

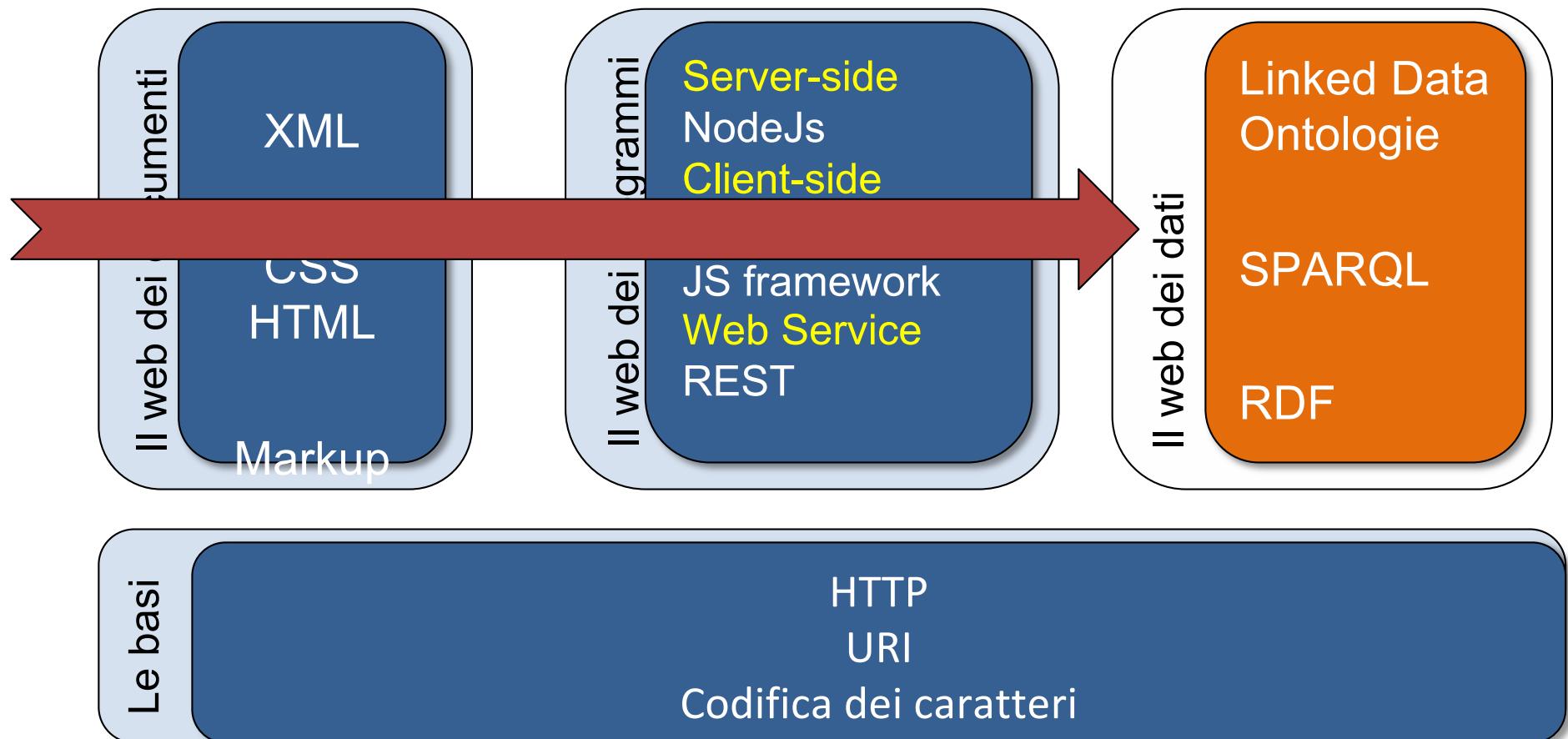
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web + Linked Open Data + RDF & JSON-LD

**Fabio Vitali e
Francesco Sovrano**

Corsi di laurea in Informatica
Alma Mater – Università di Bologna

Argomenti delle lezioni



Introduzione

Oggi parleremo di:

- **Semantic Web:** un'estensione del World Wide Web
- **Linked Data:** dati strutturati interconnessi con altri dati
- **Resource Description Framework:** utilizzato come metodo generale per la descrizione concettuale o la modellazione di informazioni
- **JSON-LD:** un metodo per codificare i Linked Data usando JSON



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

Semantic Web

Secondo il **W3C**: "*Il Web Semantico fornisce una struttura comune che consente di condividere e riutilizzare i dati tra applicazioni, aziende e comunità*".

Semantic Web

Il Semantic Web (SW) riguarda:

separare le **informazioni** dalla rappresentazione, per rendere le informazioni **facilmente navigabili** e comprensibili anche dalle macchine



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

citation links relations between documents document types (venue) document types (content)
 agent's roles document organisation discipline clusters social connections



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

Berners-Lee originariamente espresse la seguente sua visione del Web semantico:

I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A "Semantic Web", which makes this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The "[intelligent agents](#)" people have touted for ages will finally materialize.



Semantic Web

Ma, ancora, come possiamo passare dal vecchio web a quello semantico?

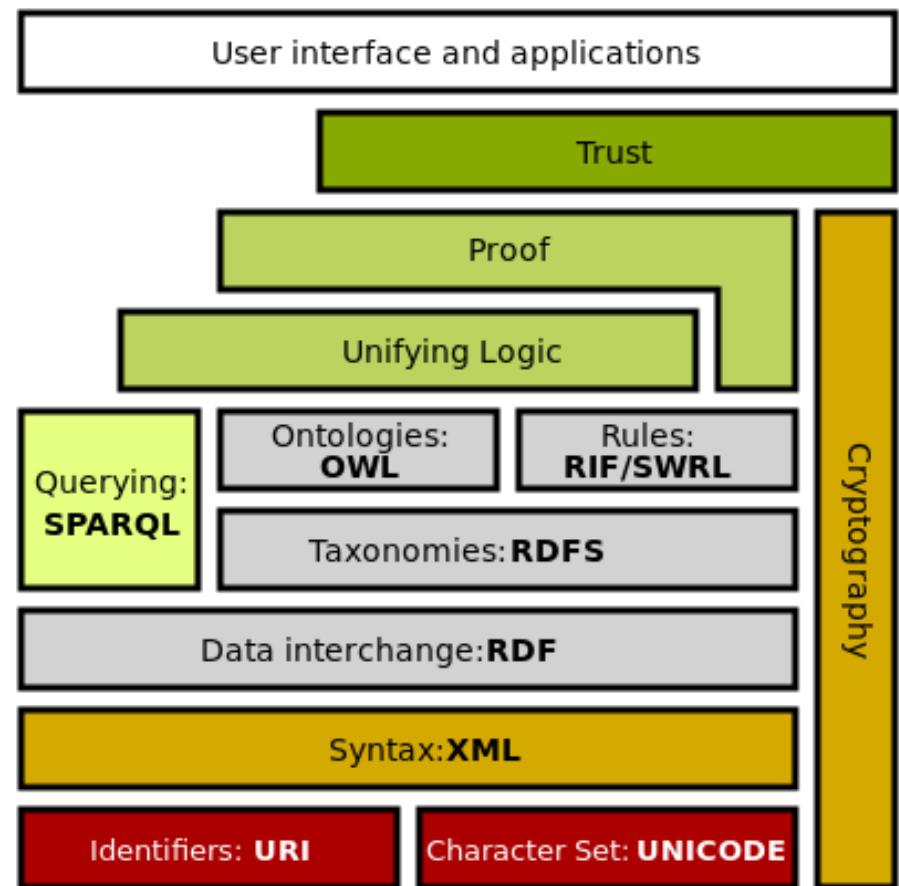
Abbiamo bisogno di tecnologie che forniscano una **descrizione formale di concetti, termini e relazioni** all'interno di un determinato **dominio di conoscenza**.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

Il Semantic Web Stack illustra l'architettura del Web Semantico.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

Lo stack semantico ha diversi livelli:

1. Al livello base ci sono le **risorse**, identificate dagli URI, e probabilmente scritte in un linguaggio naturale (ad esempio inglese, italiano, cinese, ecc..).
2. I **linguaggi di markup**, per la creazione di documenti composti da dati strutturati (eg. XML).
3. Gli **standard per lo scambio di informazioni** (eg. RDF), al fine di permettere di rappresentare le informazioni sotto forma di grafo.
4. Gli **standard per la creazione di tassonomie** (eg. RDFS), così da poter rappresentare proprietà e categorie.

Semantic Web

5. Inoltre comprendere la semantica implica anche trovare concetti, avere **standard** e linguaggi per costruire **ontologie** (eg. OWL) ed avere anche standard per definire le **regole** (eg. RIF / SWRL) che governano la semantica dei dati e che siano comprensibili agli agenti del web, e questo implica la necessità di strumenti / **linguaggi per estrarre informazioni dai dati** (eg. SPARQL) così strutturati.
6. Su queste regole e ontologie potremmo avere programmi e umani che tentano di **eseguire operazioni logiche** per la **verifica** dei fatti, l'inferenza, ecc..

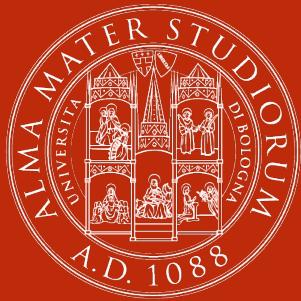


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Semantic Web

Abbiamo già parlato di molte tecnologie, un breve **riassunto**:

- **RDF**: un modello per definire link etichettati tra risorse (dati RDF)
- **OWL**: un linguaggio per la creazione di ontologie, caratterizzate da **un vocabolario di concetti e relazioni tra questi**, che possono essere usate anche per validare dati RDF. Il prodotto del matrimonio infelice tra i sostenitori di RDF e i ricercatori di Logica della Descrizione.
- **SPARQL**: un linguaggio per interrogare dataset di dati RDF (tipicamente chiamati triplestore)



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Linked Data

Linked Data

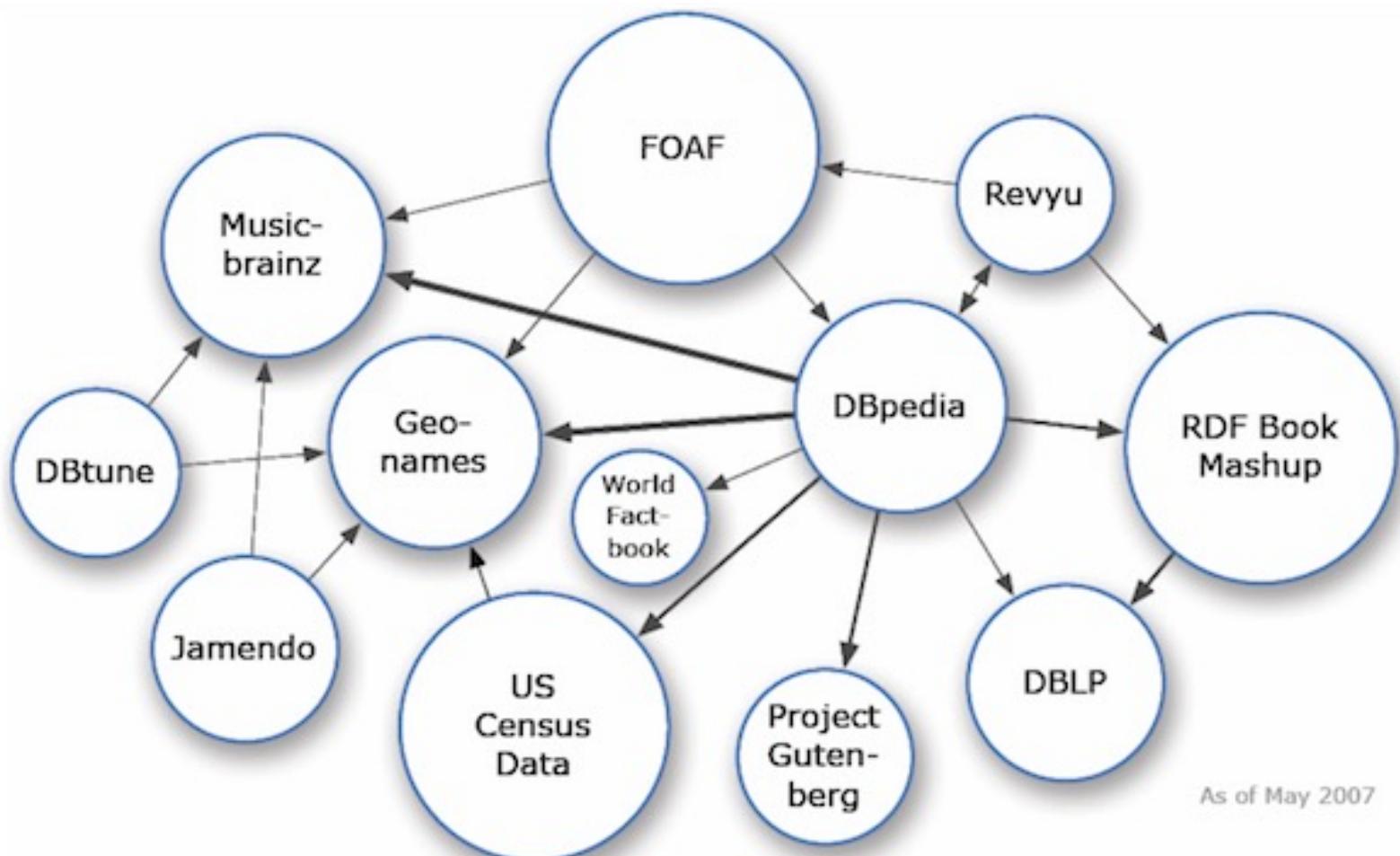
I Linked Data sono **dati strutturati** interconnessi con altri dati.

Linked Data è un'estensione di tecnologie Web standard (come HTTP, RDF e URI) pensata per condividere le **informazioni** in modo che possano essere interpretabili correttamente da **agenti automatici**.

Linked Data è pensato per essere il **database RDF** del Semantic Web. I Linked Data **accessibili ed interrogabili pubblicamente** sono detti **Linked Open Data (LOD)**.

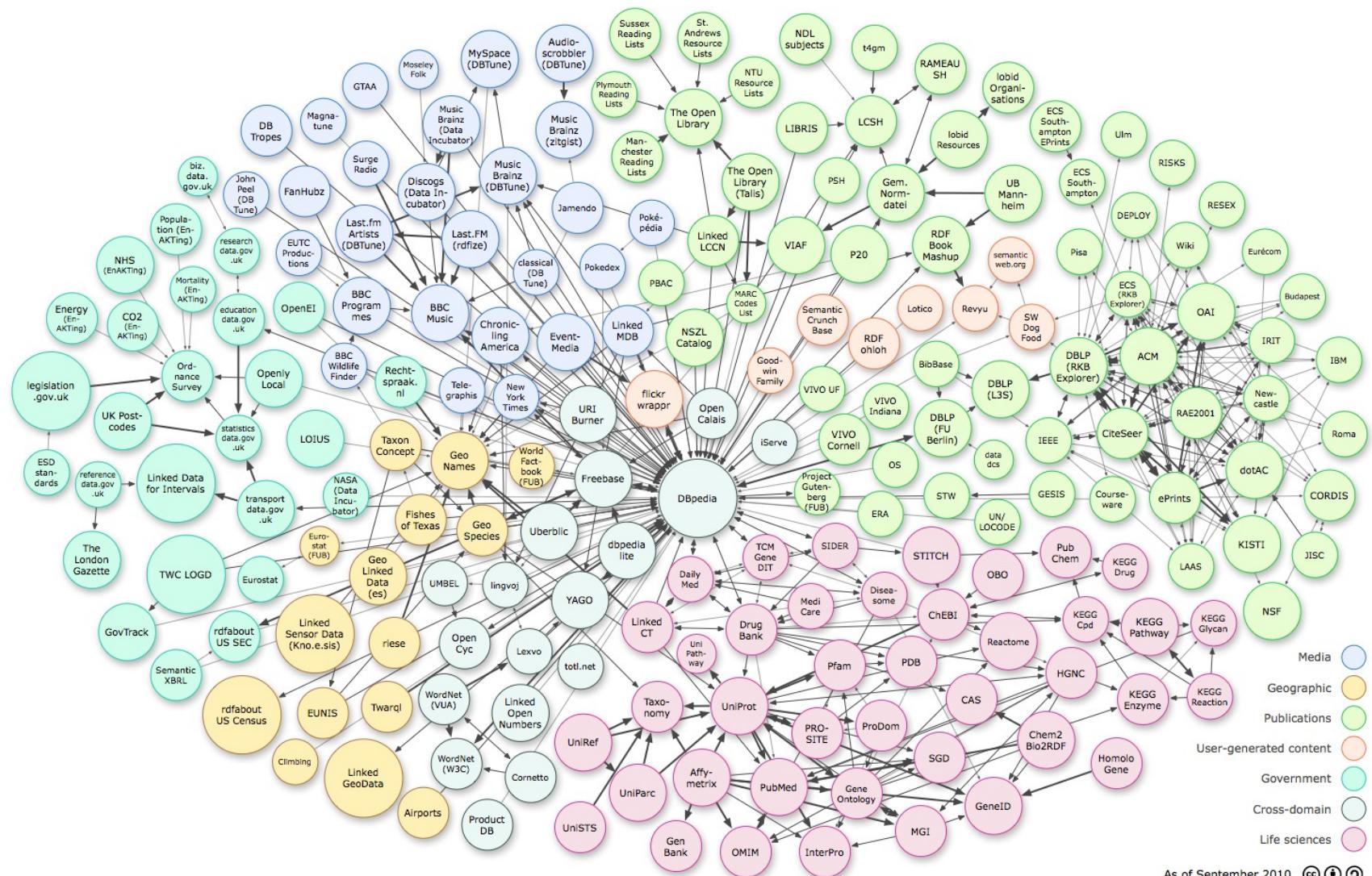


Linked Data - 2007



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Linked Data - 2010



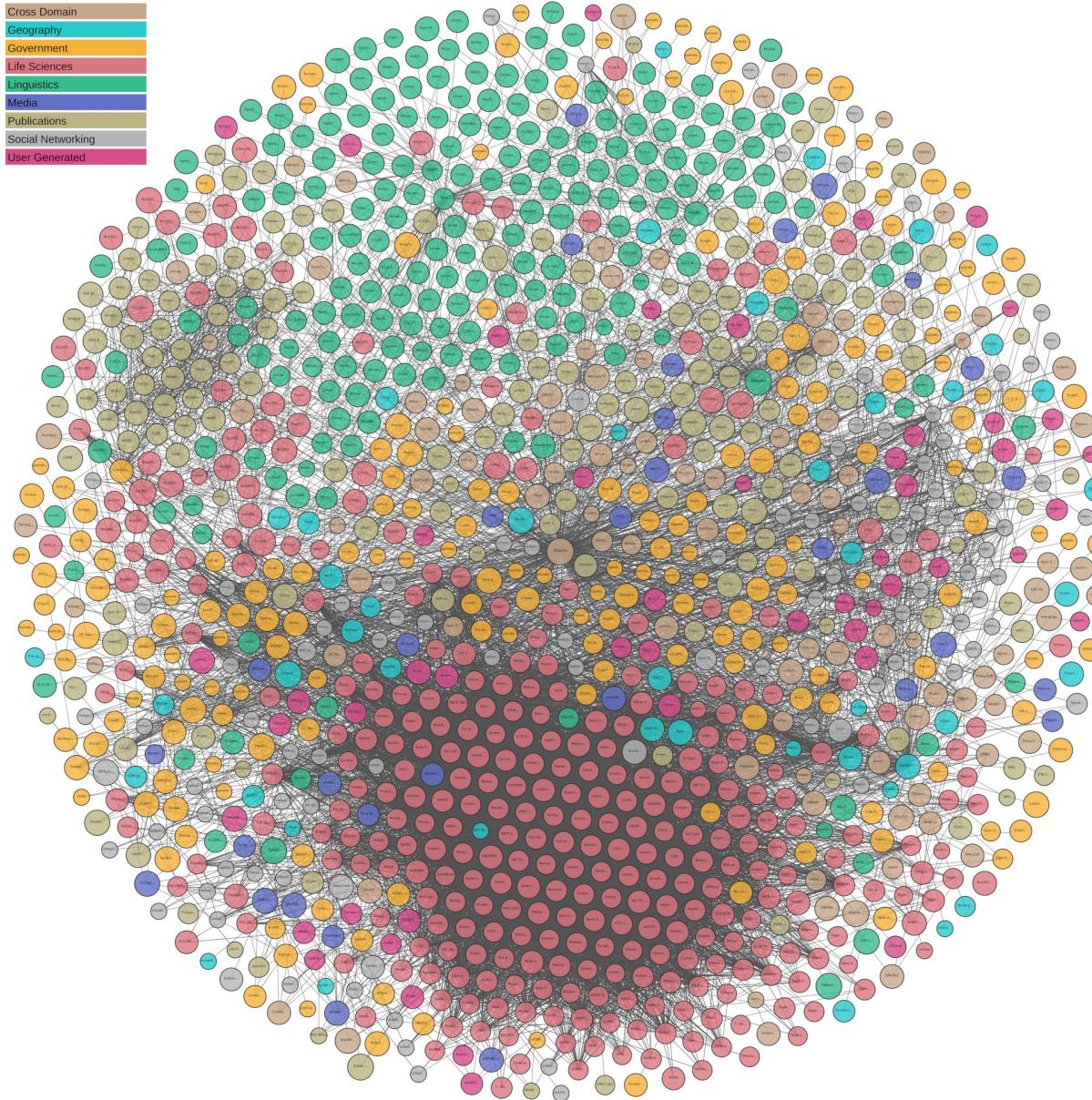
As of September 2010



Linked Data - 2019

Legend

Cross Domain
Geography
Government
Life Sciences
Linguistics
Media
Publications
Social Networking
User Generated



The Linked Open Data Cloud from lod-cloud.net

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

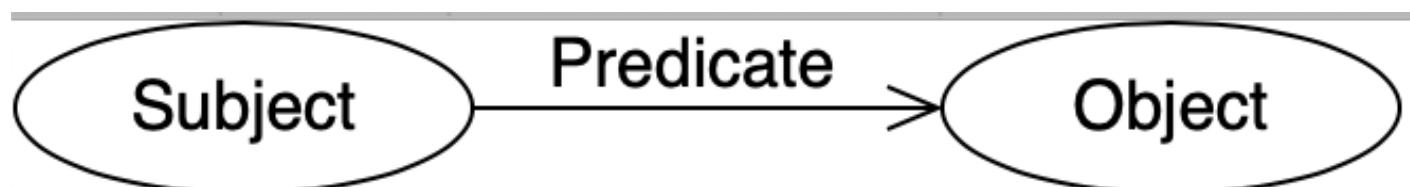
Resource Description Framework

RDF

RDF si basa sull'idea di fare **affermazioni (statement)** sulle risorse (in particolare le risorse web) nella forma di **triple** soggetto-predicato-oggetto, in cui:

- **il Soggetto** è una risorsa (ha un URI)
- **l'Oggetto** è una risorsa (ha un URI) o un letterale (una stringa, un numero, una data, ecc.)
- Il **Predicato** è una **relazione** tra risorse, una **proprietà**.

Capiamolo con un esempio...



RDF

Esempio: “Umberto Eco è autore de Il Nome della Rosa” può essere espresso **informalmente in triple RDF** (**soggetto-predicato-oggetto**):

Esiste un'entità E

identificato dall'URI <https://dbpedia.org/umberto_eco>
può essere qualunque altro indirizzo purché univoco.

E è una autore

(“scrittore” è il suo tipo).

E si chiama “Umberto Eco”.

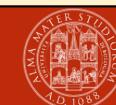
E ha scritto B

identificato da <http://www.anobii.com/books/Il_nome_della_rosa>

B è un libro

qui il tipo è "libro"

B si chiama “Il Nome della Rosa”



RDF

Esempio: “*Umberto Eco è autore de Il Nome della Rosa*”:

- Abbiamo 2 diversi **soggetti**:
 - E, B
- Abbiamo 3 diversi **predicati**:
 - è un/una, si chiama, ha scritto
- Abbiamo 5 diversi **oggetti**:
 - autore, “*Umberto Eco*”, B, libro, “*Il Nome della Rosa*”

Per un totale di **5 triple** RDF.

RDF

Esempio: “*Umberto Eco è autore de Il Nome della Rosa*”:

- Dall'esempio possiamo notare che tutti i **soggetti** sono URI
- Tutti i **predicati** sono verbi. In base allo standard RDF dobbiamo associare a questi verbi un URI univoco.
- Gli **oggetti** possono essere URI o letterali. Alcuni letterali sono titoli o nomi propri, ma altre sono tipi di oggetti, date, numeri, ecc.. Possiamo associare a questi tipi un URI, sempre grazie ad un'ontologia!



RDF

- Ricapitolando -

I **soggetti** sono un URI indicanti la risorsa di cui stiamo fornendo una descrizione.

- Ricapitolando -

I **predicati** sono un URI indicante una risorsa che rappresenti una relazione.

- Ricapitolando -

Gli **oggetti** sono URI (e allora si dirà che il predicato è una object property), oppure **stringhe letterali** Unicode (e allora si dirà che il predicato è una datatype property).



RDF - le risorse anonime

In realtà, i **soggetti** e gli **oggetti** di uno statement RDF possono essere:

- un identificatore di risorsa uniforme (**URI**), oppure
- una **risorsa anonima** (detta anche **nodo vuoto** o **blank node**).

Le risorse anonime sono risorse **variabili (esistenziali)**, non costanti. Vedremo degli esempi.

Un'analogia per gli informatici: *intuitivamente, pensate alle costanti (viste studiando il C) e alla differenza che c'è con le variabili. Una variabile può assumere diversi valori a seconda delle circostanze, la costante no.*

RDF - i grafi

Un **grafo RDF** è un insieme di triple RDF.

Si può rappresentare un grafo in molti modi, ad esempio attraverso una rete semantica tale che:

- Le **risorse** (**soggetti** e **oggetti**) sono rappresentate come nodi.
- I **predicati** sono rappresentati come archi.

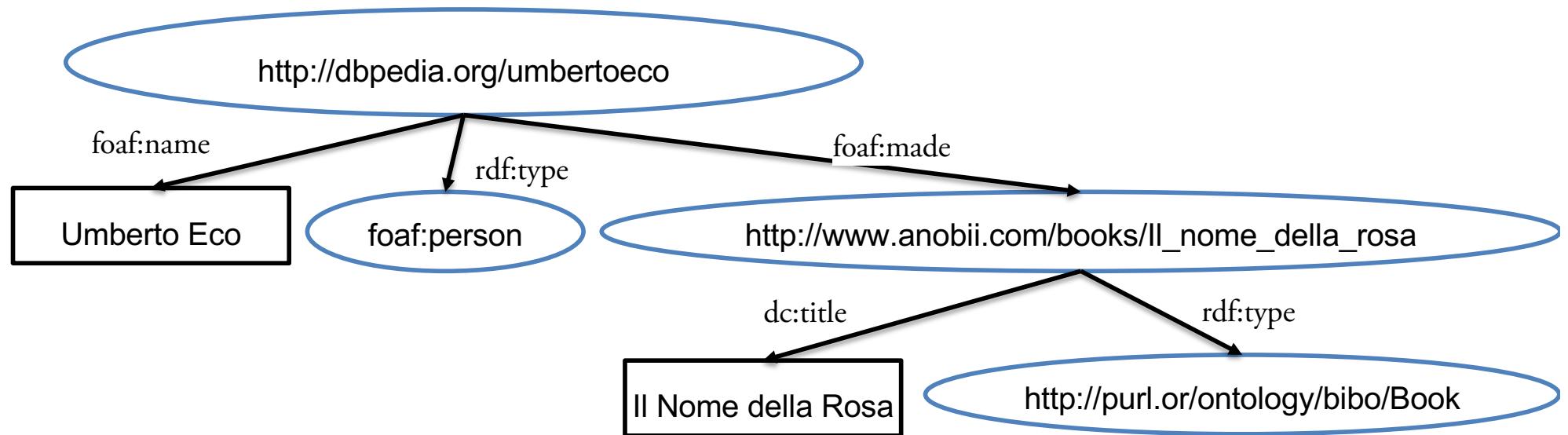
Questa particolare rappresentazione ha il vantaggio di essere facilmente leggibile dagli umani.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - i grafi

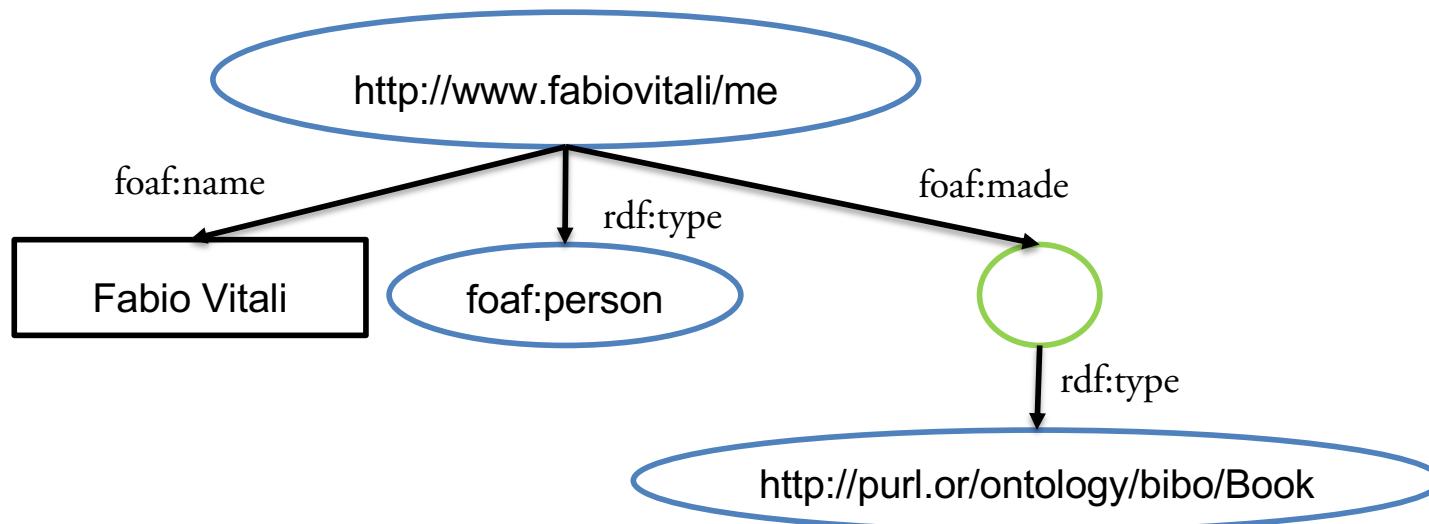
Esempio: “Umberto Eco è autore de Il Nome della Rosa”:



Attenzione, stiamo usando diverse **ontologie** per gli URI di **tutti i predicati e soggetti**, e degli **oggetti** che non siano stringhe di letterali (“Umberto Eco” e “Il nome della rosa”).

RDF - i grafi

Esempio 2: “Anche Fabio Vitali è autore di un libro”:



Ecco un esempio di **risorsa anonima**: la risorsa indicata dal cerchio verde.

La uso in questa serie di annotazioni ma non le associo un identificatore (un URI) perché serve solo qui.



RDF - il vocabolario base

Abbiamo visto come **modellare** in RDF, frasi in linguaggio naturale (italiano).

Abbiamo visto che RDF si basa fortemente sull'utilizzo di **URI**, anche definiti da ontologie.

Più formalmente:

- Esiste l'ontologia RDF che definisce il **vocabolario di base** di RDF (es. **rdf:type**).
- L'ontologia RDF definisce **regole** da rispettare per la creazione di triple RDF.
- L'ontologia RDFS **estende** RDF definendo anche le regole per creare ontologie compatibili con RDF.



RDF - Formati di Serializzazione

Per RDF sono in uso diversi formati di serializzazione comuni, tra cui citiamo:

- [Turtle](#): un formato compatto e human-friendly.
- [JSON-LD](#): una serializzazione basata su JSON.
- [RDF/XML](#): una sintassi basata su XML; il primo formato standard per la serializzazione di RDF.
- eccetera...



RDF

Il formato RDF/XML a volte viene chiamato RDF, in modo fuorviante. Questo perché RDF/XML è stato introdotto con le altre specifiche W3C che definiscono RDF ed è stato storicamente il primo formato standard W3C di serializzazione RDF.

Tuttavia, è importante **distinguere** il formato (di serializzazione) **RDF/XML** dal modello astratto **RDF** (le triple).





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Limiti e Vantaggi

RDF - Vantaggi

- Il modello a triple è molto **semplice** e minimalista.
- La struttura dati risultante dal modello RDF è praticamente un **grafo**, con gli stessi vantaggi e svantaggi.
- Il modello RDF ha l'importante proprietà di essere **modulare**. Ciò significa che:
 - L'elaborazione delle informazioni può essere completamente **parallelizzata**.
 - In presenza di **informazioni parziali** (una caratteristica essenziale in un ambiente volatile come il web) l'output è ancora un modello RDF coerente, che può essere elaborato con successo.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Vantaggi

- RDF combinato con OWL permette ad Intelligenze Artificiali di ragionare sui dati RDF e di estrarre informazioni **efficientemente**. Perchè?
 - OWL è costruito sulla teoria formale delle **Logica della Descrizione** (Description Logic - DL).
 - In DL esistono solo relazioni binarie o unarie, questo rende efficiente il **ragionamento**, limitandolo.
 - Un esempio di applicazione notevole di DL e OWL è nell'informatica biomedica in cui DL aiuta nella codificazione delle conoscenze biomediche.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Svantaggi

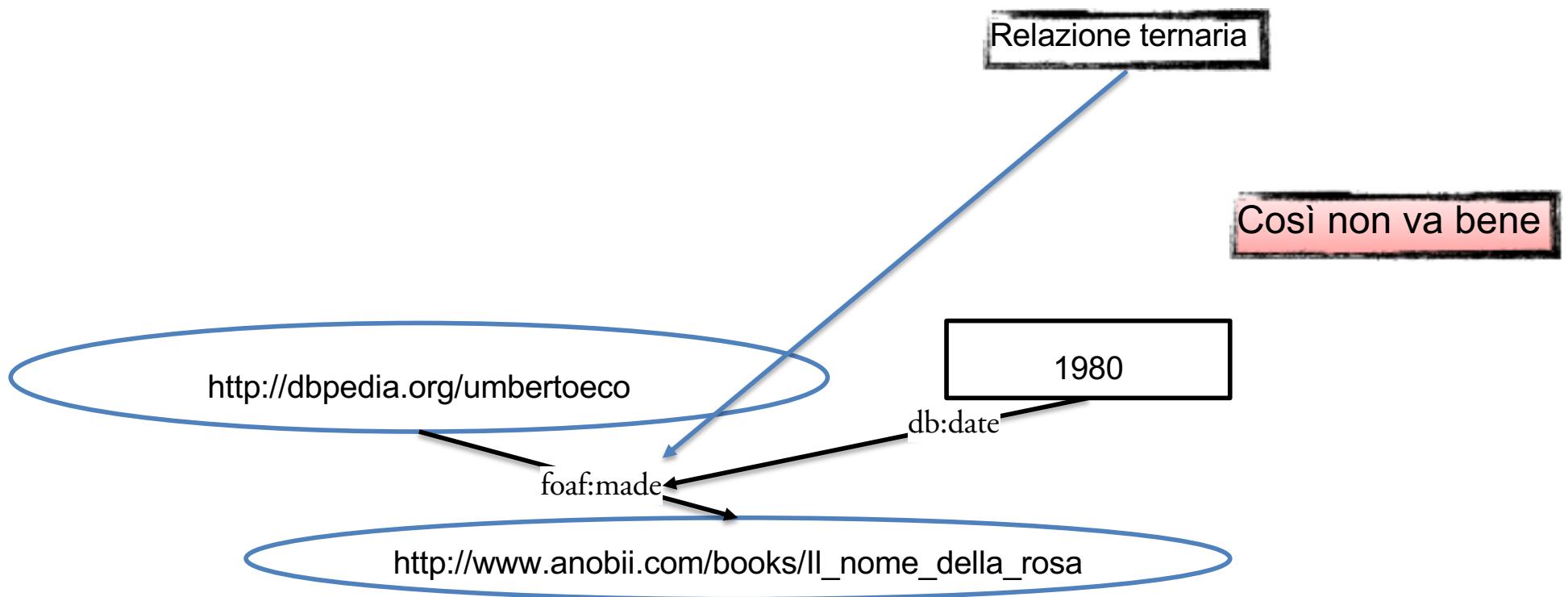
- Il modello di dati RDF è costituito da elementi di **dati** molto piccoli e **frammentati**. Di conseguenza, un database relazionale di dimensioni medie può corrispondere a un triplestore contenente miliardi di triple.
- Una seconda limitazione è la limitazione delle **relazioni N-arie**. Il modello RDF non consente modi semplici per descrivere relazioni N-arie tra i vertici del grafo semantico, il che complica la descrizione di situazioni complesse. Questa limitazione è **ereditata dalla DL**.
- Una terza limitazione è la possibilità di attribuire informazioni alle triple (e.g., provenienza, freschezza, ecc.). Una limitata risposta a questo si chiama **reificazione**. Un'altra si chiama **Named Graphs**.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

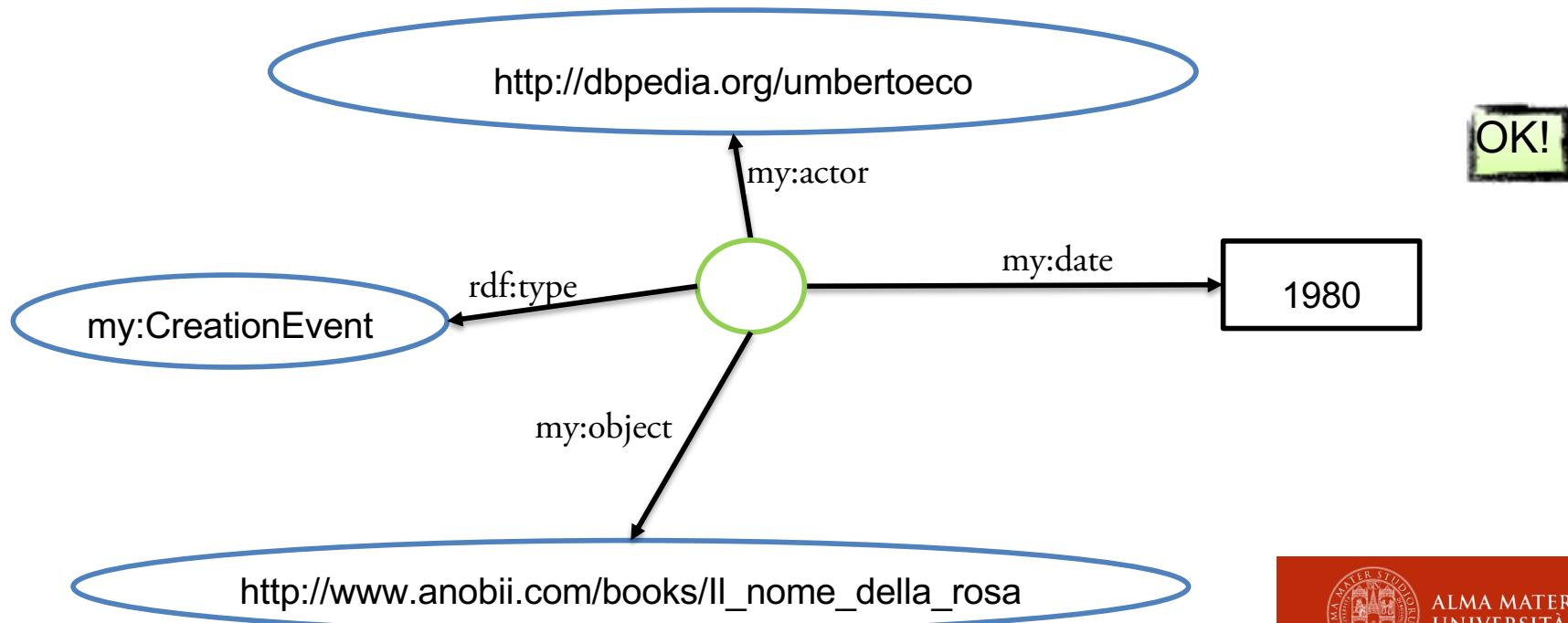
Relazioni n-arie

- **Esempio:** Umberto Eco ha scritto "Il nome della rosa" nell'anno 1980.



Relazioni n-arie (2)

- **Esempio:** Umberto Eco ha scritto "Il nome della rosa" nell'anno 1980.

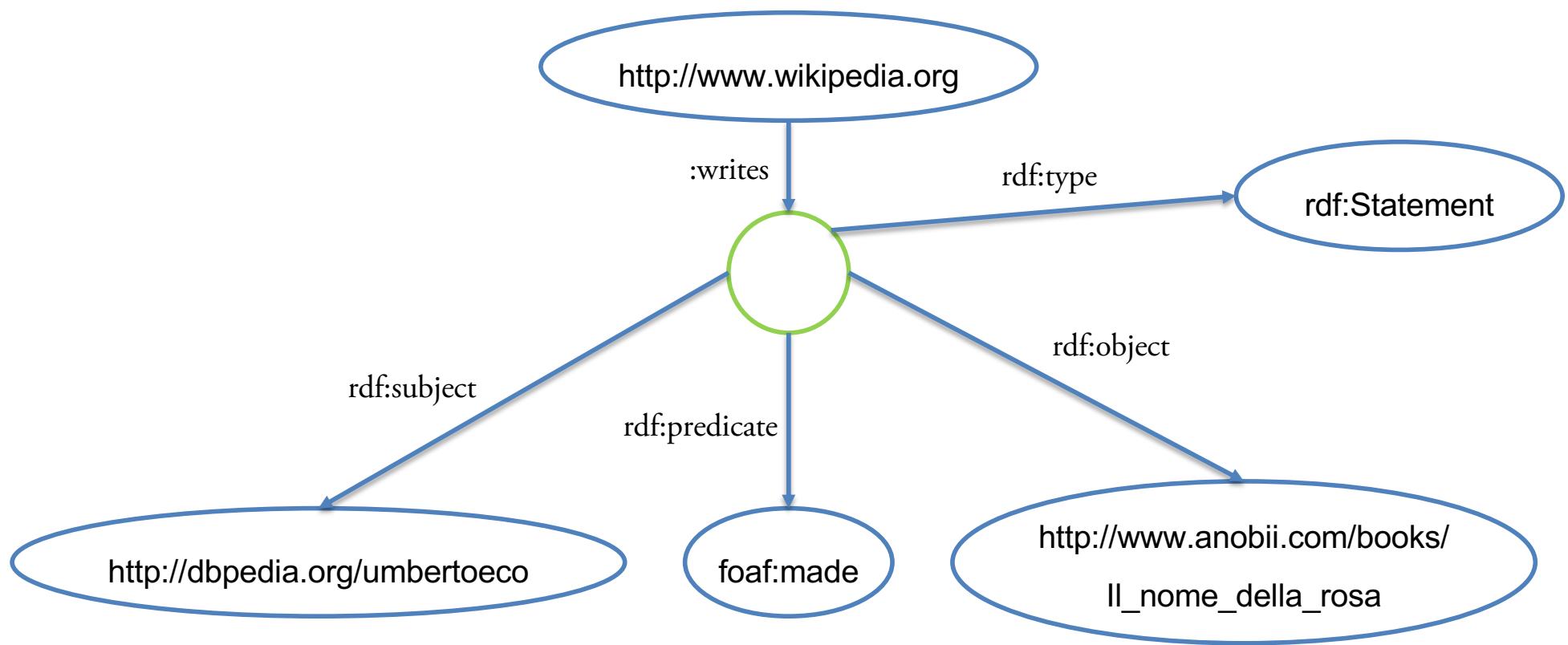


Reificazione

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".
- Qui non c'è una relazione n-aria da rendere con un blank node. Questa va considerata come composta di due frasi separate (due triple):
 - Wikipedia dice X
 - Umberto Eco ha scritto "Il nome della rosa" (*questa è X*)
- Per rendere possibile questo bisogna rappresentare la seconda frase come un'entità autonoma (*reificazione*) e poi usare l'id di questa per la prima frase.

Reificazione

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".



Reificazione

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".
- La frase interna viene spezzata nei tre elementi costitutivi: soggetto, predicato e oggetto, e lo statement viene generato (reificato) dalla tripla di base.
- Si usano classi e predicati predefiniti in RDF: rdf:Statement, rdf:subject, rdf:predicate, rdf:object.
- La frase esterna usa lo statement reificato come oggetto.
- La tripla di base non c'è! (lo statement non è asserito). Se cerco "Chi ha scritto il nome della rosa?" non lo trovo



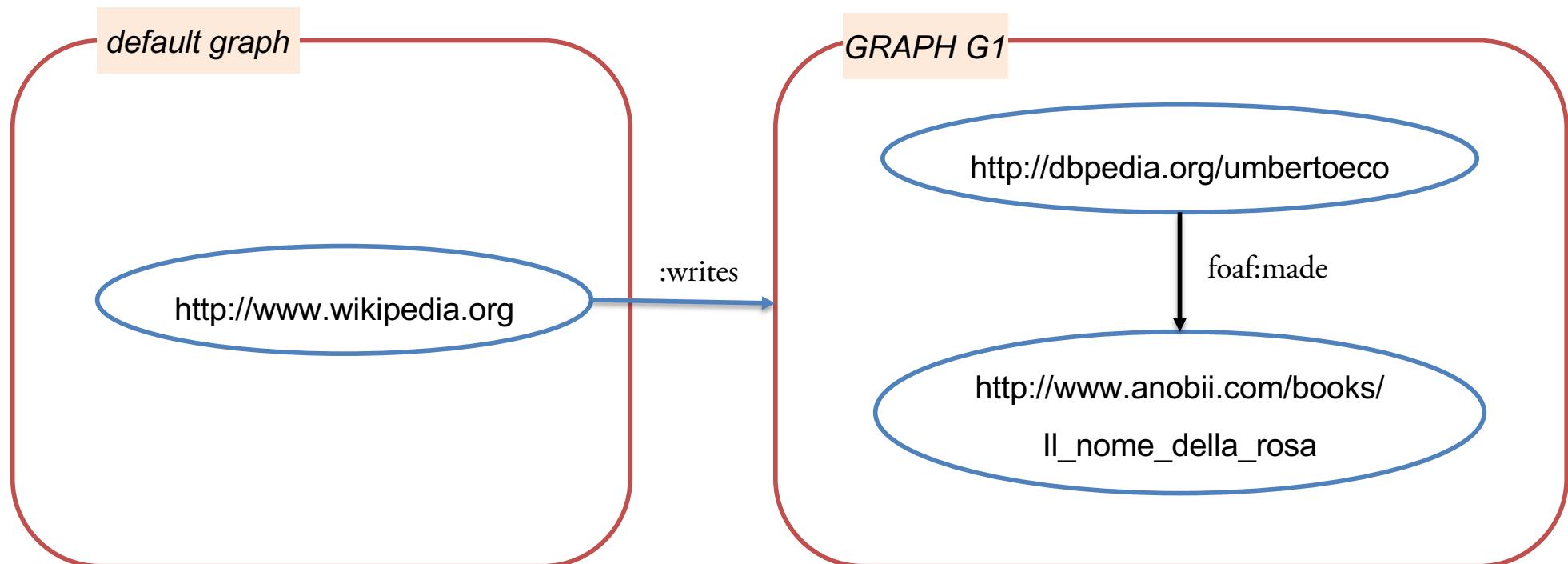
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Named graph

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".
- Sopra alla singola tripla esiste il grafo. Un dataset RDF è un grafo di triple collegate tra di loro.
- Ma... e se nello stesso dataset ci fossero più grafi? Magari se diamo un nome al grafo riusciamo a fare cose interessanti.

Named graph

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".



Named graph

- **Esempio:** Wikipedia dice che Umberto Eco ha scritto "Il nome della rosa".
- La frase interna viene inserita in un grafo separato come una normalissima tripla. Il grafo separato ha un'identità.
- La frase esterna punta al grafo separato.
- La tripla di base c'è o non c'è? Lo standard RDF 1.1 non è chiaro sul fatto. Se cerco "Chi ha scritto il nome della rosa?" lo trovo su alcuni motori ma non tutti.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Serializzazioni

RDF - Esempi

Esempio 1: “*Umberto Eco è autore de Il Nome della Rosa*”

Esempio 2: “*Fabio Vitali è autore di un libro*”

Esempio 3: “*Wikipedia scrive che Umberto Eco è autore de Il Nome della Rosa*”

Come possiamo scriverli in **Turtle**, **RDF/XML** o **JSON-LD**?

RDF - Turtle

Esempio 1: “Umberto Eco è autore de Il Nome della Rosa”

```
@prefix anobii: <http://www.anobii.com/books/>.
```

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
```

I prefissi abbreviano gli URI.

```
@prefix dc: <http://purl.org/dc/terms/>.
```

Gli URI completi debbono essere inseriti tra < e >

```
@prefix db: <https://dbpedia.org/>.
```

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

```
db:umbertoecho
```

Il punto e virgola aggrega triple con lo stesso soggetto

```
rdf:type foaf:person;  
foaf:name "Umberto Eco";  
foaf:made anobii:Il_nome_della_rosa.
```

```
anobii:Il_nome_della_rosa
```

```
a <http://purl.org/ontology/bibo/Book>;  
dc:title "Il nome della rosa".
```

Il predicato "a" è un'abbreviazione di rdf:type



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Turtle

Esempio 2: “*Fabio Vitali è autore di un libro*”:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
<http://www.fabiovitali/me>
  foaf:name "Fabio Vitali" ;
  a foaf:person ;
  foaf:made [
    a <http://purl.org/ontology/bibo/Book>
  ].
```

La parentesi quadra introduce un blank node



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Turtle

Esempio 3: “Wikipedia dice che *Umberto Eco* è autore de *Il Nome della Rosa*” (usando la reificazione)

```
@prefix anobii: <http://www.anobii.com/books/>.  
@prefix foaf: <http://xmlns.com/foaf/0.1/>.  
@prefix dc: <http://purl.org/dc/terms/>.  
@prefix db: <https://dbpedia.org/>.
```

```
<http://www.wikipedia.org> my:writes _:S1 .
```

un blank node può anche
ottenere un URI fittizio,
che inizia con underscore.

```
_:S1 a rdf:Statement;  
      rdf:subject db:umbertoecho;  
      rdf:predicate foaf:made;  
      rdf:object: anobii:Il_nome_della_rosa .
```



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Turtle

Esempio 3: “Wikipedia dice che *Umberto Eco* è autore de *Il Nome della Rosa*” (usando named graphs)

```
@prefix anobii: <http://www.anobii.com/books/>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix dc: <http://purl.org/dc/terms/>.
@prefix db: <https://dbpedia.org/>.
@prefix my: <http://example.com/>

<http://www.wikipedia.org> my:writes _:G1.

GRAPH _:G1 {
    db:umbertoecho
    rdf:type foaf:person;
    foaf:name "Umberto Eco";
    foaf:made anobii:Il_nome_della_rosa.
}
```

un grafo viene circondato da parentesi graffe

un grafo può contenere molte triple

RDF - RDF/XML

Esempio 1: “Umberto Eco è autore de Il Nome della Rosa”

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:dc="http://purl.org/dc/terms/">
    <foaf:person rdf:about="https://dbpedia.org/umbertoeco_">
        <foaf:name>Umberto Eco</foaf:name>
        <foaf:made>
            <rdf:Description
                rdf:about="http://www.anobii.com/books/Il_nome_della_rosa">
                <dc:title>Il nome della rosa</dc:title>
                <rdf:type
                    rdf:resource="http://purl.org/ontology/bibo/Book"/>
            </rdf:Description>
        </foaf:made>
    </foaf:person>
</rdf:RDF>
```

I predicati sono elementi XML

l'elemento *rdf:Description* racchiude
una o più triple con lo stesso soggetto

l'attributo *rdf:about*
identifica un soggetto

l'attributo *rdf:resource*
identifica un oggetto



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - RDF/XML

Esempio 2: “*Fabio Vitali è autore di un libro*”

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/">
    <foaf:person rdf:about="http://www.fabiovitali/me">
        <foaf:name>Fabio Vitali</foaf:name>
        <foaf:made>
            <rdf:Description>
                <rdf:type
                    rdf:resource="http://purl.org/ontology/bibo/Book"/>
                </rdf:Description>
            </foaf:made>
        </foaf:person>
    </rdf:RDF>
```

un *rdf:Description* senza *rdf:about*
crea un blank node



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - RDF/XML

Esempio 3: “Wikipedia dice che *Umberto Eco è autore de Il Nome della Rosa*” (usando la reificazione)

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:my="http://example.com/">

<rdf:Description rdf:about="http://www.wikipedia.org/">
  <my:writes>
    <rdf:Description>
      <rdf:subject rdf:resource="https://dbpedia.org/umbertoeeco"/>
      <rdf:predicate rdf:resource="http://xmlns.com/foaf/0.1/made"/>
      <rdf:object rdf:resource="http://www.anobii.com/books/Il_nome_della_rosa"/>
      <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
    </rdf:Description>
  </my:writes>
</rdf:Description>
</rdf:RDF>
```

gli URI usati come elementi o attributi possono usare i prefissi

gli URI usati come valori negli attributi debbono essere completi

RDF - RDF/XML

Esempio 3: “Wikipedia dice che *Umberto Eco* è autore de *Il Nome della Rosa*” (usando named graph)

```
<rdf:RDF  
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
    xmlns:my="http://example.com/">  
  
<rdf:Description rdf:about="http://www.wikipedia.org/">  
    <my:writes>  
        NON E' POSSIBILE ESPRIMERE  
        NAMED GRAPHS IN RDF/XML  
    </my:writes>  
</rdf:Description>  
</rdf:RDF>
```

gli URI usati come elementi o attributi possono usare i prefissi

gli URI usati come valori negli attributi debbono essere completi



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

JSON-LD

JSON-LD (**JavaScript Object Notation for Linked Data**), è un metodo di codifica di Linked Data che utilizza JSON.

JSON-LD è progettato attorno al concetto di "**contesto**" per fornire **mappature** aggiuntive da JSON (array associativi **chiave-valore**) a un modello RDF (**triple**). Il contesto collega le proprietà dell'oggetto in un documento JSON a concetti in un'ontologia.

JSON-LD non è strutturalmente diverso da nessun altro documento **JSON**.

RDF – JSON-LD

Esempio 1: “Umberto Eco è autore de Il Nome della Rosa”

```
[  
  {  
    "@id": "https://dbpedia.org/umbertoecho",  
    "@type": "http://xmlns.com/foaf/0.1/person" ,  
    "http://xmlns.com/foaf/0.1/name": "Umberto Eco",  
    "http://xmlns.com/foaf/0.1/made": {  
      "@id": "http://www.anobii.com/books/Il_nome_della_rosa",  
      "@type": "http://purl.org/ontology/bibo/Book" ,  
      "http://purl.org/dc/terms/title": "Il nome della rosa"  
    }  
  }  
]
```

Ogni individuo del dataset è un oggetto in un array del JSON.

I nomi di proprietà che iniziano con @ sono riservati al linguaggio JSON-LD.

@id è l'uri dell'oggetto, @type il rdf:type.

RDF – JSON-LD

Esempio 1: “*Umberto Eco è autore de Il Nome della Rosa*”

```
{  
  "https://dbpedia.org/umbertoecho": {  
    "@type": "http://xmlns.com/foaf/0.1/person" ,  
    "http://xmlns.com/foaf/0.1/name": "Umberto Eco" ,  
    "http://xmlns.com/foaf/0.1/made": {  
      "http://www.anobii.com/books/Il_nome_della_rosa" :{  
        "@type": "http://purl.org/ontology/bibo/Book" ,  
        "http://purl.org/dc/terms/title": "Il nome della rosa"  
      }  
    }  
  }  
}
```

Ogni array di valori può essere sostituito da una lista di oggetti le cui chiavi sono gli @id dell'oggetto di partenza.

RDF – JSON-LD

Esempio 1: “Umberto Eco è autore de Il Nome della Rosa”

```
{  
  "@context": {  
    "a": "@type",  
    "title": "http://purl.org/dc/terms/title",  
    "foaf": "http://xmlns.com/foaf/0.1/",  
    "Umberto Eco": "https://dbpedia.org/umbertoeeco",  
    "Nome della Rosa": "http://www.anobii.com/books/Il_nome_della_rosa",  
    "Person": "http://xmlns.com/foaf/0.1/person",  
    "Book": "http://purl.org/ontology/bibo/Book"  
  },  
  "Umberto Eco": {  
    "a": "Person",  
    "foaf:name": "Umberto Eco",  
    "foaf:made": {  
      "Nome della Rosa": {  
        "a": "Book",  
        "title": "Il nome della rosa"  
      }  
    }  
  }  
}  
}  l'oggetto predefinito @context può essere usato per introdurre  
nomi leggibili, anche al posto di URI o frammenti di URI (prefissi).
```



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF – JSON-LD

Esempio 2: “*Fabio Vitali è autore di un libro*”:

```
{  
  "@context": {  
    "a": "@type",  
    "foaf": "http://xmlns.com/foaf/0.1/",  
    "Fabio Vitali": "https://dbpedia.org/fabiovitali",  
    "Person": "http://xmlns.com/foaf/0.1/person",  
    "Book": "http://purl.org/ontology/bibo/Book"  
  },  
  "Fabio Vitali": {  
    "a": "Person",  
    "foaf:name": "Fabio Vitali",  
    "foaf:made": {  
      "@id": "_:B1",  
      "a": "Book"  
    }  
  }  
}
```

Un blank node non ha @id
oppure ce l'ha appartenente al prefisso predefinito “_”

RDF – JSON-LD

Esempio 3: “Wikipedia dice che *Umberto Eco* è autore de *Il Nome della Rosa*” (usando la reificazione)

```
{  
  "@context": {  
    "foaf": "http://xmlns.com/foaf/0.1/",  
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",  
    "my": "http://example.com/"  
  },  
  "@id": "http://www.wikipedia.org",  
  "my:writes": {  
    "@type": "rdf:Statement",  
    "rdf:subject": {  
      "@id": "https://dbpedia.org/umbertoeeco"  
    },  
    "rdf:predicate": {  
      "@id": "foaf:made"  
    },  
    "rdf:object": {  
      "@id": "http://www.anobii.com/books/Il_nome_della_rosa"  
    }  
  }  
}
```



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF – JSON-LD

Esempio 3: “Wikipedia dice che *Umberto Eco è autore de Il Nome della Rosa*” (usando named graphs)

```
{  
  "@context": {  
    "said": "http://example.com/said",  
    "writes": "http://example.com/writes"  
  },  
  "@id": "http://www.wikipedia.org",  
  "writes": {  
    "@graph": [{  
      "@id": "https://dbpedia.org/umbertoeeco",  
      "foaf:made": "http://www.anobii.com/books/Il_nome_della_rosa"  
    }]  
  }  
}
```

@graph dice che il valore dell'oggetto di "writes" è un intero grafo, che al momento contiene solo una tripla ma può contenerne quante se ne vuole.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDFS e OWL

Dalla rappresentazione al ragionamento

- Ogni tripla RDF rappresenta un'unità atomica di ragionamento.
- Non ci sono relazioni, vincoli, connessioni implicite tra triple separate.
- Tutte le seguenti sono triple RDF perfettamente valide:

```
@prefix anobii: <http://www.anobii.com/books/>.  
@prefix foaf: <http://xmlns.com/foaf/0.1/>.  
@prefix db: <https://dbpedia.org/>.  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
```

```
anobii:Il_nome_della_rosa  
      foaf:made db:umbertoe.  
      
```

```
db:umbertoe  
      rdf:type foaf:writer;  
      rdf:type anobi:book;
```



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Dalla rappresentazione al ragionamento

- Sarebbe bello poter immaginare dei modi per descrivere un dominio di conoscenza e in base a quello poter:
 - generare nuova conoscenza derivandola dall'esistente
 - verificare che la conoscenza fornita sia coerente, sensata e non contraddittoria.
- RDF Schema (RDFS) è un insieme di classi e di regole di ragionamento che permette di generare nuova conoscenza (inferenze) sulla base di ciò che viene fornito esplicitamente.
- OWL (Web Ontology Language) estende ulteriormente le regole di inferenza e aggiunge regole di coerenza per segnalare contraddizioni.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF Schema

- Alcune classi e proprietà. Ad esempio:
 - Classi: rdfs:Resource, rdfs:Class, rdfs:Literal, rdfs:Datatype, etc.
 - Proprietà: rdfs:domain, rdfs:range, rdfs:subClassOf, etc.
 - Sono la concettualizzazione astratta del sistema di conoscenza class-oriented di RDF, e giustificano:
- Alcune regole di inferenza. Ad esempio:
 - `?e1 rdf:type ?c1.
?c1 rdfs:subClassOf ?c2.`  `?e1 rdf:type ?c2.`
 - `?e1 ?p1 ?v.
?p1 rdf:domain ?c3.`  `?e2 rdf:type ?c3.`
- RDFS non definisce alcun modo per verificare l'incoerenza tra inferenze.

`db:umbertoecco rdf:type foaf:writer.
foaf:writer rdfs:subClassOf foaf:Person.`



`db:umbertoecco rdf:type foaf:Person.`

Web Ontology Language (OWL)

- Definisce molte più proprietà di RDFS
 - sulle classi: intersectionOf, unionOf, complementOf, disjointWith, differentFrom, sameAs, etc.
 - sui predici: transitiveProperty, SymmetricProperty, reflexiveProperty, functionalProperty, etc.
- permette di creare regole di inferenza più ricche. Ad esempio:
 - `?c4 owl:unionOf [?c5, ?c6].`
`?e3 rdf:type ?c5.`  `?e3 rdf:type ?c4.`
 - `?c7 owl:disjointWith ?c8.`
`?e4 rdf:type ?c7.`
`?e4 rdf:type ?c8.`  **consistency error!**

`foaf:Person owl:disjointWith anobi:book.`

`db:umbertoeco`

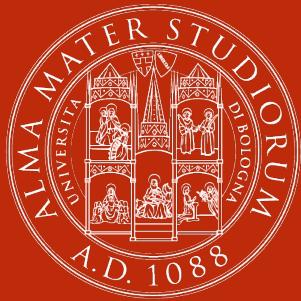
`rdf:type foaf:writer;`
`rdf:type anobi:book.`



consistency error!



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SPARQL

SPARQL

- Un acronimo ricorsivo per "SPARQL Protocol and RDF Query Language". Permette di fare ricerche su un dataset espresso in qualunque serializzazione di RDF e restituire tabelle di dati (*non* triple).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX dc: <http://purl.org/dc/terms/>
```

```
SELECT ?name
WHERE {
    ?p a foaf:Person .
    ?p foaf:name ?name .
    ?p foaf:made ?b .
    ?b a bibo:Book .
    ?b dc:title "Il Nome della Rosa".
}
```

Trova la/le persona/e presenti nel database che hanno scritto il libro il cui titolo è "Il Nome della Rosa" (e non, ad esempio, il film)



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SPARQL

Un acronimo ricorsivo per "SPARQL Protocol and RDF Query Language".

Permette di fare ricerche su un dataset espresso in qualunque serializzazione di RDF e restituire tabelle di dati (*non* triple).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX dc: <http://purl.org/dc/terms/>
```

```
SELECT ?name ?title
WHERE {
    ?p a foaf:Person .
    ?p foaf:name ?name .
    ?p foaf:made ?b .
    ?b a bibo:Book .
    ?b dc:title ?title .
}
```

Trova tutte le persone presenti nel database che hanno scritto un libro, e dammi il loro nome e il titolo del libro.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SPARQL

Un acronimo ricorsivo per "SPARQL Protocol and RDF Query Language".

Permette di fare ricerche su un dataset espresso in qualunque serializzazione di RDF e restituire tabelle di dati (*non* triple).

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX dc: <http://purl.org/dc/terms/>
```

```
SELECT ?name ?title
WHERE {
    ?p a foaf:Person .
    ?p foaf:name ?name .
    ?p foaf:made ?b .
    ?b a bibo:Book .
    ?b dc:title ?title .
}
```

Le parole che iniziano con ? sono variabili e possono assumere qualunque valore presente nel database

Ogni selezione di triple che soddisfa tutte le variabili della query viene aggiunta al grafo selezionato.

Solo le variabili specificate nel SELECT vengono restituite, però.

SPARQL Update

- Usa una sintassi simile a SPARQL per fare manipolazioni di dati su un database RDF: inserimenti, modifiche, cancellazioni, ecc.
- I comandi di inserimento semplicemente aggiungono triple al database.
- I comandi di modifica e cancellazione permettono prima di selezionare, attraverso un grafo con variabili, quali triple modificare, poi indicano come modificarle.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SPARQL Update - INSERT

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX dc: <http://purl.org/dc/terms/>
PREFIX anobii: <http://www.anobii.com/books/>.
PREFIX bibo: <http://www.anobii.com/books/>.

INSERT DATA {
    db:umbertoecho foaf:made anobii:Pendolo_di_focault.
    db:umbertoecho foaf:made anobii:Isola_del_Giorno_Prima.
    db:umbertoecho foaf:made anobii:Baudolino.
    anobii:Pendolo_di_focault
        a bibo:Book;
        dc:title "Il Pendolo di Foucault";
        dc:date "1988".
    anobii:Isola_del_Giorno_Prima
        a bibo:Book;
        dc:title "L'Isola del Giorno Prima";
        dc:date "1994".
    anobii:Baudolino
        a bibo:Book;
        dc:title "Baudolino";
        dc:date "2000".
}
```



SPARQL Update - DELETE

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bibo: <http://purl.org/ontology/bibo/>
PREFIX dc: <http://purl.org/dc/terms/>
PREFIX anobii: <http://www.anobii.com/books/>.
PREFIX bibo: <http://www.anobii.com/books/>.
```

```
DELETE {
    ?book ?pred ?obj .
} WHERE {
    ?book dc:date ?date .
    ?book ?pred ?obj .
    FILTER ( ?date > "1990" )
}
```

Cancella qualunque tripla che faccia match con il template specificato nel WHERE

?pred1 e ?obj sono qualunque, mentre ?book è tale per cui esiste anche la tripla con dc:date, e che il valore idi questa tripla sia superiore a 1990.

Il blocco WHERE dunque identifica tutte le triple che hanno anobii:Pendolo_di_focault come soggetto.

Attenzione: il libro "Il nome della Rosa" non ha una data, e verrà lasciato intatto.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Conclusioni

- Il Semantic Web è una disciplina a cavallo tra World Wide Web, teoria dei database, logica proposizionale e del primo ordine, e filosofia.
- Da sempre è stata considerata molto complessa ed astratta per le applicazioni web "normali", e quindi studiata con fastidio o ansia.
- In realtà, una volta prese un po' di precauzioni e cautele, è possibile realizzare applicazioni di GRANDE sofisticazione con sforzi sostanzialmente limitati.
- La possibilità di rappresentare grosse quantità di dati in maniera destrutturata e riorganizzabile a piacimento, con una struttura a grafo e con la possibilità di fare inferenze in maniera molto efficiente rende il SW molto significativo.



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Speaker: **Fabio Vitali**

Dipartimento di Informatica – Scienze e Ingegneria
Alma mater – Università di Bologna