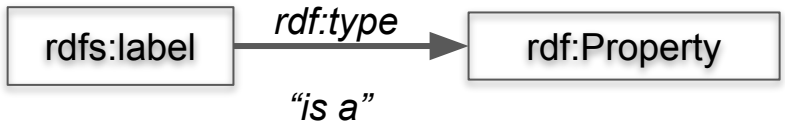
Note: the same applies to RDF subjects and IRI objects that represent types / classes (there are built-in types in RDF and a host of others in external vocabularies).

RDF Predicates (statements about properties)

A predicate denotes a relationship between the subject and the object.
A predicate is specified with an IRI

the IRI `http://www.w3.org/2000/01/rdf-schema#label` is a predicate defined by the RDFS vocabulary

By making statements about a predicate IRI, it then also becomes a node in the RDF graph.



From natural language to RDF data model

natural language

RDF is a W3C standard

abstract syntax
(data model)

From natural language to RDF data model to machine readable statements

natural language

RDF is a W3C standard

abstract syntax
(data model)

serialization

```
<http://example.org/RDF>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://example.org/W3C-standard> .
```

Poll point 2

RDF syntaxes: RDF/XML

RDF/XML

N-Triples

Turtle

Namespace declarations

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    <rdf:Description
      rdf:about="http://example.org/RDF">
         <rdfs:label lang="en">Resource Description Framework</rdfs:label>
    </rdf:Description>
</rdf:RDF>
```

Uses the eXtensible Markup Language to represent RDF triples; can define and use namespaces.
A namespace in XML refers to a location on the web where resources are defined.

RDF Syntaxes: N-Triples

RDF/XML

N-Triples

Turtle

```
<http://example.org/RDF>
     <http://www.w3.org/2000/01/rdf-schema#label>
          "Resource Description Framework"@en .
```

Each line composed of subject, predicate, object
IRIs are encapsulated by angled brackets < >
Literals by quotes “”
lines terminated by periods .

RDF Syntaxes: Turtle

RDF/XML

N-Triples

Turtle

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix ex: <http://example.org>
ex:RDF          rdfs:label          "Resource Description Framework"@en .
```

Each line composed of subject, predicate, object
Prefixes can be defined, and used in a shorthand notation prefix:suffix that expands to valid IRI
IRIs are encapsulated by angled brackets < >
Literals by quotes “”
lines terminated by periods .

Turtle: Multiple statements about one resource

- A property of a resource may contain multiple values
- A list of values for a property of a resource can be specified by separating the values by commas:

```
<http://maastrichtuniversity.nl>
    ex:location ex:Maastricht, ex:Netherlands .
```

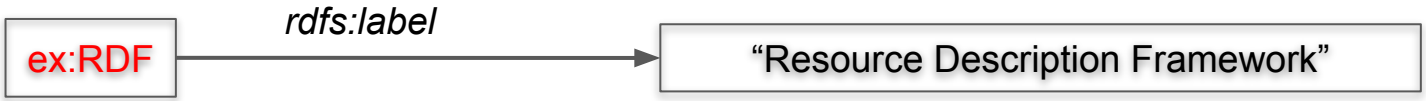
Multiple statements about one resource

- A resource may have multiple properties with values
- A list of properties with values of a resource can be specified by separating the properties with values by semicolons:

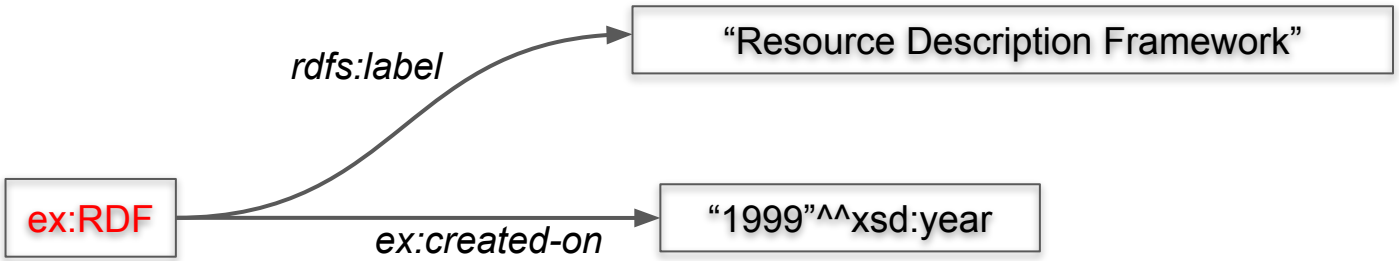
```
@prefix UM: <http://maastrichtuniversity.nl>
UM:
    rdfs:label "Maastricht University";
    rdf:type ex:University;
    ex:location ex:Maastricht, ex:Netherlands.
```

From resource description to data graph

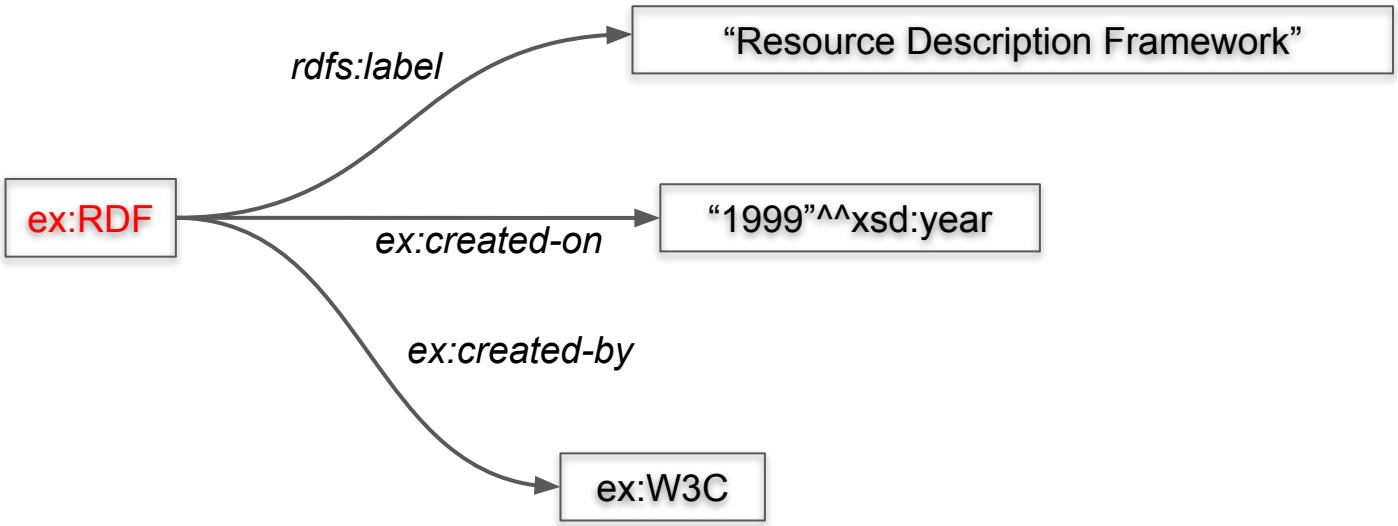
A first statement



A second statement

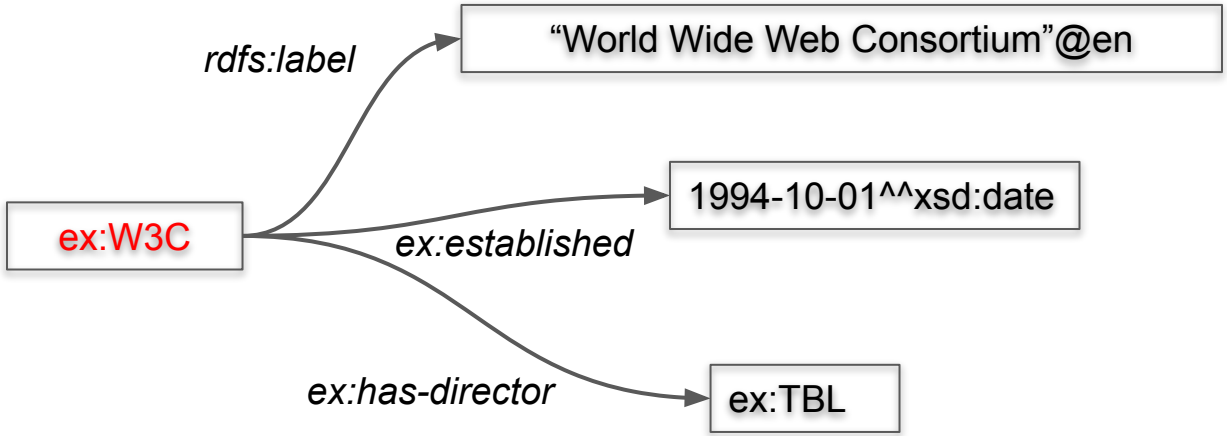


A set of connected triples describes a resource

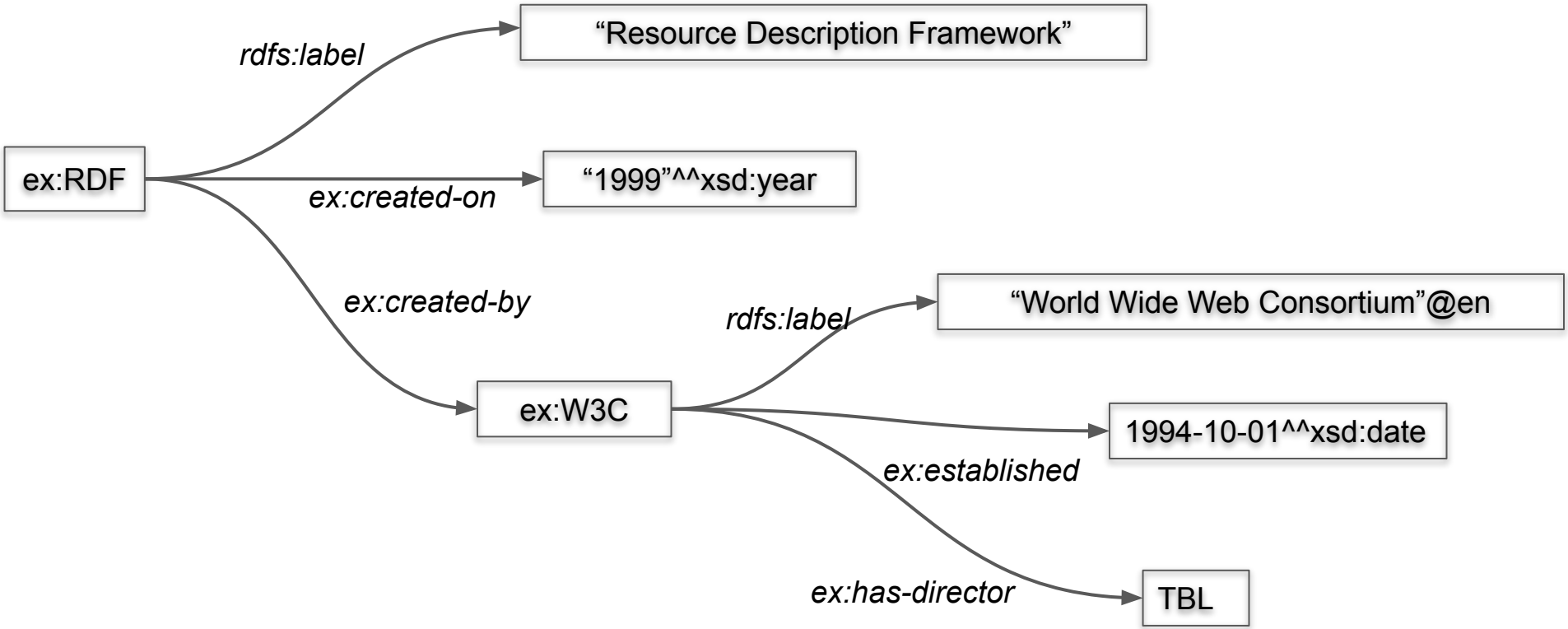


The semantics (or meaning) comes not from the IRI used, but from the description of the resource

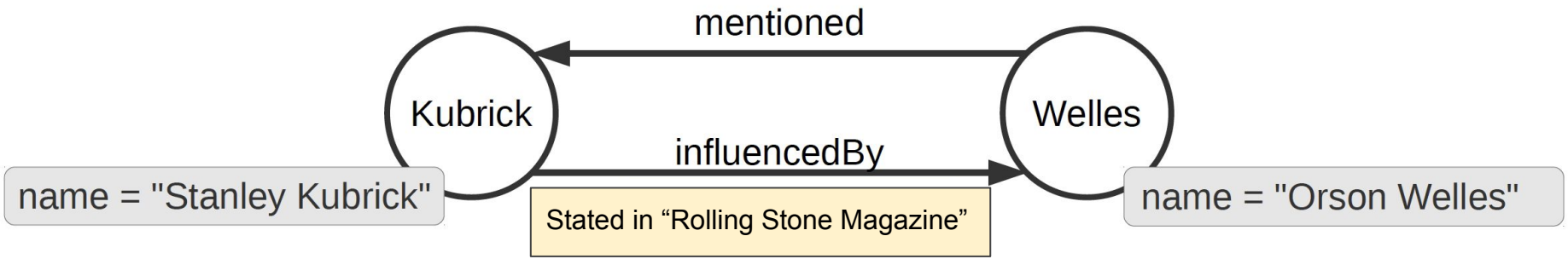
What if another RDF description refers to ex:W3C?



We can connect them and a graph emerges where the node IRIs are shared.



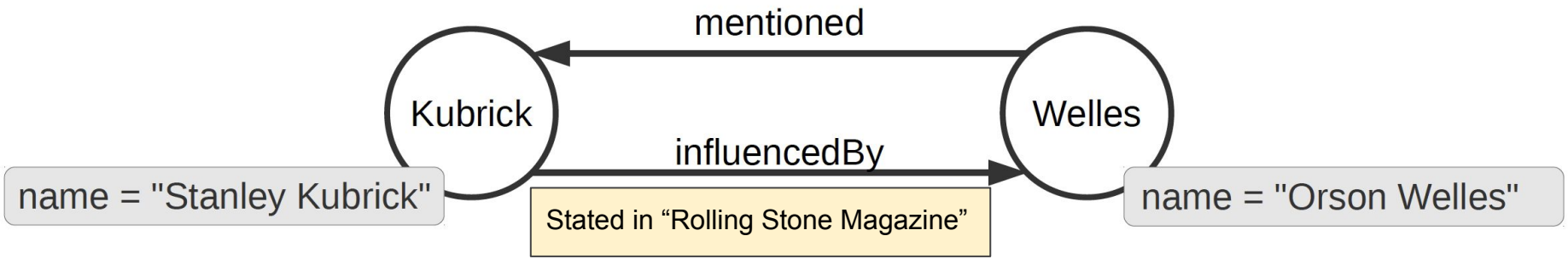
RDF reification (example)



Can you represent the information in this picture with RDF triples?

Adapted from Olaf Hartig (WSP2018)

RDF reification (example)



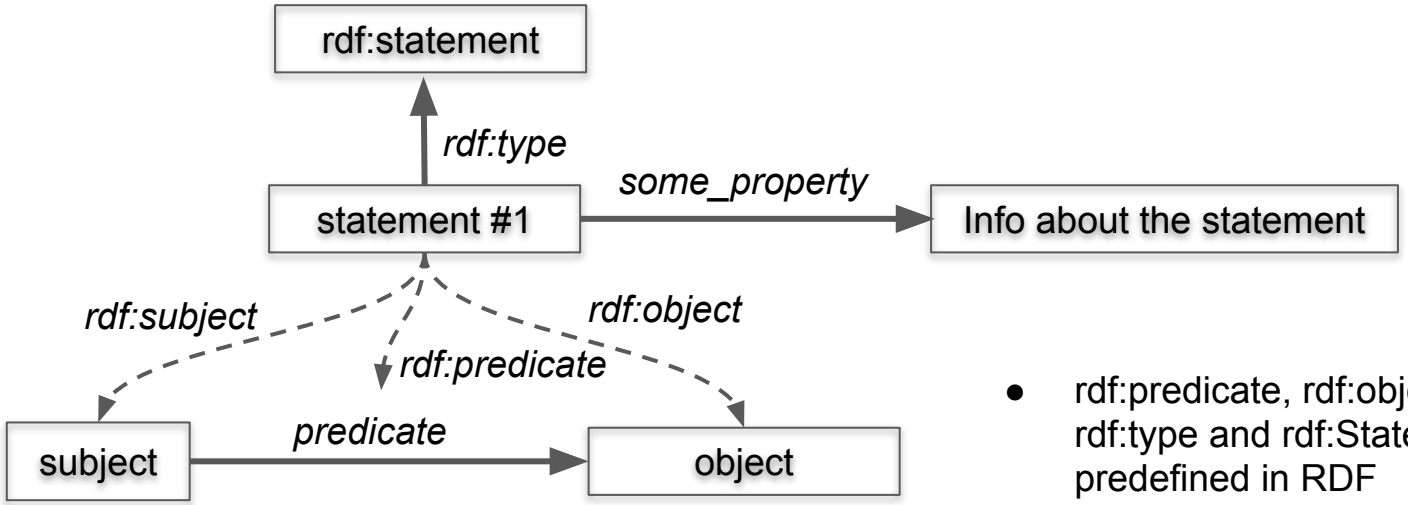
- RDF triples:

```
Welles   name         "Orson Welles" .
Welles   mentioned    Kubrick .
Kubrick  name      "Stanley Kubrick" .
Kubrick  influencedBy  Welles .
```

??? statedIn “Rolling Stone Magazine” .

Adapted from Olaf Hartig (WSP2018)

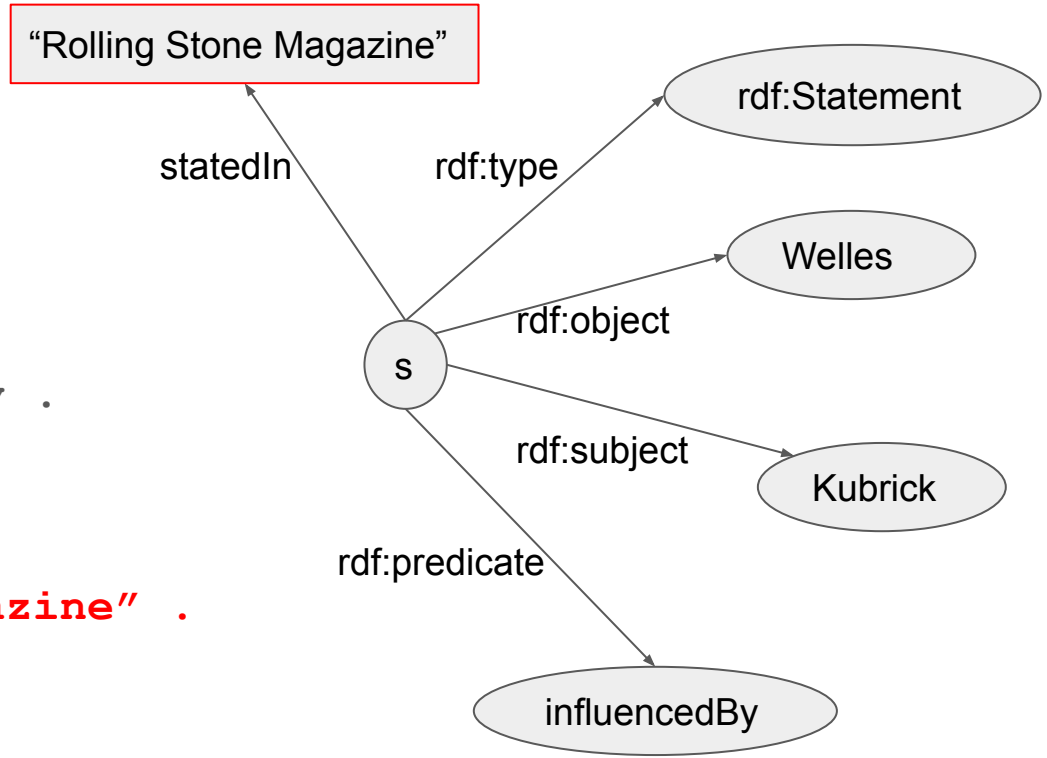
Reification: making statements about statements



- rdf:predicate, rdf:object, rdf:subject, rdf:type and rdf:Statement are all predefined in RDF
- How many triples in total are there here?

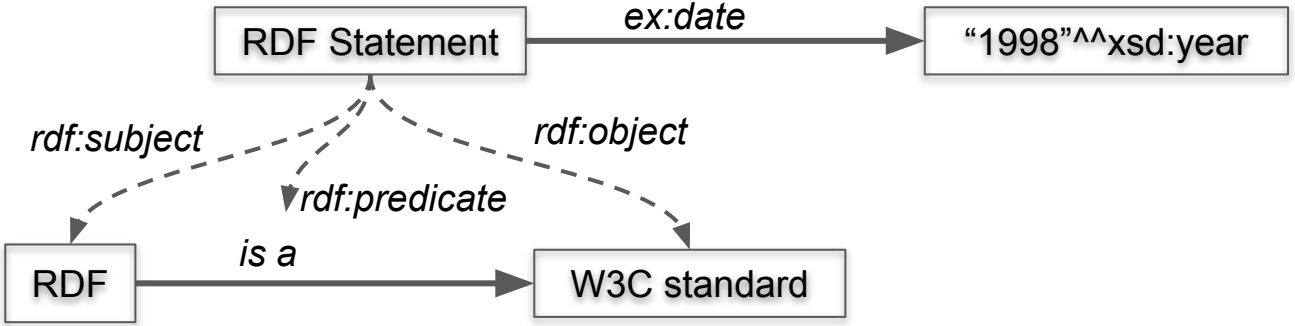
RDF reification (example solution)

```
s rdf:type rdf:Statement .

s rdf:subject Kubrick .

s rdf:predicate influencedBy .

s rdf:object Welles .

s statedIn "RollingStoneMagazine" .
```



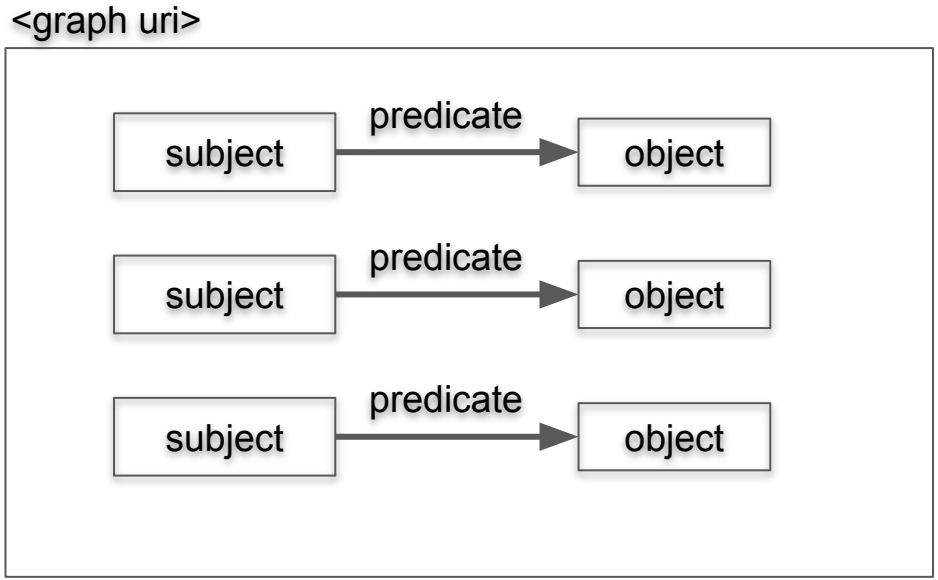
Reification: Making statements about statements

- Reification is a way to obtain an identifier for a RDF triple, and to use it as the subject or object of another triple.
- Reification enables you to represent
  - metadata: date created, creator, license, etc.
  - modal operators: believes, is supported, etc.



RDF Graphs

An **RDF Graph** is a **labeled, directed multigraph**. An **RDF Named Graph** is identified by an URI. The resources in this graph will share this URI as their common namespace



- N-Quads
- TriG
- JSON-LD

RDF Graph Syntaxes: N-Quads

N-Quads

```
<http://maastrichtuniversity.nl>
    <http://www.w3.org/2000/01/rdf-schema#label>
        "Maastricht University"
            <http://example.org/mygraph> .

<http://maastrichtuniversity.nl>
    <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
        <http://example.org/University>
            <http://example.org/mygraph> .
```

RDF Graph Syntaxes: TriG

TriG

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

:mygraph {
  <http://maastrichtuniversity.nl>
     rdfs:label "Maastricht University" ;
     a :University .
}
```

RDF Graph Syntaxes: JSON-LD

see
https://json-ld.org/playground/

JSON-LD

```
{
  "@context": {
    "ex": "http://example.org/",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#"
  },
  "@id": "ex:mygraph",
  "@graph": [
  {
    "@id": "http://maastrichtuniversity.nl",
    "@type": "ex:University",
    "rdfs:label": "Maastricht University"
  }
 ]
}
```

Poll point 3