RDF

Mario Arrigoni Neri

Le generazioni del WEB

- Internet fase 1: contenuti statici
  - Pagine HTML
  - Risorse FTP
  - L'utente sa cosa vuole e dove recuperarlo
- Internet fase 2: applicazioni web
  - Personalizzazione del livello presentation.. VB-Script, J-Script, DHTML
  - Contenuti dinamici (Servlet, JSP, ASP, Applet, …)
  - L'applicazione interagisce con l'utente
- Internet fase 3: semantic web
  - Documenti "leggibili" da agenti artificiali
  - Servizi utilizzabili da agenti artificiali

La semantica dei documenti

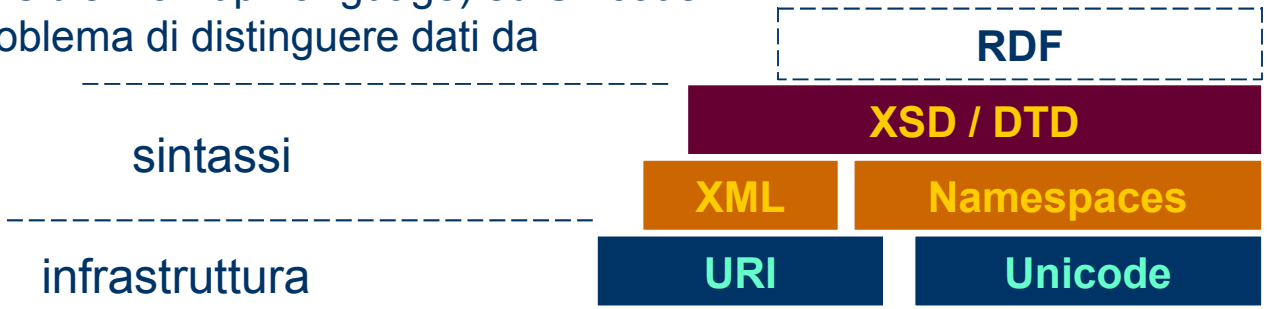
- Documenti non semplicemente destinati alla lettura, ma processabili da sistemi artificiali
- Nei documenti l'informazione circa il contenuto è "nascosta" nel testo. Serve un lettore umano per estrarla
- Serve una sovrastruttura semantica che catturi il significato delle risorse in un formalismo processabile dalla macchina
- Nel web semantico sostituisco (affianco) al documento human-readable una serie di affermazioni (o asserzioni) che ne rappresentano gli aspetti semantici di interesse

Il web semantico

- I metadati, cioè dati sui dati, descrivono quegli aspetti delle risorse che mi interessa rendere processabili in maniera automatica.
- Rappresentano conoscenza in quanto informazione utilizzata dal sistema durante l'elaborazione
- Metadati interni al documento
  - Es: i tag che inserisco in un documento di testo per descrivere particolari proprietà dello stesso
  - Uso procedurale: "questo testo è in grassetto"
  - Uso informativo: "estrai i titoli di tutti i capitoli"
- Metadati esterni
  - Costruisco un descrittore esterno
  - Distinguo il contenuto dalla descrizione del contenuto
  - Tipicamente guadagno in potenza espressiva
  - Posso standardizzare il linguaggio rispetto al formato del contenuto

Supporti di base al semWeb

- Prima di porsi il problema di come rappresentare la conoscenza occorre risolvere due problemi:
  - Come riferirsi agli oggetti del nostro modello formale ?
    Anche se i meta sono interni la descrizione può richiedere di far riferimento ad oggetti esterni
  - Come garantire l'interoperabilità per lo meno sintattica ?
- Le risposte a queste due richieste sono:
  - URI (Uniform Resource Identifiers) ed URIref
  - XML (eXtensible Markup Language) ed Unicode risolve il problema di distinguere dati da meta-dati



URL o URI ?

- Il dominio è più vasto rispetto ai documenti accessibili sulla rete
  - Es: il documento ha un autore, ma questo non è a sua volta una risorsa internet
- In generale utilizzo URI: stringhe identificative univoche per:
  - Risorse sulla rete: URL (Uniform Resource Locator):
    http://www.elet.polimi.it/index.jsp
    http://www.elet.polimi.it/upload/colombet/IC_2004/rdf.ppt
    ftp://ftp.server.it/file.txt
  - Risorse fisiche non accessibili sulla rete:
    http://www.elet.polimi.it/people/D02005
    http://www.elet.polimi.it/bib/book0001
  - Concetti astratti che non esistono né fisicamente né come risorse di rete:
    http://pur1.org/dc/elements/1.1/creator
    http://www.elet.polimi.it/terms/Book

URI reference

- E’ il caso più generale di URI
- E’ un URI seguito opzionalmente da un identificatore di elemento (fragment identifier)
- Esempio: http://www.server.com/index.jsp#title1
- Già noto ed utilizzato in HTML per riferirsi a particolari elementi all’interno del documento
- In RDF spesso si usa:
  - URI per identificare il documento RDF di specifica
  - Fragment identifier per riferirsi alla specifica di una particolare risorsa

Il modello RDF – 1

- Per esprimere una affermazione dobbiamo identificare:
  - L'oggetto che vogliamo descrivere
  - La specifica **proprietà** dell'oggetto (o **relazione** tra oggetti) su cui vogliamo predicare
  - Il **valore** assunto dalla proprietà o **l'oggetto** con cui viene messa in relazione l'entità su cui stiamo predicando

http://www.elet.polimi.it/bib/book0001 ha un autore il cui valore è Mario Arrigoni Neri

- Le tre componenti di una affermazione RDF prendono i nomi di:
  - Soggetto
  - Predicato
  - Oggetto

Il modello RDF – 2

- Risorse: tutto ciò che può essere è rappresentato da un URIref.
- Letterali: stringhe di testo
- Proprietà: sono gli attributi che voglio associare ad una risorsa. Corrispondono alle coppie:
  - Attributo – valore
  - Relazione – filler
  - Predicato – oggetto
- Asserzioni: sono gli elementi fondamentali di una descrizione RDF:
  - Tripla soggetto – predicato – oggetto
  - Coppia risorsa – proprietà
  - I letterali possono essere solo oggetti in una asserzione

Soggetto	http://www.elet.polimi.it/bib/book0001
Predicato	autore
Oggetto	“Mario Arrigoni Neri”

Rappresentazione

- Per disambiguare i termini del discorso si utilizzano URIref anche per rappresentare le relazioni (o predicati)

Soggetto	http://www.elet.polimi.it/bib/book0001
Predicato	http://www.elet.poli.it/terms/author
Oggetto	Mario Arrigoni Neri

- RDF fornisce uno schema logico, NON un linguaggio.
- Esistono linguaggi differenti per la rappresentazione
  - **N3**: formato “originale”, esprime le asserzioni come triple soggetto – predicato – oggetto
  - **Grafico**: le risorse sono nodi ed i predicati archi
  - **RDF/XML**: sintassi XML per descrivere modelli RDF (serializzazione)
- Diversi linguaggi per diversi scopi…

N3 – 1

- Gli URI vengono indicati tra parentesi angolari “<“ e “>”
- I letterali vengono racchiusi tra virgolette

<http://www.elet.polimi.it/bib/book0001> <http://www.elet.polimi.it/terms/author> “Mario Arrigoni Neri”

- L’utilizzo dell’URI anche per il predicato permette all’applicazione di “capire” che si sta utilizzando proprio il concetto di “autore” definito, ad esempio, in www.elet.polimi.it/terms

N3 – 2

- L'utilizzo della notazione a triple richiede che gli URIref siano scritti completamente
- Per semplicità si introducono nomi qualificati (QName) come in XML
  - Prefisso : nomeNonQualificato
  - Ad ogni prefisso viene associato un namespace
  - L'URI viene costruito giustapponendo il namespace ed il nome non qualificato

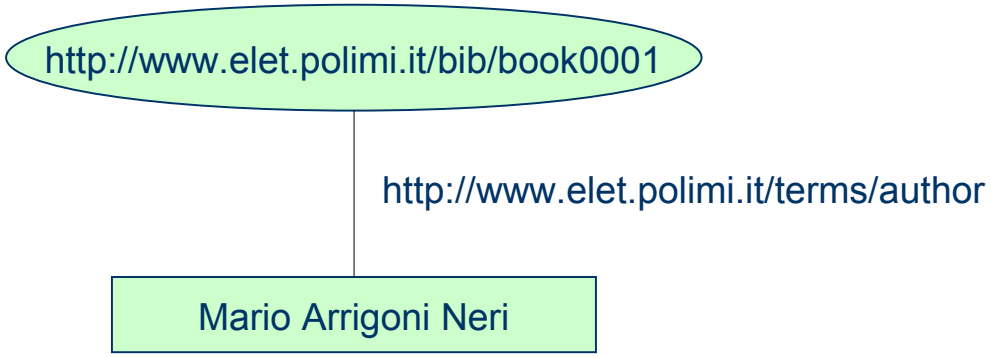
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
terms	http://www.elet.polimi.it/terms/
bib	http://www.elet.polimi.it/bib/

bib:book0001	terms:author	"Mario Arrigoni Neri"

<http://www.elet.polimi.it/bib/book0001>

Grafico

- La rappresentazione grafica è pensata principalmente per un utente umano
- Risorse rappresentate da ovali
- Letterali rappresentati da rettangoli
- Predicati rappresentati da frecce che collegano risorse a risorse o risorse a letterali



Risorse con più relazioni

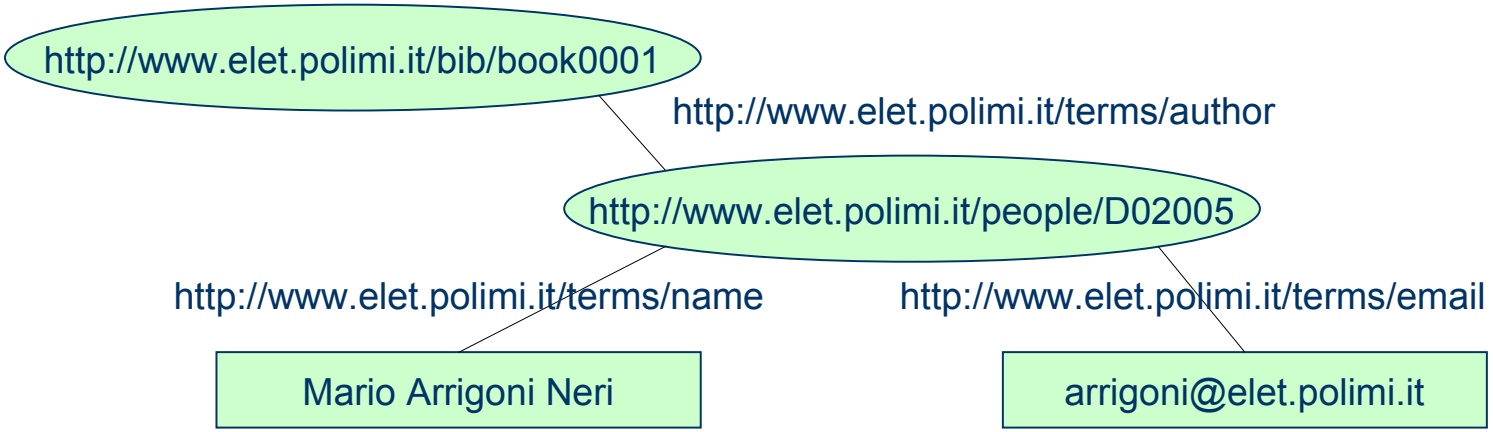
- Ovviamente ogni risorsa può essere soggetto e/o oggetto di più asserzioni
- In N3 la risorsa viene ripetuta per ogni asserzione
- Nel grafico il nodo si indica una sola volta, ma si indicano le diverse asserzioni tramite frecce diverse… anche se …

bib:book0001	terms:author	“Mario Arrigoni Neri”
bib:book0001	terms:author	“Marco Colombetti”

Vantaggi degli URIref – 1

- Descrivere l'autore con un letterale significa identificare una persona con il suo nome.
- Non ha molto senso dire che l'autore è la stringa "Mario Arrigoni Neri"
- Come gestire i casi di omonimia?
- E se voglio fare delle asserzioni sulla stringa in quanto tale?
- Costruisco un URIref che identifica univocamente la risorsa
- Tramite l'URIref l'oggetto diventa a sua volta una risorsa su cui predicare descrivendone le proprietà, ad esempio il nome, mediante ulteriori asserzioni
- Utilizzando gli URIref come soggetti, predicati ed oggetti si promuove la costruzione di un vocabolario per le asserzioni sul web
- Ovviamente la macchina non può veramente "capire" la semantica delle affermazioni, però può trattarle in maniera da far sembrare che ne capisca la semantica

Vantaggi degli URIref – 2

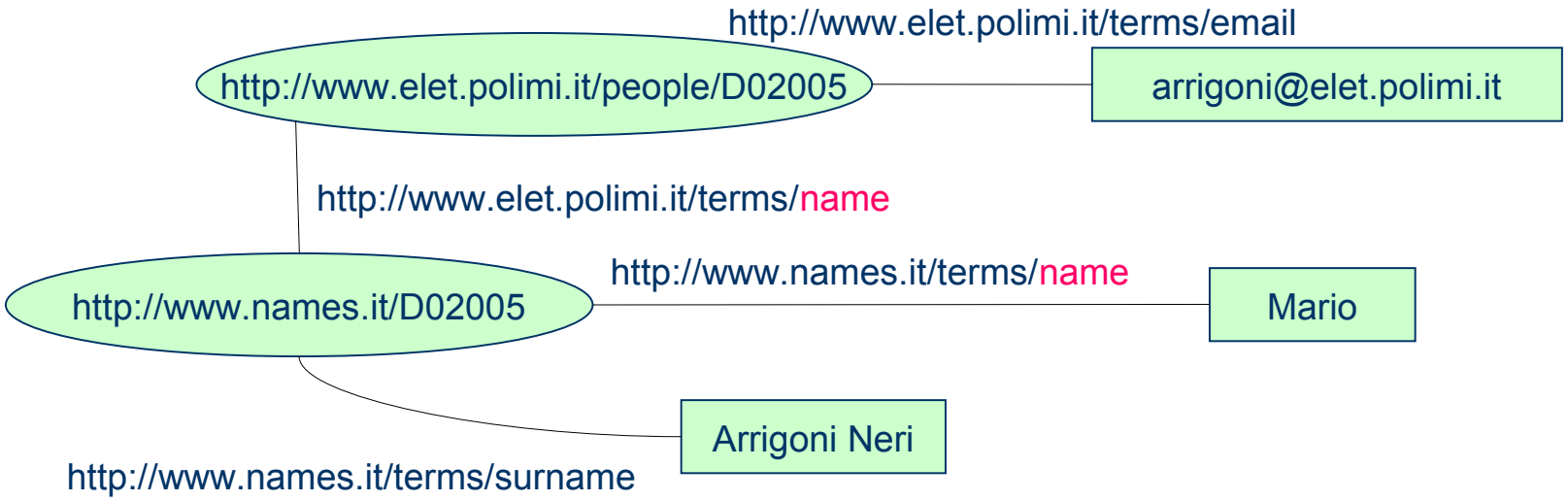


people	http://www.elet.polimi.it/people/

bib:book0001	terms:author	people:D02005
people:D02005	terms:name	“Mario Arrigoni Neri”
people:D02005	terms:email	“arrigoni@elet.polimi.it”

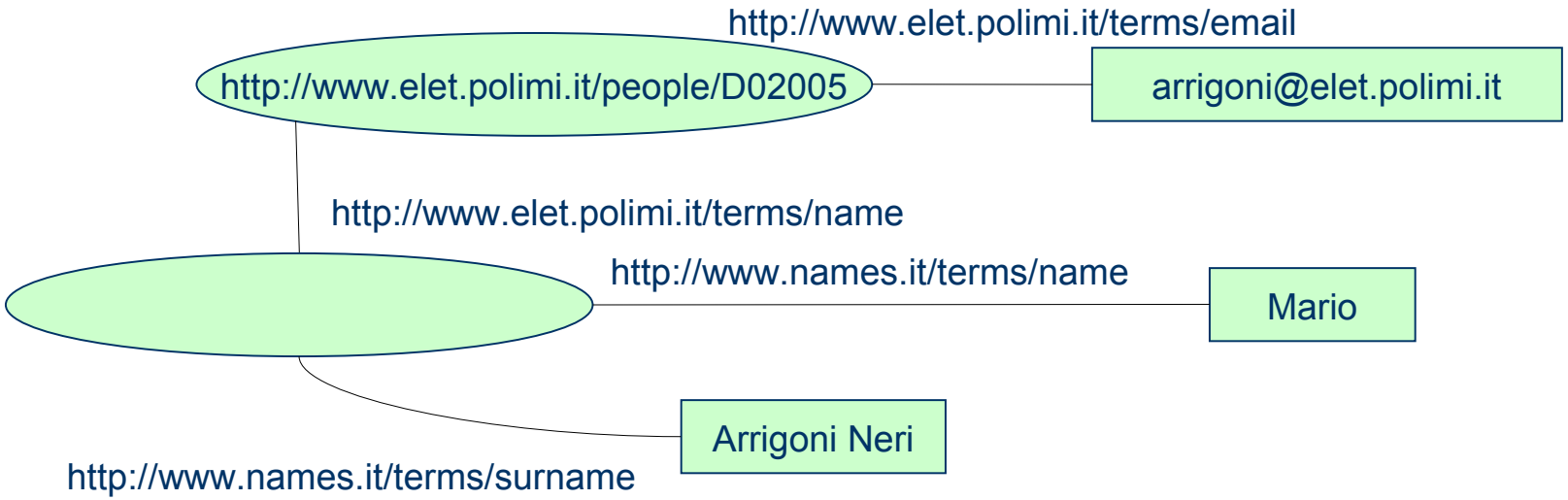
Proprietà strutturate

- Nel mondo reale esistono relazioni più complesse e tipicamente non binarie
- Es: il nome in generale può essere scomposto in elementi più semplici come nome, cognome
- Occorre creare una risorsa apposita per rappresentare l'aggregato



Risorse anonime – 1

- La risorsa names:D02005 viene utilizzata solo come aggregato del nome e del cognome
- In un modello complesso diventa necessario utilizzare un gran numero di nodi intermedi
- Posso usare dei nodi vuoti



Risorse anonime – 2

- I nodi vuoti sono la rappresentazione grafica delle risorse anonime
- Non vengono mai accedute dall'esterno del file

bib:book0001	terms:author	people:D02005
people:D02005	terms:email	"arrigoni@elet.polimi.it"
people:D02005	terms:name	???
???	names:name	"Mario"
???	names:surname	"Arrigoni Neri"

- Tuttavia è necessario costruire degli identificatori "locali" per riferirsi alla particolare risorsa anonima all'interno della stessa specifica

bib:book0001	terms:author	people:D02005
people:D02005	terms:email	"arrigoni@elet.polimi.it"
people:D02005	terms:name	_:D02005Name
_:D02005Name	names:name	"Mario"
_:D02005Name	names:surname	"Arrigoni Neri"

Risorse anonime – 3

- Le risorse anonime sono utili per fare riferimento all'entità di interesse e non ad un suo particolare attributo.
- Questo rimane vero anche quando l'attributo è una chiave
- Es: potrei usare l'e-mail come "alias" di una persona
- Però se poi voglio esprimere un'asserzione sull'indirizzo (ad esempio che è irraggiungibile)?

Letterali tipizzati – 1

- Il parser può avere necessità di estrarre l'informazioni dalle componenti letterali associate ad una risorsa
- In generale in RDF è possibile associare un tipo a ciascun letterale

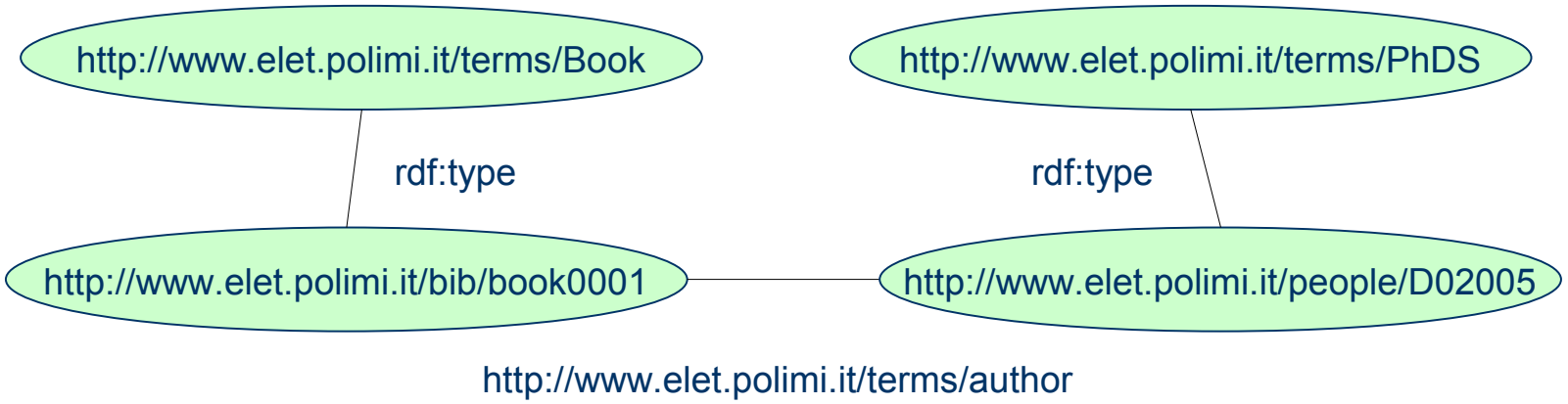
bib:book0001	terms:pubDate	"2004-01-01"^^xsd:date

Letterali tipizzati – 2

- RDF non definisce tipi built-in
- RDF semplicemente fornisce un metodo per associare un generico tipo ad ogni letterale
- L'interpretazione del tipo viene lasciato all'applicazione
- La libreria di tipi di più vasto utilizzo è quella definita dall'XSD
- I datatype XSD non sono tuttavia "speciali", ma sono gestiti esattamente come qualsiasi altro tipo che possa essere definito dall'utente

Risorse tipizzate

- Per descrivere compiutamente una risorsa occorre specificare la categoria di risorse a cui appartiene
- Programmazione: classe, Logica: concetto
- Ad esempio, se il mio agente vuole cercare un libro deve poter sapere che http://www.elet.polimi.it/bib/book0001 rappresenta un libro
- In RDF l'associazione tra istanza e tipo è rappresentata dalla particolare relazione *rdf:type*



RDF / XML

- XML si propone naturalmente come supporto sintattico per la descrizione delle risorse
- L'elemento radice è *rdf:RDF*
- Ogni asserzione può essere rappresentata da un elemento *Description*

```
<?xml version="1.0"?>
<rdf:RDF   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:terms = "http://www.elet.polimi.it/terms/"
           xmlns:bib = "http://www.elet.polimi.it/bib/"
           xmlns:people="http://www.elet.polimi.it/people/">

           <rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
                      <terms:author>Mario Arrigoni Neri</terms:author>
           </rdf:Description>
</rdf:RDF>
```

Gruppi di asserzioni

- Un documento RDF in formato RDF/XML è composto da una sequenza di elementi Description

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author>Mario Arrigoni Neri</terms:author>
</rdf:Description>
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author>Marco Colombetti</terms:author>
</rdf:Description>
```

- Le asserzioni riguardanti uno stesso soggetto possono essere raggruppate

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author>Mario Arrigoni Neri</terms:author>
        <terms:author>Marco Colombetti</terms:author>
</rdf:Description>
```

Riferimenti e risorse interne

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author rdf:resource="http://www.elet.polimi.it/people/D02005"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name>Mario Arrigoni Neri</terms:name>
</rdf:Description>
```

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author>
          <rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
                <terms:name>Mario Arrigoni Neri</terms:name>
          </rdf:Description>
        </terms:author>
</rdf:Description>
```

Risorse anonime

- Una risorsa anonima può essere rappresentata da un elemento Description con l'attributo rdf:ID in luogo di rdf:about

```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:ID="D02005Name"/>
</rdf:Description>
<rdf:Description rdf:ID="D02005Name">
        <names:name>Mario</names:name>
        <names:surname>Arrigoni Neri</names:surname>
</rdf:Description>
```

- In realtà non è proprio così (lo vedremo dopo), ma è una tecnica piuttosto comune

```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name><rdf:Description>
                <names:name>Mario</names:name>
                <names:surname>Arrigoni Neri</names:surname>
        </rdf:Description></terms:name>
</rdf:Description>
```

Tipizzazione

- Tipizzazione dei letterali

```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:ID="D02005Name"/>
        <terms:age rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
                28</terms:age>
</rdf:Description>
```

- Tipizzazione delle risorse

```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:ID="D02005Name"/>
        <terms:age rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
                28</terms:age>
        <rdf:type rdf:resource="http://www.elet.polimi.it/terms/PhDS"/>
</rdf:Description>
```

Abbreviazioni – 1

- Gli attributi con valore letterale e che non si ripetono possono essere espressi tramite attributi anzichè tramite sottoelementi

```
<rdf:Description rdf:ID="D02005Name"  names:name="Mario">
        <names:surname>Arrigoni Neri</names:surname>
</rdf:Description>
```

- E' possibile eliminare le descrizioni interne indicando nel predicato che l'oggetto (il filler) deve essere considerato una risorsa (ANONIMA)

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <terms:author rdf:parsetype="resource">
                <terms:name rdf:resource="http://www.elet.polimi.it/people/D02005Name"/>
                <terms:age rdf:datatype="..">28</terms:age>
        </terms:author>
</rdf:Description>
```

Abbreviazioni – 2

- La (una) relazione rdf:type può essere resa in maniera più compatta sostituendo l'elemento rdf:Description con il tipo a cui appartiene la risorsa

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
        <rdf:type rdf:resource="http://www.elet.polimi.it/terms/Book"/>
        …
</rdf:Description>
```

```
<terms:Book rdf:about="http://www.elet.polimi.it/bib/book0001">
        …
</terms.Book>
```

- Vantaggio: posso utilizzare i prefissi invece dei namespace

QNames ed attributi – 1

- Attenzione: i Qname sono ammessi solo negli elementi e negli attributi.
- RDF non garantisce di risolvere le qualificazioni all'interno del testo, ad esempio nei valori degli attributi

```
<rdf:Description rdf:about="bib:book0001">
        <rdf:type rdf:resource= "terms:Book"/>
        …
</rdf:Description>
```

QNames ed attributi – 2

- Posso però usare un trucco: introduco delle entità XML che vengono sostituite dal parse XML prima di passare a quello RDF

```
<!DOCTYPE uridef [
  <!ENTITY rdf          "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <!ENTITY terms        "http://www.elet.polimi.it/terms/">
  <!ENTITY bib          "http://www.elet.polimi.it/bib/">
]>
<rdf:RDF xmlns:rdf="&rdf;" xmlns:terms="&terms;" xmlns:bib="&bib,">
  <rdf:Description rdf:about="&bib;book0001">
    <rdf:type rdf:resource="&terms;Book"/>
                …
  </rdf:Description>
</rdf:RDF>
```

Gli identificativi univoci – 1

- L'attributo rdf:ID è molto simile all'ID di XML: costruisce un identificativo univoco all'interno del file
- L'identificativo viene costruito componendo l'URI di base (cioè l'indirizzo del documento corrente), il simbolo '#' ed usando il parametro rdf:ID come fragment identifier
- Quando si usa un riferimento tramite rdf:resource "#pippo" si riferisce all'URI di base attuale (come in HTML)
- Immaginando che la descrizione sia recuperabile su internet a: http://www.elet.polimi.it/bib/biblio.rdf

```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:resource="#D02005Name"/>
</rdf:Description>
<rdf:Description rdf:ID="D02005Name">
        <names:name>Mario</names:name>
        <names:surname>Arrigoni Neri</names:surname>
</rdf:Description>
```

Gli identificativi univoci – 2

- Da un file esterno per riferirmi al nodo dovrei usare l'URI completo

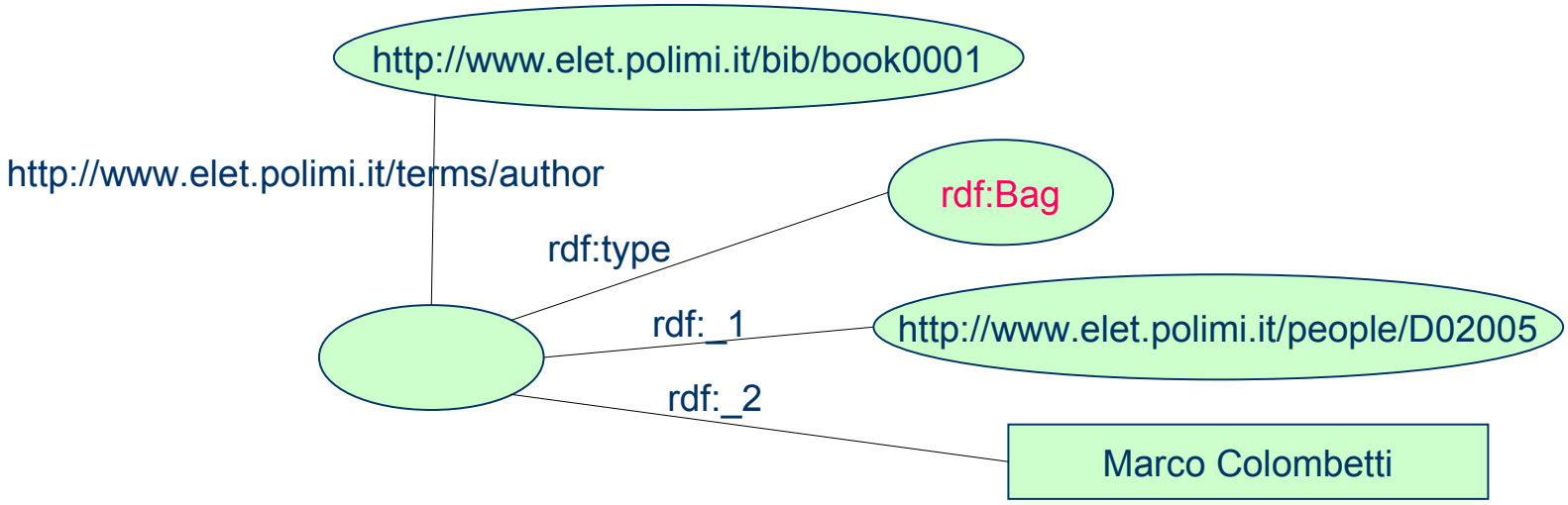
```
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:resource="http://www.elet.polimi.it/bib/biblio.rdf#D02005Name"/>
</rdf:Description>
```

- Problema: cosa succede se eseguo un mirror del file su un server differente? La risorsa "#D02005Name" non corrisponderebbe a quella originale

```
<rdf:RDF xml:base="http://www.elet.polimi.it/bib/biblio.rdf" …>
<rdf:Description rdf:about="http://www.elet.polimi.it/people/D02005">
        <terms:name rdf:resource="http://www.elet.polimi.it/bib/biblio.rdf#D02005Name"/>
</rdf:Description>
</rdf:RDF>
```

Contenitori

- Spesso è utile rappresentare una collezione di risorse invece che collegare ogni risorsa al medesimo soggetto.
- Es: voglio rappresentare l'insieme degli autori del libro



- Il container (di tipo rdf:Bag) rappresenta la collezione degli individui

Contenitori in RDF / XML

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
  <terms:author>
    <rdf:Description>
      <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
      <rdf:_1 rdf:resource="http://www.elet.polimi.it/people/D02005"/>
      <rdf:_2>Marco Colombetti</rdf:_2>
    </rdf:Description>
  </terms:author>
</rdf:Description>
```

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001">
  <terms:author>
    <rdf:Bag>
      <rdf:li rdf:resource="http://www.elet.polimi.it/people/D02005"/>
      <rdf:li>Marco Colombetti</rdf:li>
    </rdf:Bag>
  </terms:author>
</rdf:Description>
```

Alt e Seq

- Oltre al Bag RDF fornisce altri due costrutti per costruire collezioni
- rdf:Alt: indica che la relazione (predicato) può avere come filler una delle risorse che compongono l'aggregato

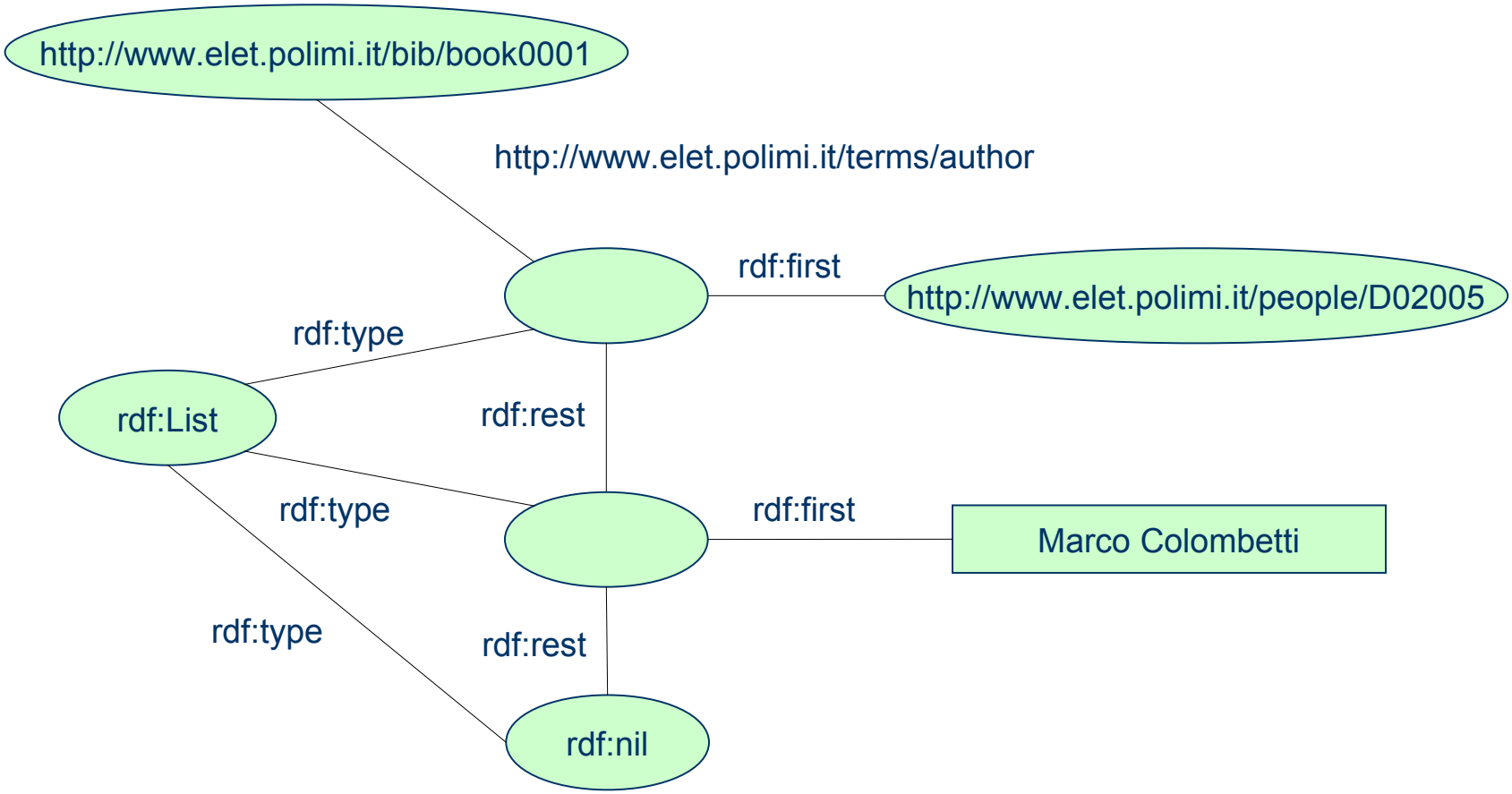
```
<rdf:Description rdf:about="http://www.elet.polimi.it/Application">
  <terms:mirror>
    <rdf:Alt>
      <rdf:li rdf:resource="http://www.example.com/Application"/>
      <rdf:li rdf:resource="http://www.example1.com/Application"/>
      <rdf:li rdf:resource="http://www.example2.com/Application"/>
    </rdf:Bag>
  </terms:mirror>
</rdf:Description>
```

- con l'alternativa l'ordine degli elementi rdf:_n può indicare l'ordine di preferenza
- rdf:Seq indica un insieme ordinato

Collezioni – 1

- RDF fa l'assunzione di mondo aperto, questo significa che chiunque può effettuare asserzioni su risorse definite ovunque. Il modello complessivo è dato dall'unione delle asserzioni
- Questo significa che usando i contenitori non vi è modo di elencare tutte e sole le componenti di un aggregato, evitando che in altri documenti si aggiungano elementi
- Per elencare i componenti si usa una struttura ricorsiva di lista simile a quella del lisp
- Il lessico per le collezioni è dato da:
  - il tipo rdf:List: è il tipo a cui appartiene ogni singolo nodo della lista
  - Il predicato rdf:first: collega ogni nodo della lista con il primo elemento
  - Il predicato rdf:rest: collega la testa della lista con la coda
  - La risorsa rdf:nil (di tipo rdf:List). Terminatore della lista

Collezioni – 2



WEB of trust

- RDF fa l'assunzione di mondo aperto.
- Posso sempre introdurre un elemento Description con un attributo rdf:about che lo associa ad una risorsa descritta altrove
- E se l'informazione introdotta da terze parti è scorretta?
  - Il sistema gestirà in maniera scorretta le risorse
- E se l'informazione è addirittura contraddittoria?
  - Sappiamo che in una logica contraddittoria tutte le formule sono teoremi
  - Questo significa che possiamo derivare qualunque cosa

$$A, \bot \models *$$

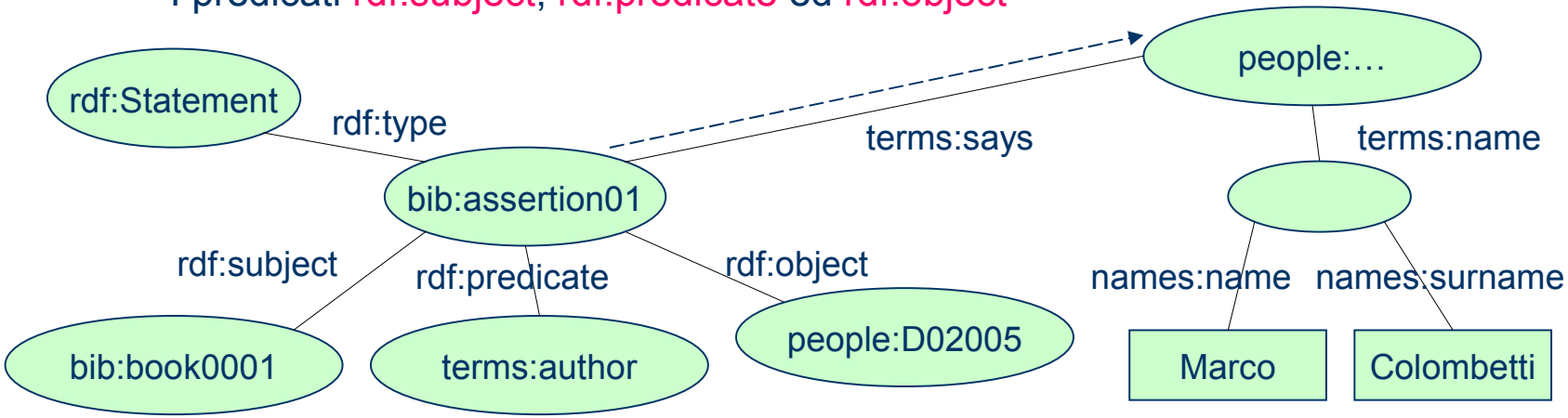
- Servono quindi dei criteri di certificazione dell'informazione
- Questo però richiede delle meta-asserzioni, cioè delle asserzioni che riguardano altre asserzioni

Reificazione – 1

- le meta-asserzioni sono meta-informazioni su altre meta-informazioni.
- Es: “Marco Colombetti afferma che Mario Arrigoni Neri è autore del testo”
- Una possibilità è costruire una affermazione con predicato “afferma”, soggetto “Marco Colombetti” e come oggetto l’affermazione “Mario Arrigoni Neri è autore del testo”
- Occorre quindi rileggere l’affermazione come una risorsa disponibile per la descrizione
- Questo processo prende il nome di “reificazione” (rendere come oggetto)
- E’ simile a quello che si fa con gli ER per tradurre in uno schema logico le relazioni molti-molti

Reificazione – 2

- L'affermazione è una relazione ternaria che associa soggetto, predicato ed oggetto
- Posso reificarla costruendo una risorsa che la identifichi e rendendo la relazione ternaria tramite tre relazioni binarie
- Il lessico della reificazione è composto da:
  - Il tipo rdf:Statement che "tipizza" le asserzioni
  - I predicati rdf:subject, rdf:predicate ed rdf:object



Sintassi compatta di reificazione

- L'attributo rdf:bagID identifica l'asserzione come risorsa per eseguire meta-asserzioni su di essa

```
<rdf:Description rdf:about="http://www.elet.polimi.it/bib/book0001"
                 bagID="assertion0001">
  <terms:author rdf:resource="http://www.elet.polimi.it/people/D02005"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.elet.polimi.it/people/...">
                 <terms:says rdf:resource="#assertion0001"/>
<rdf:Description>

<rdf:Description rdf:about="#assertion0001">
                 <terms:by rdf:resource="http://www.elet.polimi.it/people/..."/>
</rdf:Description>
```

- Attenzione: RDF NON identifica l'affermazione reificata con quella (eventualmente) presente nel modello