



Linked Data



L'evoluzione del Web
Semantico



Finalità

- Testo di riferimento: T. Heath and C. Bizer (2011) *Linked Data: Evolving the Web into a Global Data Space* (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology, 1:1, 1-136. Morgan & Claypool.
- <http://linkeddatabook.com/editions/1.0/>
- “Every ontology is an island”
vs.
- I dati provenienti da sorgenti diverse devono essere collegati e interrogati

Definizione

- Linked Data è un metodo per pubblicare dati strutturati in modo che possano essere collegati e diventare più utili attraverso queries semantiche
- Si basa su tecnologie Web standard come HTTP, RDF e URI.
- Queste tecnologie, normalmente usate per i documenti destinati agli esseri umani, vengono estese per condividere informazioni leggibili automaticamente dai computer.

Principi cardine

1. Usare gli URI per denominare (identificare) le entità.
2. Usare URI HTTP in modo che queste entità possano essere trovate (o interpretate, o "de-referenziate").
3. Fornire informazioni sulla risorsa quando viene richiesta usando standard come RDF, SPARQL, ecc.
4. Nel contesto di un'entità, riferirsi sempre alle altre entità usando i loro URI HTTP quando si pubblicano i dati su Web.

Informazioni sulla risorsa

- Non solo la risorsa deve essere de-referenziata (= all'URI della risorsa trovo la risorsa stessa), ma insieme alla risorsa deve essere fornita una descrizione della risorsa stessa.
- La descrizione può essere fornita ad esempio via SPARQL o RDF all'URI della risorsa
- Dovrebbe includere relazioni con altre risorse, indicate via i loro URI.

The background of the slide features several thin, curved lines in shades of gray, some solid and some dashed, creating a sense of motion or data flow. A blue rectangular box with a speech bubble tail at the bottom is positioned on the left side.

Linked Data Platform (LDP)

- Recommendation: Linked Data Platform 1.0 – 26 Feb 2015
“Use of HTTP for accessing, updating, creating and deleting resources from servers that expose their resources as Linked Data.”
- Una piattaforma per l’accesso ai linked data che utilizza paradigma REST
- Web service basato sui metodi di HTTP per leggere e scrivere i Linked Data

Cosa significa de-referenziabile

- Ogni URI HTTP deve essere “dereferenziabile”
- Significa che un client può richiedere la risorsa denominata da un URI HTTP e ottenerla via HTTP, accompagnata da una descrizione della risorsa.
- Le descrizioni sono indirizzate alle macchine e rappresentate in formato RDF, i documenti agli esseri umani
 - Es. Iconclass

RDF e content negotiation

- Content negotiation: permettere a chi referencia un URL di specificare se vuole la risorsa come documento HTML o come RDF
- Secondo il meccanismo di negoziazione proprio di HTTP:
 - Il client manda degli *header* specificando che tipo di documento preferisce
 - Il server riceve gli *header* e seleziona la response appropriata.

Rendere gli URIs de- referenziabili

- Due strategie principali:
 - *303 URIs*
 - *hash URIs*

303 URIs (2 step)

1. Il cliente effettua una *request* HTTP di tipo GET di un URI (che identifica un oggetto reale o un concetto)
2. Invia un header `Accept: application/rdf+xml`
3. Il server risponde usando il codice HTTP 303 See Other e invia l'URI di un documento RDF che descrive la risorsa
4. Il client effettua una request HTTP GET dell'URI restituito dal server

HASH URIs

- Il metodo precedente richiede due coppie *request-response* via HTTP per ottenere il documento RDF
- La strategia denominata *hash URI* si basa sul fatto che gli URI possono contenere una parte speciale separata dall'URI di base dal simbolo #
 - Questa parte speciale è chiamata fragment identifier.

in HTTP

- In HTTP la parte di URI che segue # è eliminata dall'URI prima che esso sia cercato al dominio specificato
 - HTTP ignora la parte che segue #
- Questa tecnica viene usata per associare un vocabolario a un certo URI:
 - Il vocabolario viene scaricato per intero
 - Il client cerca il termine che gli interessa nel file

Hash URIs:

- Vantaggi: solo 1 scambio
- Svantaggi: richiede interpretazione da parte del client
- Quindi: per pubblicare intere terminologie (il Linked Data-aware client le gestisce per conto proprio)

303 URIs

- Vantaggi: il client riceve direttamente la descrizione della risorsa
- Svantaggi: richiede due scambi
- Quindi: adatto per pubblicare descrizioni di singole risorse

Confronto tra Hash e 303



Tipologie di links nelle triple RDF

- **Relationship links:** punta a oggetti collegati in altre sorgenti di dati
 - Es. Un luogo
- **Identity links:** punta uno o più URI che fungono da alias in altre sorgenti di dati per identificare la stessa entità URI (normalmente un oggetto reale).
- **Vocabulary links:** puntano alle definizioni dei termini del vocabolario

Relationship link: example

6 foaf:name "Dave Smith" ;

7 foaf:based_near <<http://sws.geonames.org/3333125/>> ;

8 foaf:based_near <<http://dbpedia.org/resource/Birmingham>> ;

Il predicato foaf:based_near collega la risorsa con due entità diverse

<http://dbpedia.org/resource/Birmingham>

e <http://sws.geonames.org/3333125/> sono entità collegate alla risorsa che si trovano in altre sorgenti di dati

Identity links

- Nel mondo 'aperto' dei linked data è comune che la stessa entità venga riferita da sorgenti di dati diversi con URI diversi.
- Linked Data indica l'utilizzo del predicato **<http://www.w3.org/2002/07/owl#sameAs>** per affermare che due URI si riferiscono alla stessa risorsa
- Esempio:
<<http://www.dave-smith.eg.uk#me>>
<<http://www.w3.org/2002/07/owl#sameAs>>
<<http://biglynx.co.uk/people/dave-smith>> .

Vocabulary links: Schema

- Nei Linked Data, una descrizione è un misto di termini diversi provenienti da vocabolari RDF diversi (sia proprietari che standard)
- LD suggeriscono l'utilizzo di vocabulary standard ma quando ciò non è possibile, è necessario fare in modo che i propri termini siano de-referenziabili (e collocati in un vocabolario che li descrive).
- Se si scopre che esiste già un termine equivalente, è possibile stabilire la relazione tra i due mediante
owl:equivalentClass e **owl:equivalentProperty**
(rdfs:subClassOf, rdfs:subPropertyOf, skos:broadMatch, eskos:narrowMatch per una relazione più lasca)

Esempio vocabulary links

```
1  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
3  @prefix owl: <http://www.w3.org/2002/07/owl#> .
4  @prefix co: <http://biglynx.co.uk/vocab/sme#> .
5
6  <http://biglynx.co.uk/vocab/sme#SmallMediumEnterprise>
7  rdf:type rdfs:Class ;
8  rdfs:label "Small or Medium-sized Enterprise" ;
9  rdfs:subClassOf <http://dbpedia.org/ontology/Company> .
10 rdfs:subClassOf <http://umbel.org/umbel/sc/Business> ;
11 rdfs:subClassOf <http://sw.opencyc.org/concept/Mx4rvV...ycA> ;
12 rdfs:subClassOf <http://rdf.freebase.com/ns/m/0qb7t> .
```

Qui la classe SmallMediumEnterprise è collegata come sottoclasse con quattro termini (righe 9-12) di altri vocabolari

Design degli URI

- Coniare gli URI: un *data publisher* un namespace `http://namespace` che controlla, per il fatto di possederne il dominio, installa un Web server in quella location, e *conia* gli URI in quel *namespace* per identificare le entità nel suo data set.
 - Per esempio, *Big Lynx* possiede il dominio `biglynx.co.uk` e ha server `http` all'indirizzo
`http://biglynx.co.uk/`.
 - Se *Big Lynx* desidera forgiare gli URIs per identificare i membri del proprio staff, può utilizzare quel namespace
`http://biglynx.co.uk/people/`.

Linee guida per gli URI (1)

- Evitare i dettagli implementativi
 - NO:
<http://tiger.biglynx.co.uk/people.php?id=dave-smith&format=rdf>
 - SI:
<http://biglynx.co.uk/people/dave-smith.rdf>
(Usare il mod_rewrite module di Apache per convertire gli indirizzi)
- Usare termini significativi nel dominio dato



Linee guida per gli URI (2)

Distinguere vocabolario, dati e risorse

Con i path

- http://dbpedia.org/resource/Wildlife_photography
- http://dbpedia.org/page/Wildlife_photography
- http://dbpedia.org/data/Wildlife_photography

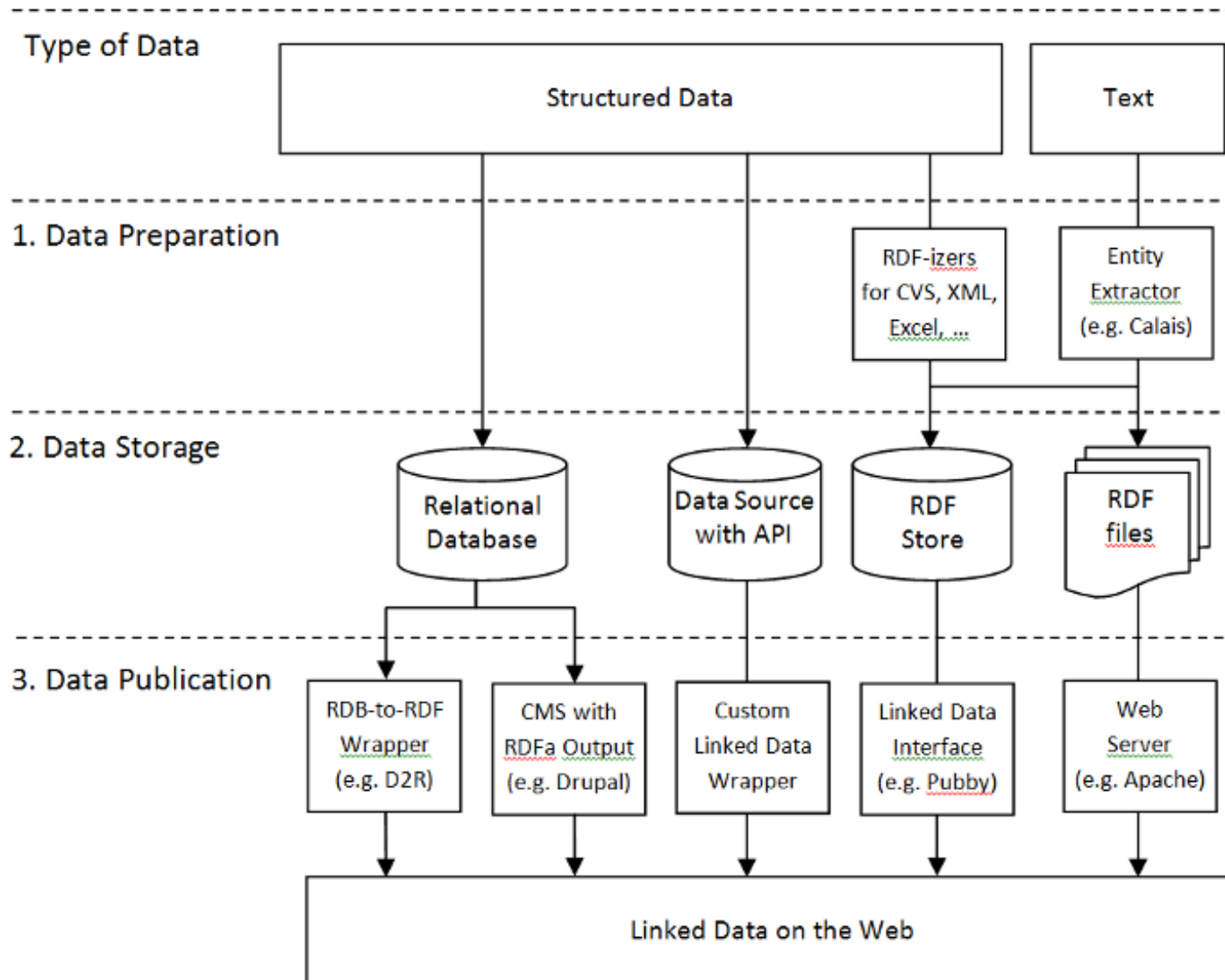
Oppure sotto-domini

- <http://id.biglynx.co.uk/dave-smith>
- <http://pages.biglynx.co.uk/dave-smith>
- <http://data.biglynx.co.uk/dave-smith>

Oppure percorsi

- <http://biglynx.co.uk/people/dave-smith> → risorsa
- <http://biglynx.co.uk/people/dave-smith.html> → html
- <http://biglynx.co.uk/people/dave-smith.rdf> → rdf

Schema pubblicazione

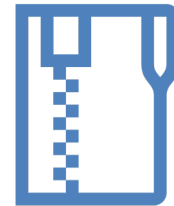


Tipi di dato (1)

- Il primo criterio per selezionare un workflow per la pubblicazione dei Linked Data riguarda il tipo di dati in input
 - Dinamici vs statici

Dati Dinamici

- Dati strutturati conservati in un db → mapping to RDF via software (wrappers, e.g. DR2)
- Dati pubblicati via API → mapping to RDF via *custom mapping*
- Formati statici diversi da RDF (XML, csv, ecc.) → RDFizers

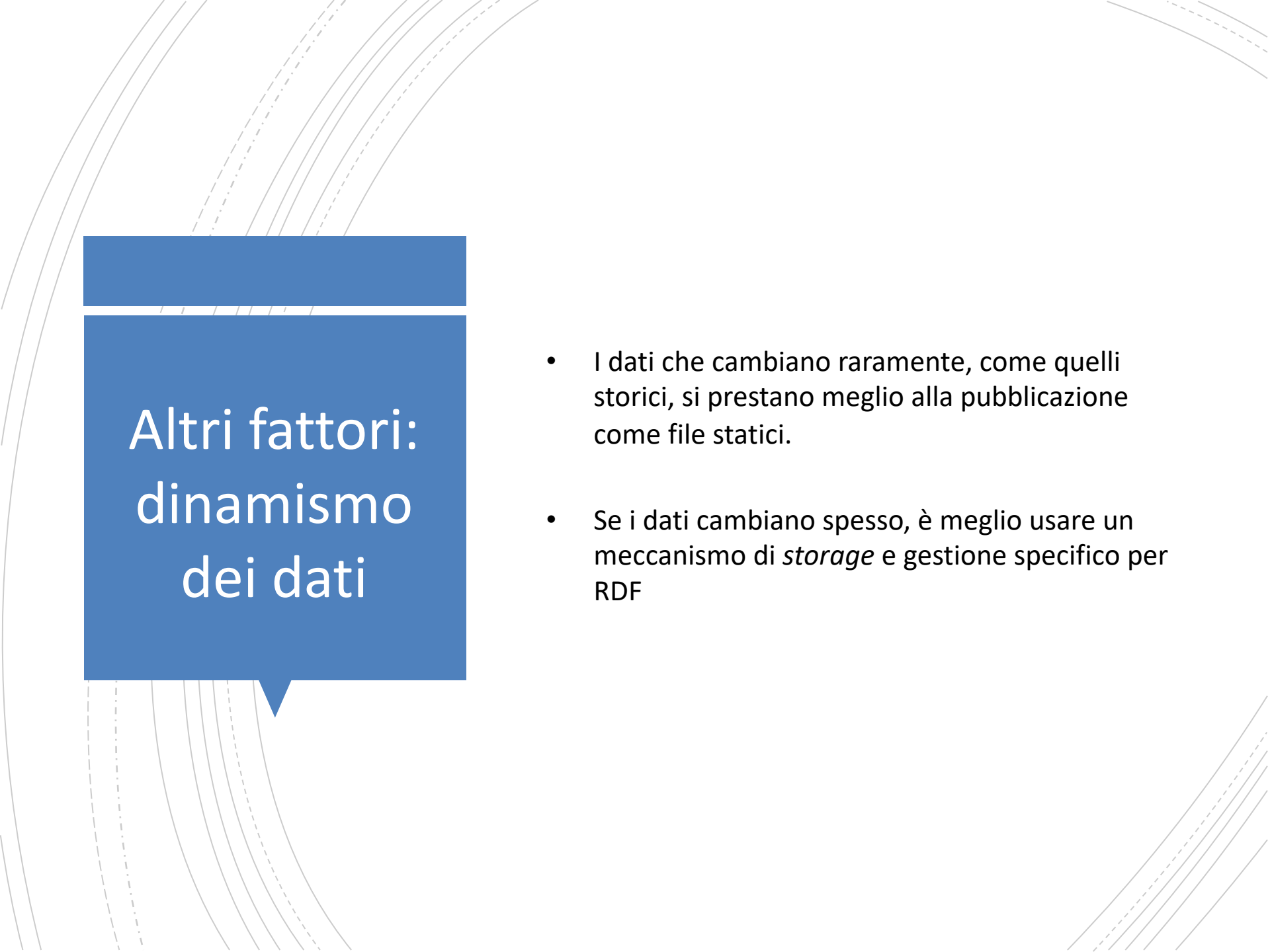


Tipi di dato (2)

- Dati Statici
- Se i dati statici sono già in formato RDF e seguono i principi dei Linked Data:
 - Possono essere pubblicati sul Web con un normale server
 - Caricati in uno store RDF che abbia un'interfaccia per i Linked Data
- Documenti testuali: possono essere analizzati automaticamente via software
 - Linked Data entity extraction

Altri fattori: volume dei dati

- Volume dei dati:
 - Piccola quantità di dati (poche centinaia di triple RDF su una singola entità) → file RDF statico
 - Data sets grande che descrive entità multiple → dividerlo in file separati, uno per ogni entità.
 - Possono essere serviti come file statici o tramite interfacce per LD

The background of the slide features several thin, curved lines in shades of gray, some solid and some dashed, creating a sense of motion or data flow. A blue rectangular box with a speech bubble tail at the bottom is positioned on the left side, containing the title text.

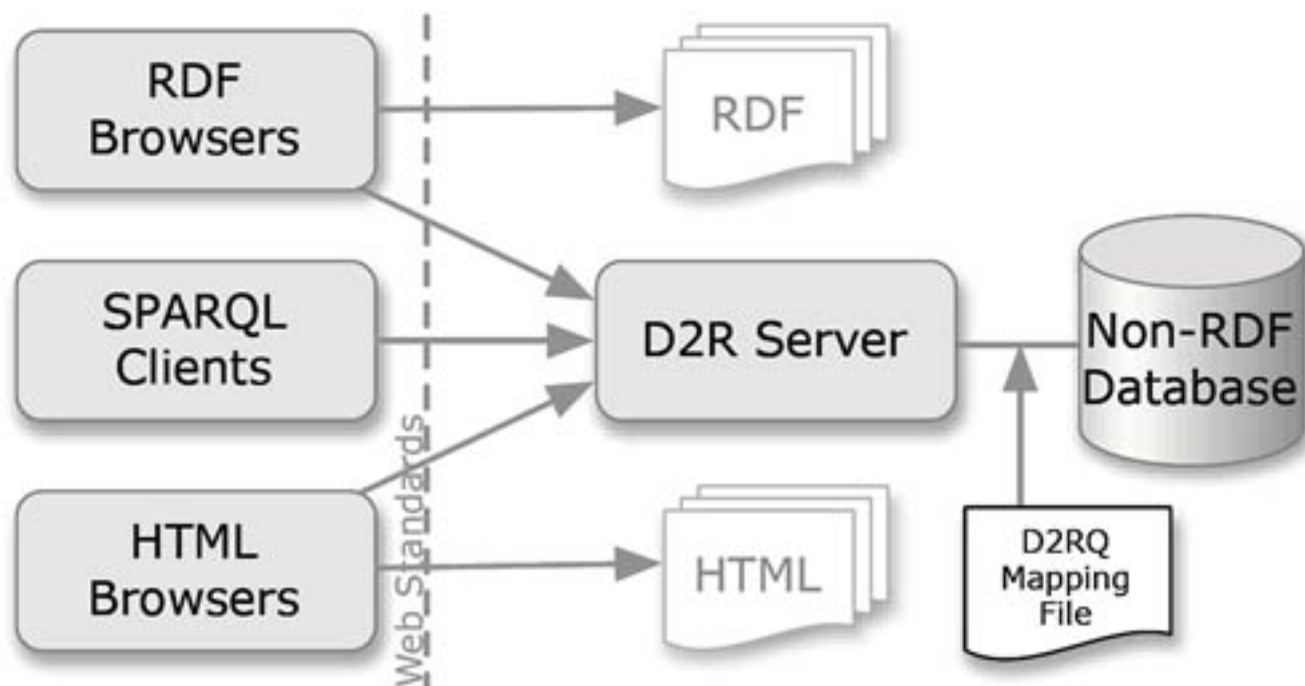
Altri fattori: dinamismo dei dati

- I dati che cambiano raramente, come quelli storici, si prestano meglio alla pubblicazione come file statici.
- Se i dati cambiano spesso, è meglio usare un meccanismo di *storage* e gestione specifico per RDF

Pubblicazione manuale (1)

- Produrre un file RDF statico e caricarlo su un server è probabilmente il modo più semplice per pubblicare Linked Data
- Normale quando
 - Una persona crea e mantiene file RDF di dimensioni contenute manualmente
 - per esempio per pubblicare un vocabolario
 - Un software genera o esporta i dati come file statici RDF
 - Singole schede d'archivio
- Ricordarsi di configurare il tipo MIME per xml/rdf sul server

Pubblicazione da un DB relazionale



QUADRO GENERALE

Software DR2

(anche: Virtuoso)

- I *wrapper* sono in grado di generare un mapping automatico da sql a rdf, che può essere poi corretto
1. Il Server D2R auto-genera un mapping D2RQ (D2RQ mapping file) a partire dallo schema del database.
 2. Il mapping può essere corretto manualmente, per esempio rimpiazzando i termini autogenerati con termini che derivano da vocabolari noti e pubblicamente accessibili

“D2R Server is a tool for publishing relational databases on the Semantic Web. It enables RDF and HTML browsers to navigate the content of the database, and allows querying the database using the SPARQL query language”

Attenzione, questo metodo generico non assume un formato RDF di arrivo!

Pubblicazione via API

- Lo fanno le società grandi, pubbliche e private: Amazon, Flickr, ecc.
1. They assign HTTP URIs to the resources about which the API provides data.
 2. When one of these URIs is dereferenced asking for application/rdf+xml, the wrapper rewrites the client's request into a request against the underlying API.
 3. The results of the API request are transformed to RDF and sent back to the client.

Il quadro completo

Application Layer

Application Code

SPARQL or RDF API

Data Access,
Integration and
Storage Layer

Web Data
Access
Module

Vocabulary
Mapping
Module

Identity
Resolution
Module

Quality
Evaluation
Module

Integrated
Web Data

HTTP

Web of Data

Publication Layer

LD Wrapper

Database A

HTTP

LD Wrapper

Database B

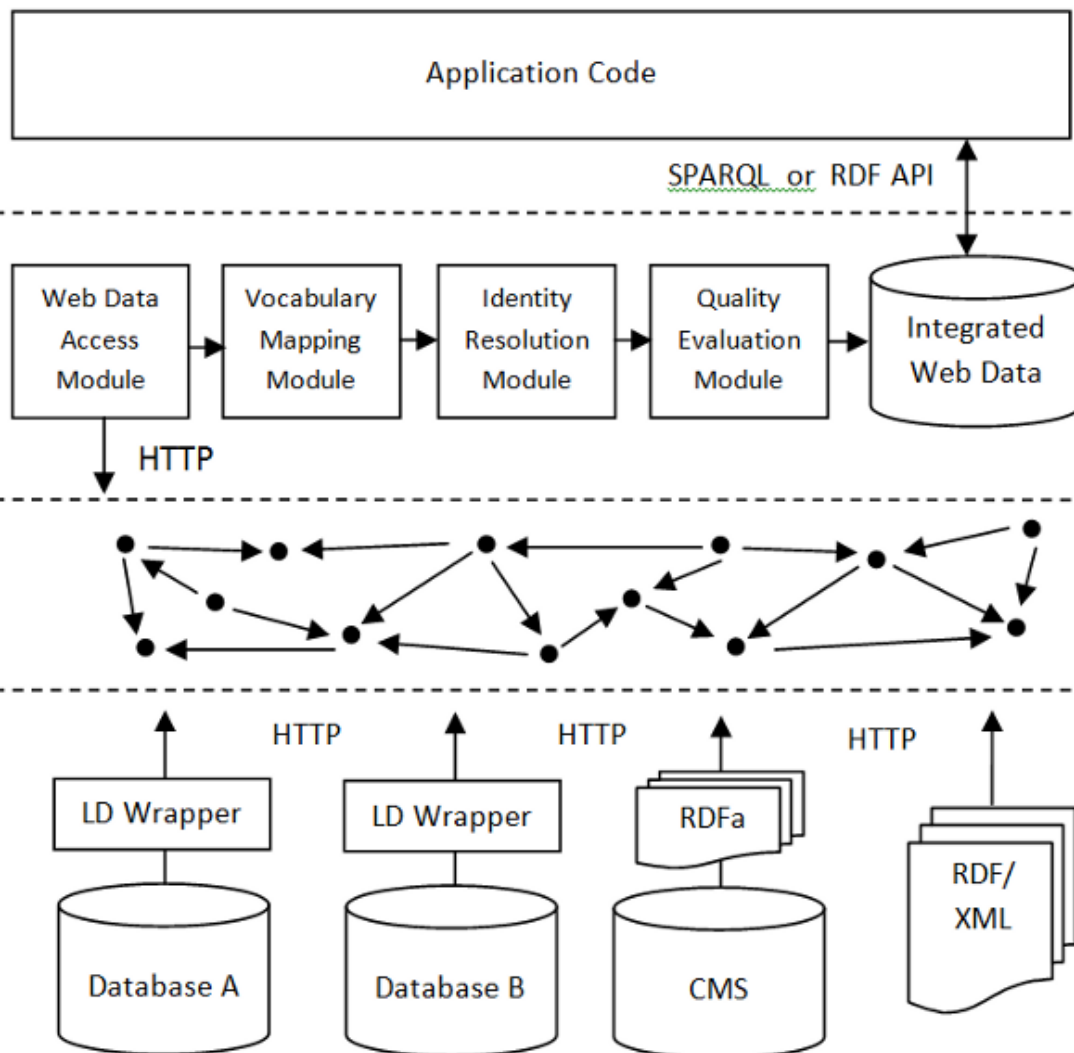
HTTP

RDFa

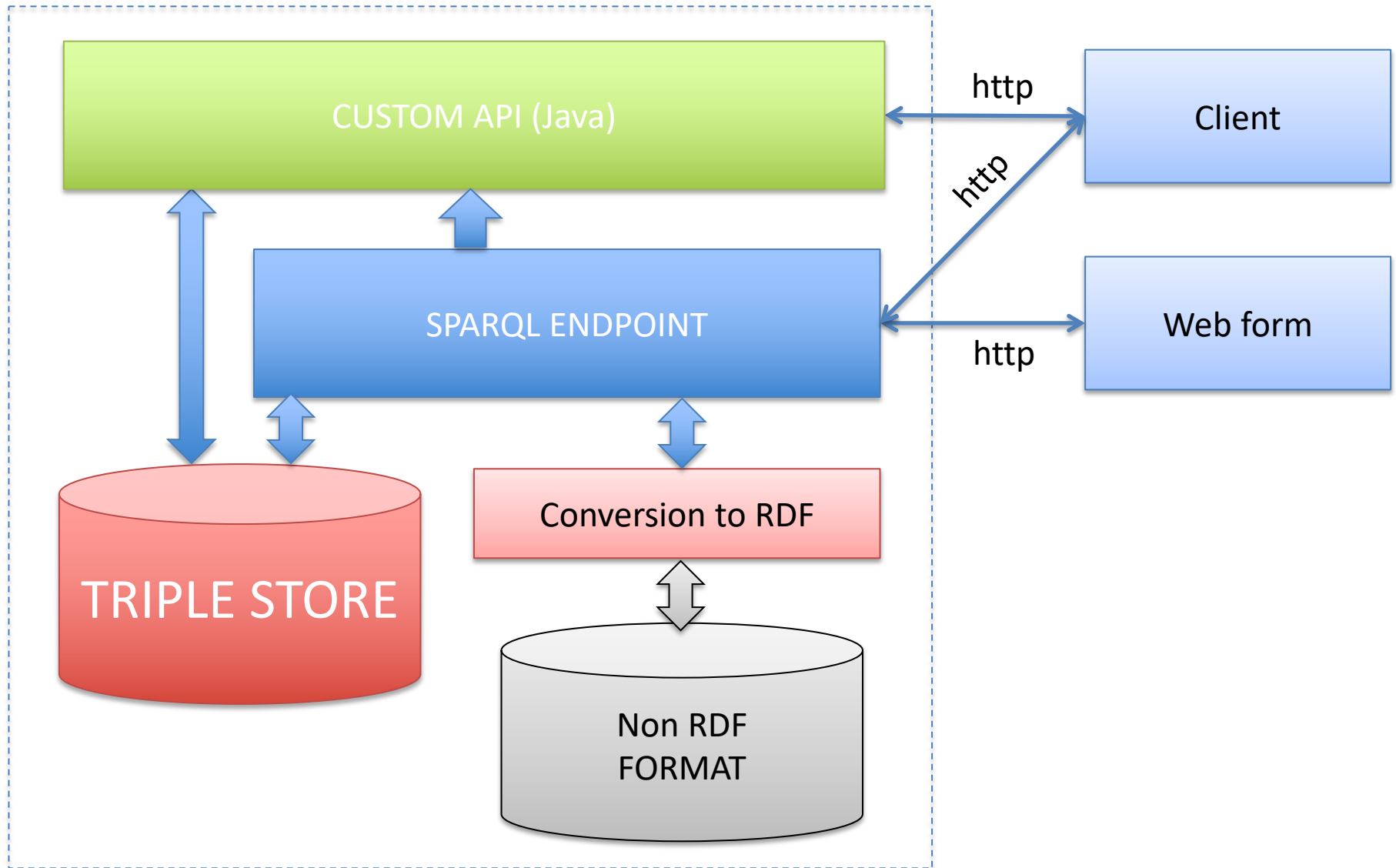
CMS

HTTP

RDF/
XML



LINKED DATA PLATFORM



RDF4J (ex SESAME)

- RDF4J è un framework open source, scritto in Java, per *processare* i dati RDF.
 - <http://rdf4j.org>
- Funzionalità
 - Parsing
 - Storing
 - Inferencing
 - Querying
- Può essere usato con i suoi database RDF o con altre soluzioni di storage delle triple.
- Incorporato in Graph-DB

JENA

- Open source Semantic Web framework for Java.
 - <https://jena.apache.org/index.html>
- Fornisce
 - Una API per leggere e scrivere dati in RDF. I grafi sono rappresentati come “abstract model” (possono essere alimentati con data from file, database, ecc).
 - Interrogabile via SPARQL
 - Aggiornabile via SPARUL (update language)
- Simile a Sesame; tuttavia, a differenza di Sesame, Jena fornisce supporto per OWL.
- Ha vari reasoner interni, altri possono essere installati, per esempio Pellet (un reasoner open source per OWL) oppure Hermit.

VIRTUOSO

- Software open-source rilasciato con licenza GPL da OpenLink
- Cross-platform
- Triplestore, RDBMS, Application server, Web server
- Object-relational database engine for (SQL, XML, RDF and plain text)
- Supports SQL and SPARQL
(<https://virtuoso.openlinksw.com/sparql-tutorials/>)
- <https://virtuoso.openlinksw.com>