



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

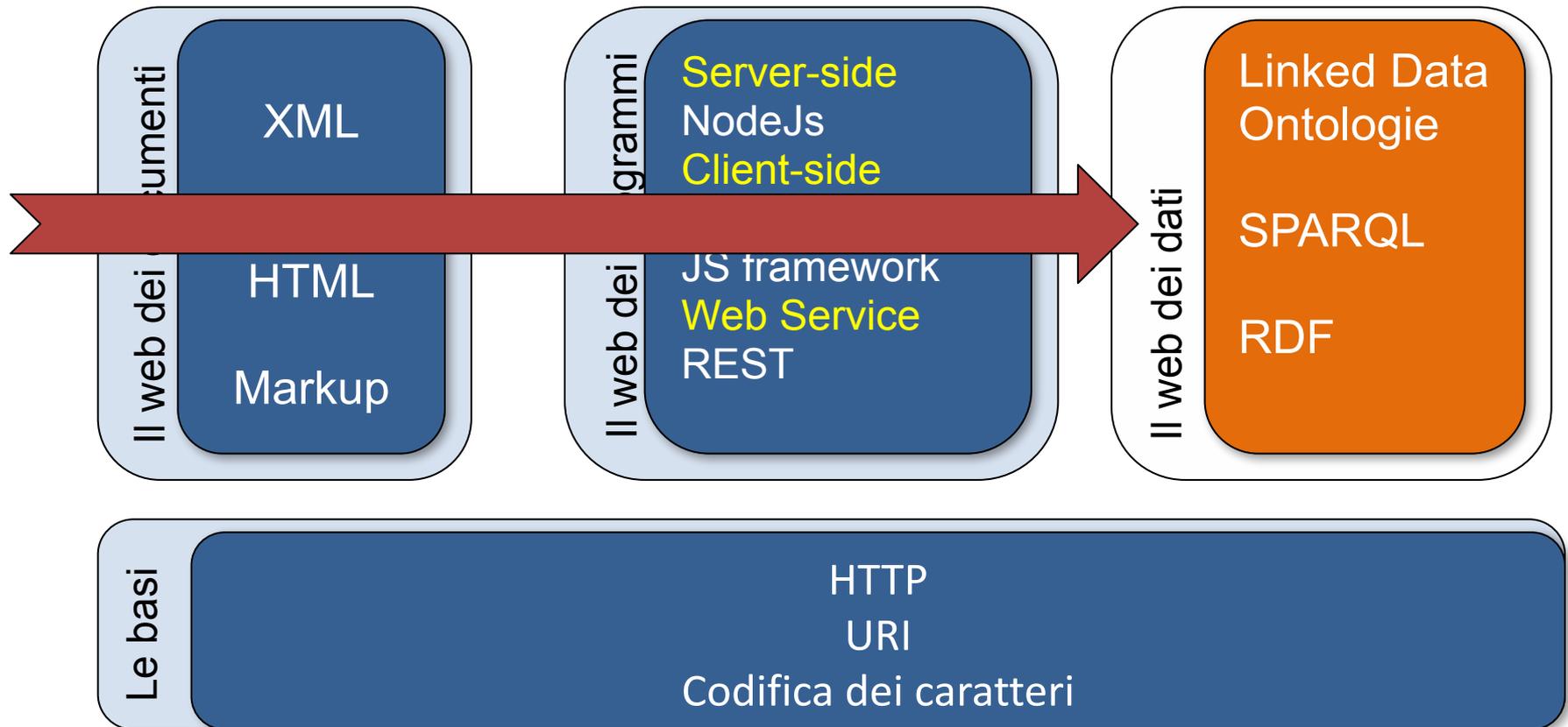
Semantic Web + Linked Open Data + RDF & JSON-LD

F. Sovrano

Corsi di laurea in Informatica e Informatica per il
Management

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

Semantic Web

Semantic Web

C'era una volta un gruppo di bibliotecari che avevano bisogno di un modo per descrivere i metadati: Topic Maps (ISO) vs. RDF (W3C).

RDF ha vinto ed è ha dato vita al Semantic Web e ai Linked Data.



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

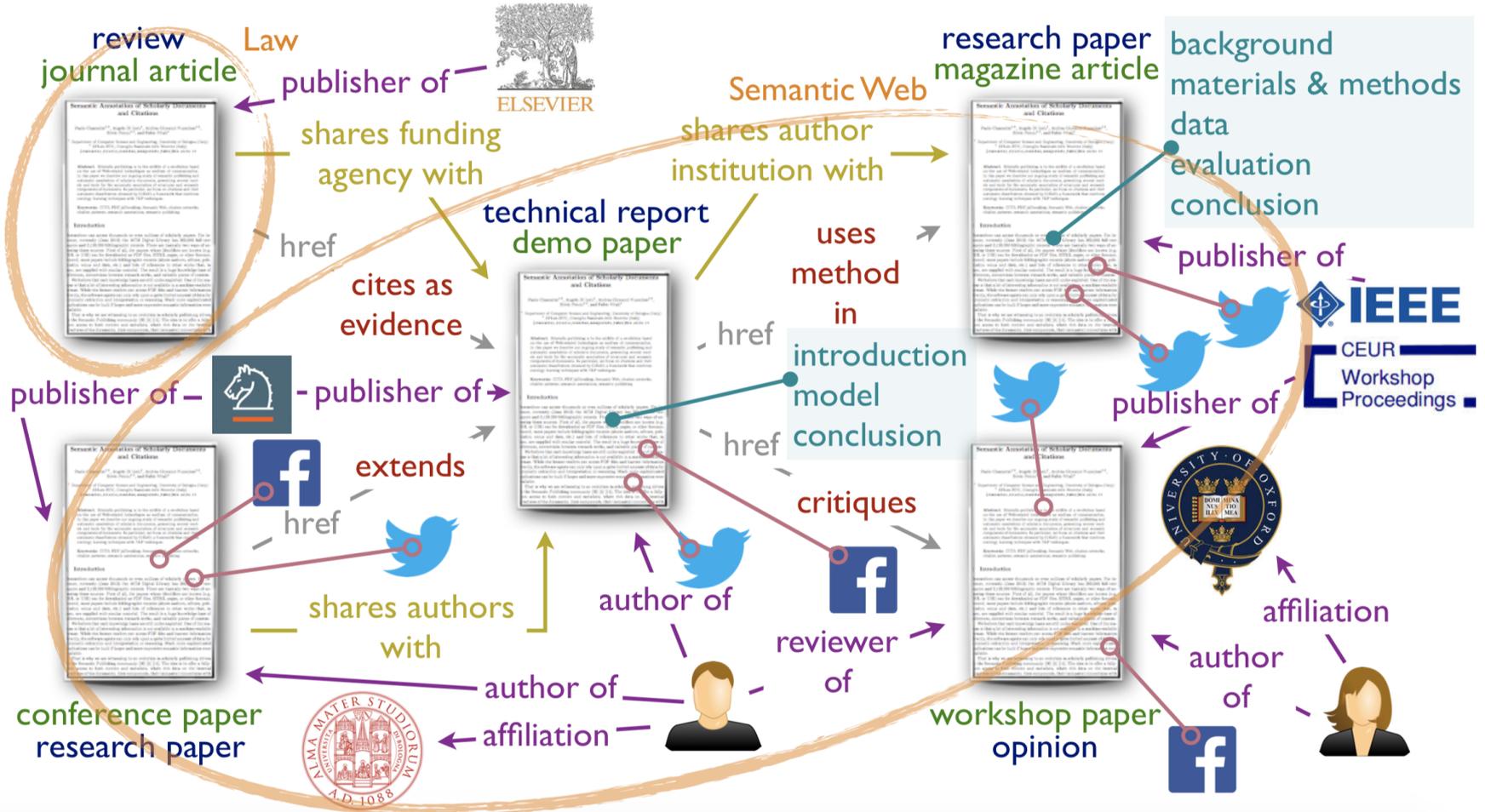
Il Semantic Web (SW) riguarda:

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



Semantic Web

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



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

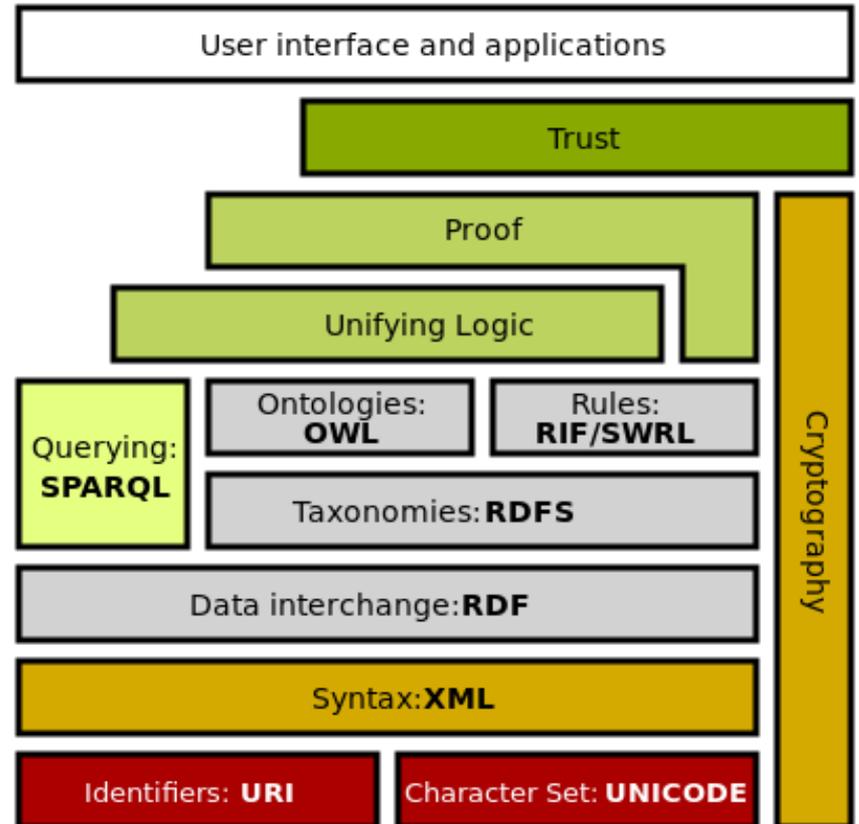
Ma, ancora, come possiamo passare dal vecchio web a quello semantico?

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



Semantic Web

Il [Semantic Web Stack](#) illustra l'architettura del Web Semantico.



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..



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

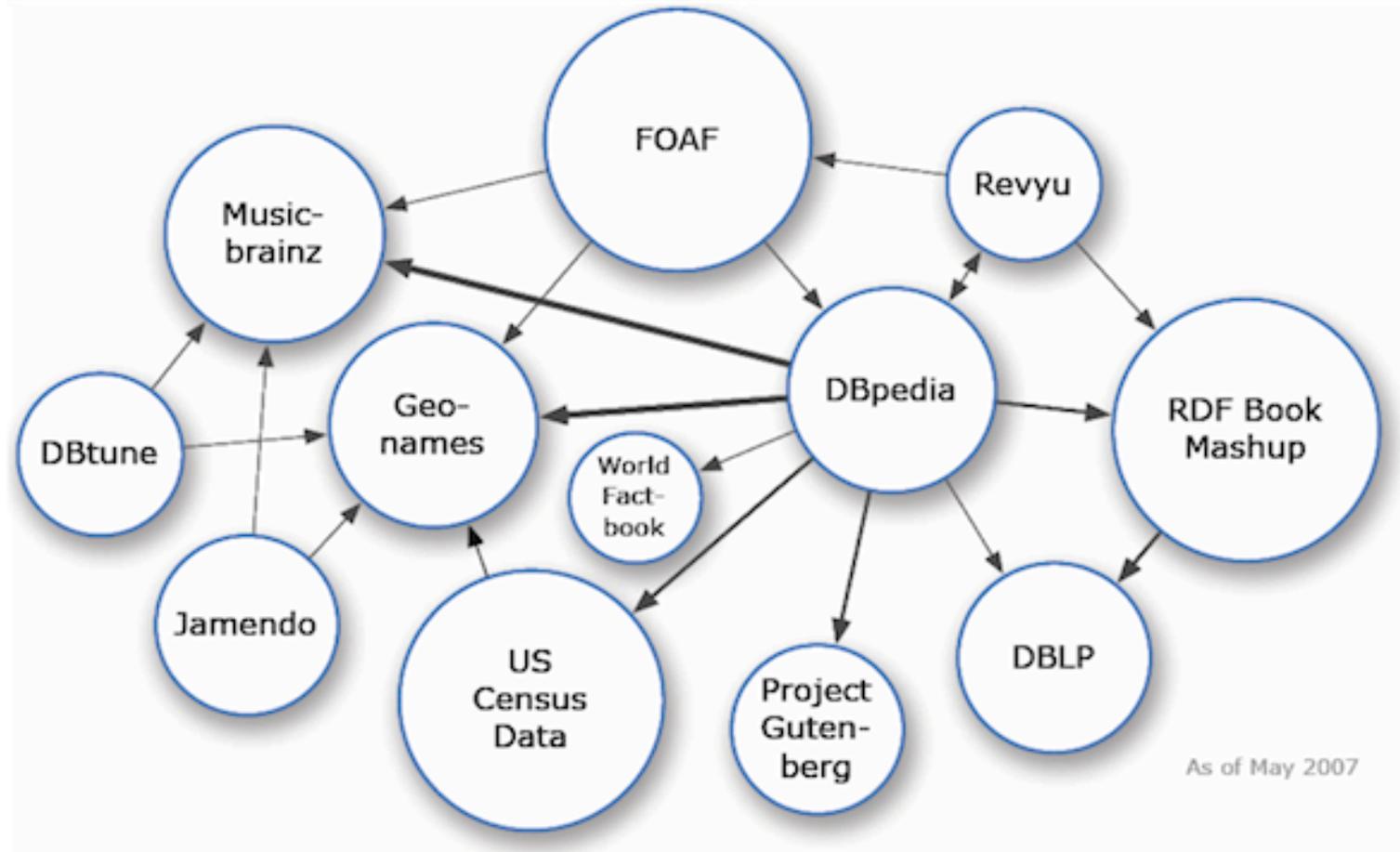
I Linked Data sono **dati strutturati** interconnessi con altri dati.

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**.

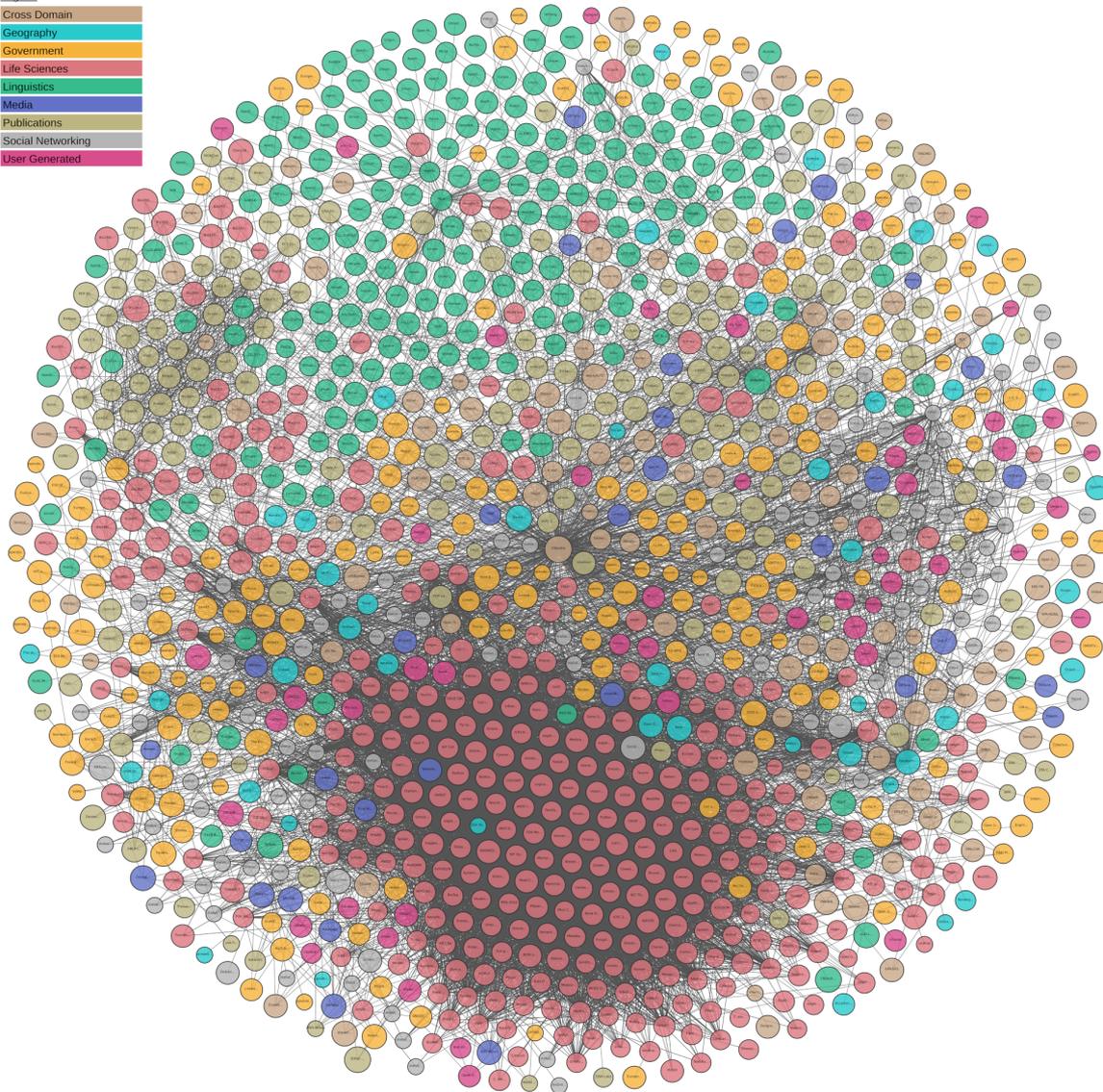
Linked Data è pensato per essere il **database RDF** del Semantic Web. I Linked Data **accessibili ed interrogabili pubblicamente** sono detti **Linked Open Data (LOD)**.



Linked Data - 2007



Linked Data - 2019





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

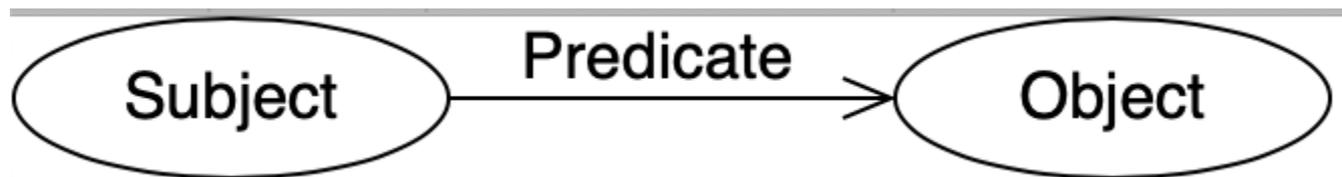
Resource Description Framework

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 (<https://twitter.com/umbertoeco_> l'URI dell'entità, può essere qualunque altro indirizzo però!).

E è una **autore** (“persona” è il suo tipo).

E si chiama “*Umberto Eco*”.

E ha scritto B (<http://www.anobii.com/books/Il_nome_della_rosa> l'URI del libro).

B è un **libro** (qui libro è il tipo RDF di B)

B si chiama “*Il Nome della Rosa*”

N.B. Questa notazione a colori non è standard, serve solo per far capire come esprimere in RDF.

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, cioè Resource Identifier. In base allo standard RDF dobbiamo associare questi verbi ad 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 **predicati** sono un URI indicanti una risorsa che rappresenti una relazione. Lo standard prevede l'utilizzo di URI.

- **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).

P.S. *A partire da RDF 1.1 parliamo di IRI invece che di URI.*



RDF

- **Ricapitolando** -

Il **soggetto** di uno statement RDF è anche lui un **URI**. Lo standard **non** prevede l'utilizzo di **stringhe letterali** come soggetti.

Ma, se il soggetto fosse qualcosa che esiste ma di cui non sappiamo il nome? Se volessimo dire in RDF che esiste qualcosa che è un libro, senza specificare cosa?

.. BOH! (risposta sbagliata)



RDF - le risorse anonime

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

- un identificatore di risorsa uniforme (**URI**),
- 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:

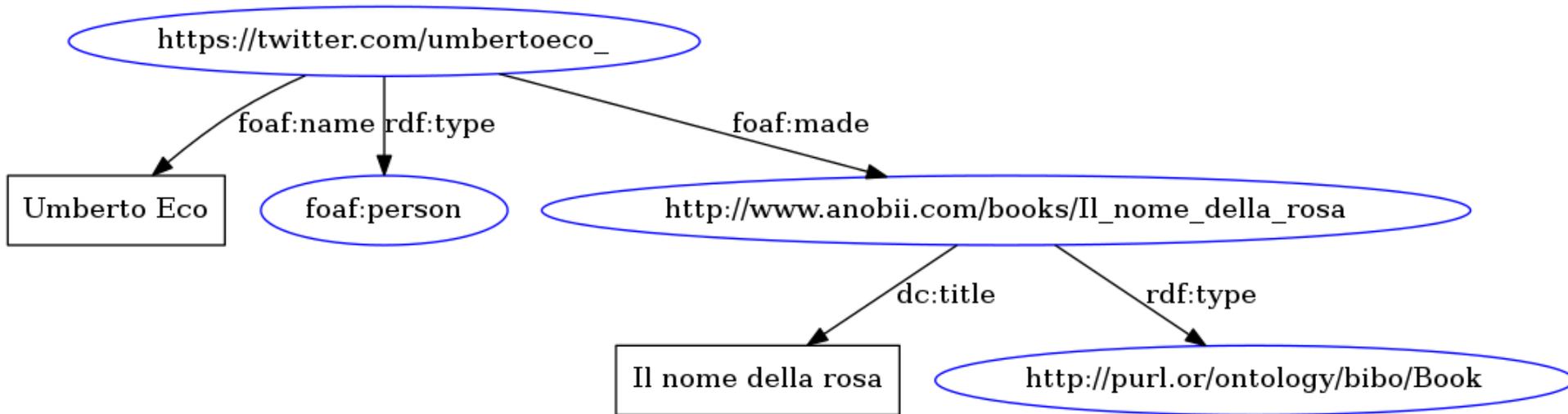
- Le **risorse** (**soggetti** e **oggetti**) sono rappresentate come nodi.
- I **predicati** sono rappresentati come archi.

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



RDF - i grafi

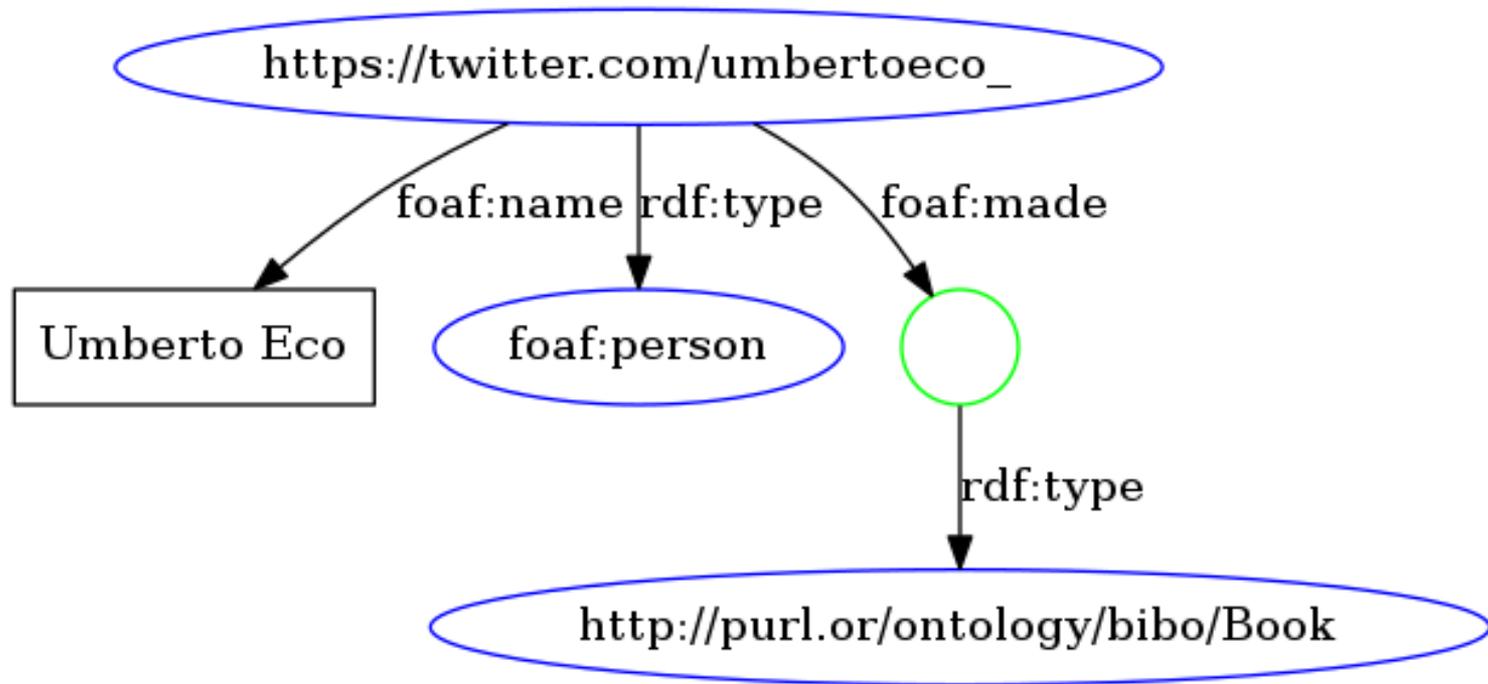
Esempio: *“Umberto Eco è autore de Il Nome della Rosa”*:



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: *“Umberto Eco è autore di un libro”*:



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

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 - il vocabolario base

RDF ha un **vocabolario** con classi e proprietà

Esempi di **classi** sono:

- **rdf:XMLLiteral** – la classe dei **valori letterali** XML (**N.B.** non è la classe delle risorse!)
- **rdf:Property** – la classe delle **proprietà**
- **rdf:Statement** – la classe delle triple RDF
- **rdf:Alt**, **rdf:Bag**, **rdf:Seq** – contenitori di alternative, contenitori non ordinati, e contenitore ordinati (rdfs:Container è super-classe di tutti i contenitori)
- **rdf:List** – la classe delle liste RDF
- **rdf:nil** – una istanza di rdf:List che rappresenta la lista vuota



RDF - il vocabolario base

Esempi di **proprietà** sono:

- **rdf:type** – un'istanza di `rdf:Property` usata per dire che una risorsa è istanza di una classe
- **rdf:first** – il primo elemento nella lista RDF **soggetto**
- **rdf:rest** – il resto della lista RDF **soggetto**; gli elementi diversi da **rdf:first**
- **rdf:value** – proprietà idiomatica usata per valori strutturati
- **rdf:subject** – il **soggetto** della tripla RDF **soggetto**
- **rdf:predicate** – il **predicato** della tripla RDF **soggetto**
- **rdf:object** – l'**oggetto** della tripla RDF **soggetto**

Usato spesso

RDFS estende il vocabolario base di RDF. Con RDFS possiamo definire classi, sottoclassi, ecc..

OWL usa RDFS e 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).



RDF

Le triple RDF possono essere memorizzate in un tipo di database chiamato **triplestore**. A differenza di un db relazionale, un triplestore è **ottimizzato per** la memorizzazione e il recupero di **triple RDF**. Alcuni esempi di triplestore (nativi) sono:

- [4Store](#)
- [AllegroGraph](#)
- [BigData](#)
- [Jena TDB](#)
- [Sesame](#)
- [Stardog](#)
- [OWLIM](#)
- [uRiKa](#)





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.



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.



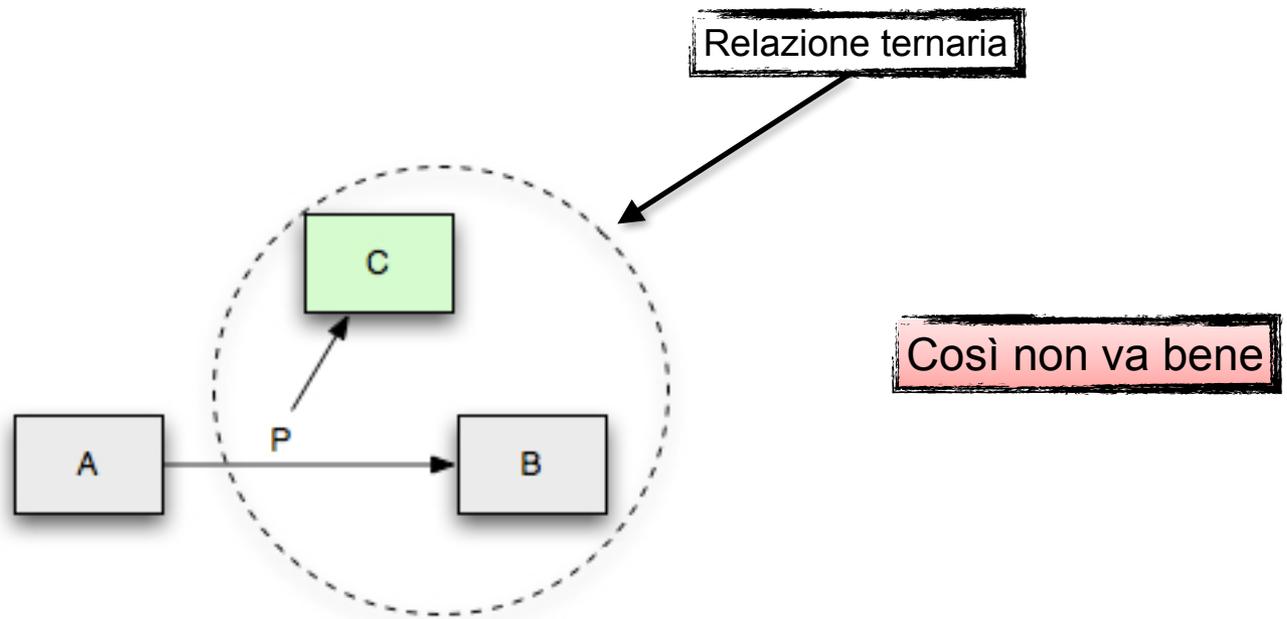
RDF - Svantaggi

- 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.
- La 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**.



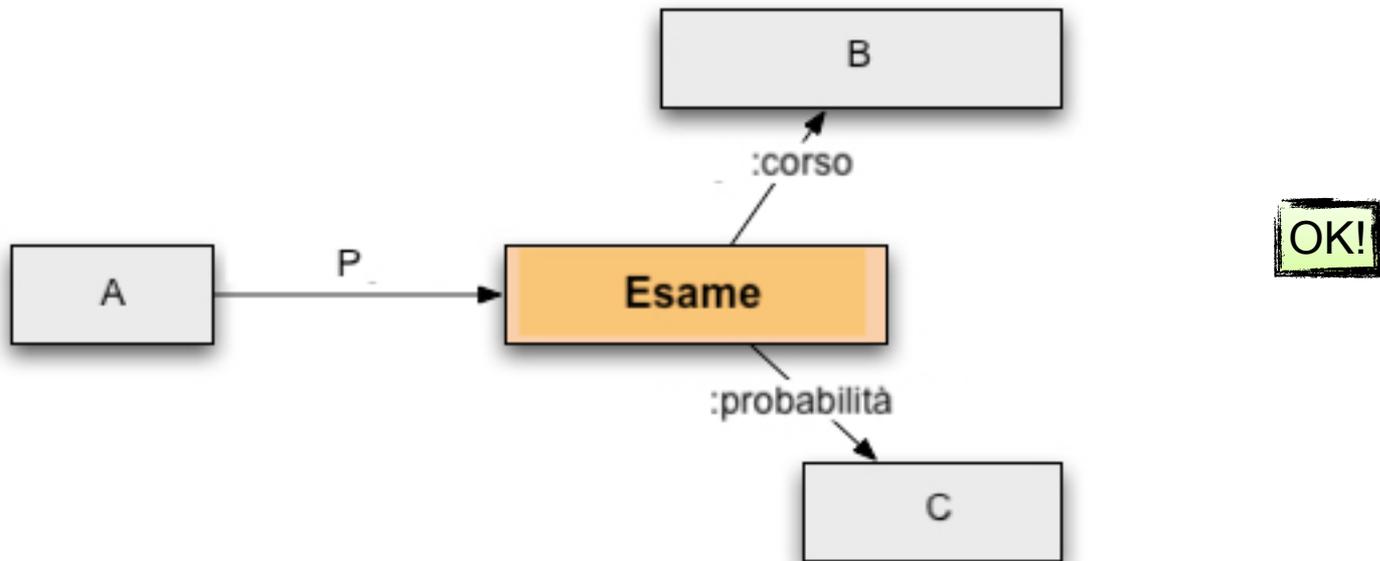
RDF - Svantaggi

- **Esempio:** Pincopallina (A) parteciperà all'esame (P) di TW (B), con alta probabilità (C).



RDF - Svantaggi

- **Esempio:** Pincopallina (A) parteciperà all'esame (P) di TW (B), con alta probabilità (C).





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

RDF - Esempi

RDF - Esempi

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

Esempio 2: *“Umberto Eco è autore di un libro”*

Come possiamo scriverli in **Turtle** o **RDF/XML**?



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/> .
```

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

```
@prefix twitter: <https://twitter.com/> .
```

```
twitter:umbertoeco_
```

```
  foaf:name "Umberto Eco" ;
```

```
  rdf:type foaf:person ;
```

```
  foaf:made anobii:Il_nome_della_rosa .
```

```
anobii:Il_nome_della_rosa
```

```
  dc:title "Il nome della rosa" ;
```

```
  a <http://purl.or/ontology/bibo/Book>
```



RDF - Turtle

Esempio 2: *“Umberto Eco è autore di un libro”*:

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

<https://twitter.com/umbertoeco_>

foaf:name "Umberto Eco" ;

a foaf:person ;

foaf:made [a <http://purl.org/ontology/bibo/Book>] .



RDF - RDF/XML

Esempio 2: “Umberto Eco è autore di un libro”

```
1. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-  
rdf-syntax-ns#"
2.         xmlns:foaf="http://xmlns.com/foaf/0.1/">
3.   <foaf:person rdf:about="https://twitter.com/  
umbertoeco_">
4.     <foaf:name>Umberto Eco</foaf:name>
5.     <foaf:made>
6.       <rdf:Description>
7.         <rdf:type rdf:resource="http://purl.or/  
ontology/bibo/Book"/>
8.       </rdf:Description>
9.     </foaf:made>
10.  </foaf:person>
11.</rdf:RDF>
```



RDF - RDF/XML

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

```
1. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
2.     xmlns:foaf="http://xmlns.com/foaf/0.1/"
3.     xmlns:dc="http://purl.org/dc/terms/">
4.   <foaf:person rdf:about="https://twitter.com/umbertoeco_">
5.     <foaf:name>Umberto Eco</foaf:name>
6.     <foaf:made>
7.       <rdf:Description rdf:about="http://www.anobii.com/books/
8.         Il_nome_della_rosa">
9.           <dc:title>Il nome della rosa</dc:title>
10.          <rdf:type rdf:resource="http://purl.or/ontology/bibo/Book"/>
11.        </rdf:Description>
12.      </foaf:made>
13.    </foaf:person>
14.  </rdf:RDF>
```





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

JSON-LD

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**.



JSON-LD - Esempio 1

Esempio 1: “Il link YouTube del video ‘Il Pulcino Pio’ è [juqyzgnbspY](https://www.youtube.com/watch?v=juqyzgnbspY)”

Turtle

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

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

@prefix mpeg: <<http://purl.org/ontology/mpeg7/>> .

@prefix my: <<http://my-wonder-site/ontology/video/>> .

<http://my-wonder-site/pulcino_pio>

a mpeg:Video ;

dc:title "Il Pulcino Pio"^^xsd:string ;

my:link <<https://www.youtube.com/watch?v=juqyzgnbspY>> .

Attenzione: facile confondere letterali con URI, in JSON-LD. Il contesto serve a disambiguare

Annotazione di tipo sul letterale



JSON-LD - Esempio 1

Esempio 1: “Il link YouTube del video ‘Il Pulcino Pio’ è [juqyzgnbspY](https://www.youtube.com/watch?v=juqyzgnbspY)”

JSON-LD

```
{
  "@context": {
    "title": "http://purl.org/dc/terms/title" ,
    "link": "http://my-wonder-site/ontology/video/link",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "@id": "http://my-wonder-site/pulcino\_pio" ,
  "@type": "http://purl.org/ontology/mpeg7/Video" ,
  "title": {
    "@type": "xsd:string",
    "@value": "Il Pulcino Pio"
  },
  "link": "https://www.youtube.com/watch?v=juqyzgnbspY"
}
```

2 triple con
soggetto anonimo

JSON-LD

JSON-LD contiene specifici **nomi riservati**, *tutti con prefisso '@'*:

- “@id”: permette di specificare l’IRI della risorsa di cui si vuol parlare
- “@value”: permette di specificare un valore letterale (es.: una stringa, un numero)
- “@type”: permette di specificare l’IRI del tipo (rdf:type) associato alla risorsa o al valore
- “@context”: permette di specificare dei nomi abbreviati da usare all’interno di un documento JSON-LD



JSON-LD

Ogni coppia “chiave-valore” definisce un nuovo statement RDF che ha per soggetto la risorsa definita in “@id”.

A seguire altri esempi..

Esempio 2:

Il video “[Volevo un gatto nero](#)”, è stato pubblicato il 9 Maggio 2012 dallo [Zecchino d’Oro](#).



JSON-LD - Esempio 2

```
1. @prefix dc: <http://purl.org/dc/terms/> .
2. @prefix video: <http://purl.org/ontology/video/> .
3. @prefix temporal: <http://swrl.stanford.edu/ontologies/
  built-ins/3.3/temporal.> .
4. @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

5. <https://www.youtube.com/watch?v=z_aVNv_gNdM>
6.   a <http://purl.org/ontology/mpeg7/Video> ;
7.   dc:title "Volevo un gatto nero"^^xsd:string ;
8.   dc:created "2012-05-09"^^xsd:date ;
9.   video:madeBy "Zecchino d'Oro"^^xsd:string ;
10.  dc:description "I Cartoni dello Zecchino d'Oro -
  Volume 4"^^xsd:string ;
11.  dc:publisher <https://www.youtube.com/channel/UC-
  U2fFVqtrPTDo_QmiLSiIg> ;
12.  temporal:duration "PT2M33S"^^xsd:duration .
```

Turtle



JSON-LD - Esempio 2

JSON-LD

```
{
  "@context": {
    "dcterms": "http://purl.org/dc/terms/",
    "video": "http://purl.org/ontology/video/",
    "temporal": "http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "@id": "https://www.youtube.com/watch?v=z_aVNv_gNdM",
  "@type": "http://purl.org/ontology/mpeg7/Video",
  "dcterms:created": {
    "@type": "xsd:date",
    "@value": "2012-05-09"
  },
  "dcterms:description": "I Cartoni dello Zecchino d'Oro - Volume 4",
  "dcterms:publisher": {
    "@id": "https://www.youtube.com/channel/UC-U2fFVqtrPTDo\_QmiLSilg"
  },
  "dcterms:title": "Volevo un gatto nero",
  "temporal:duration": {
    "@type": "xsd:duration",
    "@value": "PT2M33S"
  },
  "video:madeBy": "Zecchino d'Oro"
}
```



JSON-LD - EasyRDF



Converter

Input Data:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://www.linkeddatatools.com/johndoe>
  a foaf:Person ;
  foaf:name "John Doe"^^xsd:string ;
  foaf:workplaceHomepage <http://www.linkeddatatools.com/> .
```

or Uri:

http://njh.me/

Input Format:

Turtle Terse RDF Triple Language

Output Format:

JSON-LD

Raw output

Clear

Submit

JSON-LD - EasyRDF

Output

Number of triples parsed: 3

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://www.linkeddatatools.com/johndoe>
  a foaf:Person ;
  foaf:name "John Doe"^^xsd:string ;
  foaf:workplaceHomepage <http://www.linkeddatatools.com/> .
```

This converter is running version **0.9.0** of EasyRdf.





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Discussioni & Conclusioni

Conclusioni

Abbiamo visto più o meno velocemente:

- **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

Ancora un paio di considerazioni prima di salutarci..



Semantic Web

Questa idea di web semantico è in circolazione dal 1999. Ma è ancora lontana dall'essere realizzata, perché?

Problema 1: Mancanza di un vantaggio generico evidente, nel breve termine, che giustifichi l'**enorme sforzo richiesto** (in termini di anni-uomo) per la realizzazione del Semantic Web. In altre parole, troppa fatica per l'uomo: l'apprendimento dei linguaggi e degli strumenti di rappresentazione della conoscenza richiede a chi lavori sul SW di conoscere i metodi di rappresentazione dell'astrazione e il loro effetto sul ragionamento.



Semantic Web

Soluzione 1: Il web semantico dovrebbe essere costruito da agenti automatici.

Problema 2: Esiste una tecnologia per ottenere agenti automatici in grado di costruire il web semantico?



Semantic Web

Soluzione 2: Abbiamo tecnologie per *information retrieval*, *automatic feature extraction*. Le macchine ora possono **imparare** a capire i dati ed estrarre informazioni da essi: il linguaggio naturale, le immagini, i suoni, ecc.. È solo questione di tempo?

Problema 3: Ma se un agente automatico fosse già in grado di capire ed analizzare il web (non semantico), allora qual è la necessità di avere un web semantico?



Semantic Web

Soluzione 3: La *comprensione (dei dati)* può essere computazionalmente costosa. Possiamo vedere il web semantico come il prodotto di un lungo processo di comprensione dei dati, come una **cache**, un **database** per il recupero rapido delle informazioni relative alla semantica dei dati nel web.

Ma il caching a volte ha un costo. Possiamo permettercelo? Qual è questo costo?

- **spazio** fisico in memoria, con tutti i problemi di gestione di big-data annessi?
- essere in grado di accettare formalismi e standard comuni?



Semantic Web

Quale potrebbe essere il **grande vantaggio** dietro ad un possibile successo del Semantic Web?

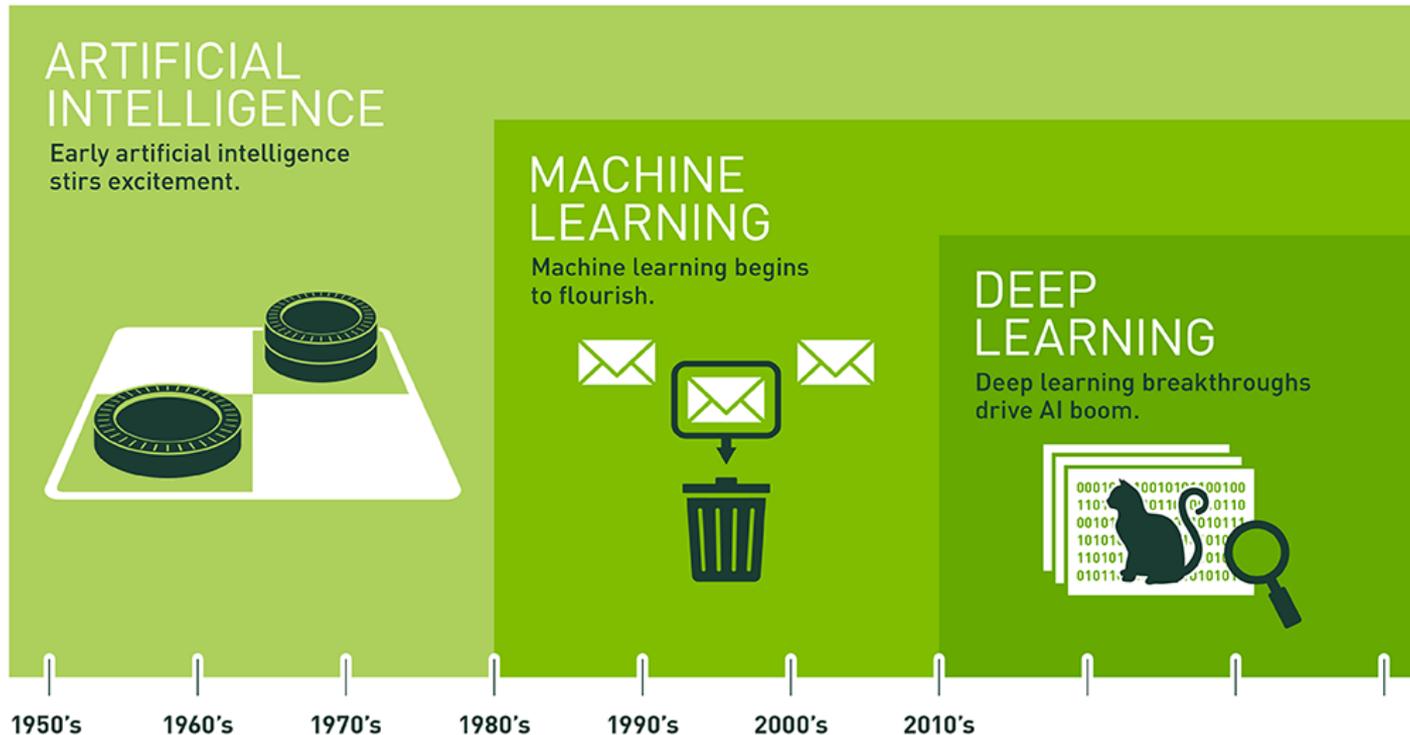
Avere un **database pubblico distribuito** potrebbe essere sicuramente utile per **potenziare**, migliorare le **IA** che apprendono dai dati.

Più nel dettaglio, le IA basate su tecnologie di **deep learning**.



Semantic Web

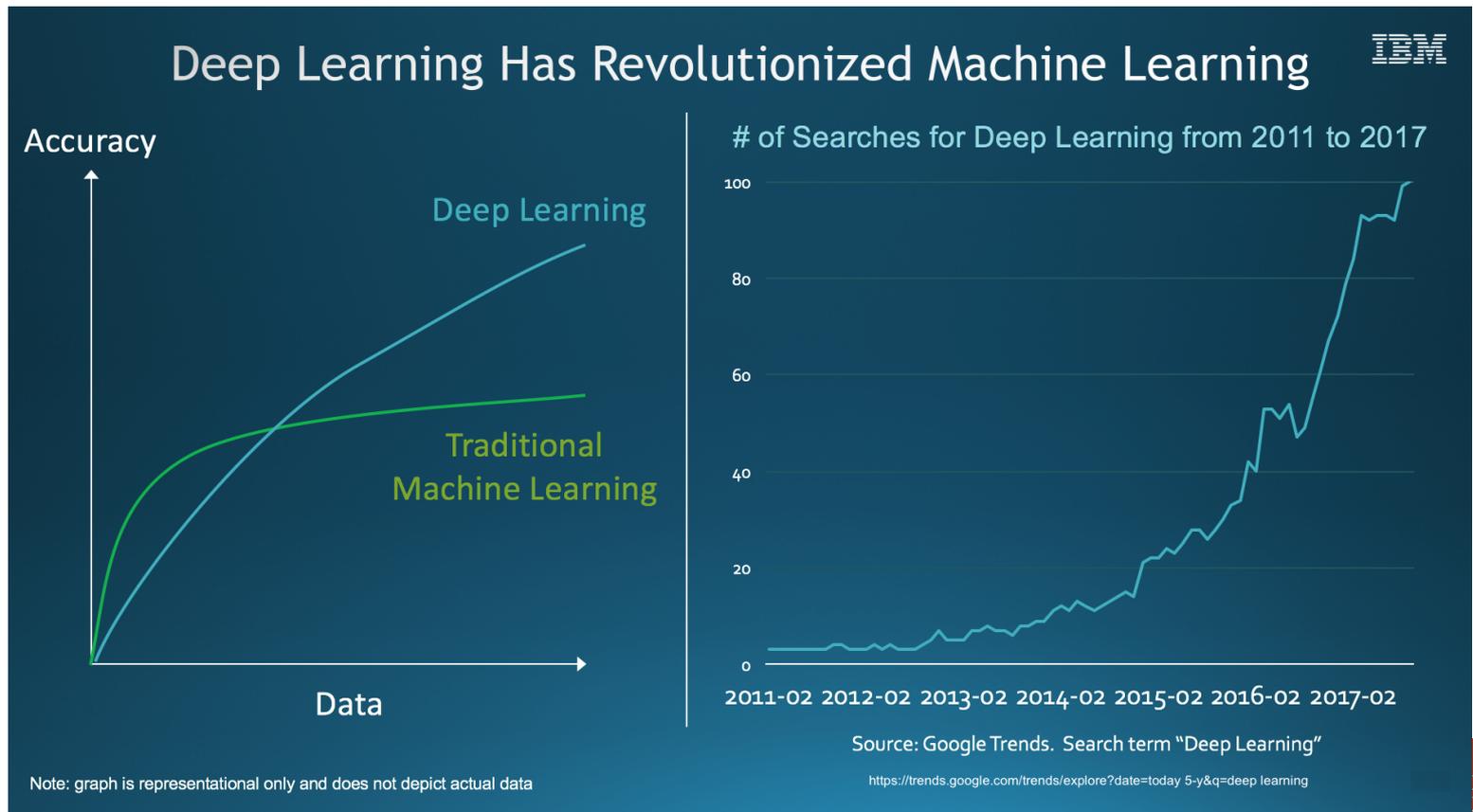
Grazie al **deep learning**: più sono i dati, più la macchina è potenzialmente in grado di imparare.



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Semantic Web

In altre parole: l'**intelligenza** della macchina sembra essere **proporzionale** anche alla **quantità di dati** usati per il suo allenamento, e i big-data non sono sempre facili da reperire.



Semantic Web

Il Problema 1 sta ancora guidando lo **stallo** del web semantico. Non sorprendentemente, la maggior parte del web non è stata ancora semantizzata.

Ad ogni modo, sono stati fatti molti sforzi per costruire le basi del web semantico. Ecco perché, molto spesso il termine "*web semantico*" viene associato a **un insieme di tecnologie e standard**.



Idee e Proposte

Ci piacerebbe che i più volenterosi di voi proponessero **idee e soluzioni** innovative ai problemi di oggi del semantic web.

Fate autonomamente una piccola ricerca sul web prima di proporci le idee, giusto per farvi un'idea sulla fattibilità.

N.B. Quanto scritto in questa slide è assolutamente facoltativo, ma consigliato



Esempi di Idee

- **Open Information Extraction**: Da testo in linguaggio naturale a RDF (<https://resources.mpi-inf.mpg.de/d5/clusie/clusie-www13.pdf>).
- **Multilingual RDF Verbalizer**: Da RDF a testo in linguaggio naturale (<https://github.com/dbpedia/neural-rdf-verbalizer>).
- **Visualizing Large Knowledge Graphs**: Visualizzazione di grafi RDF estesi (<https://www.sciencedirect.com/science/article/pii/S0167739X17323610>).
- **Aligning large ontologies for semantic interoperability**: Disambiguazione di concetti (<https://www.inderscience.com/info/inarticle.php?artid=102120>).





ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Speaker: **Francesco Sovrano**

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

Website: unibo.it/sitoweb/francesco.sovrano2

E-mail: francesco.sovrano2@unibo.it