RDF

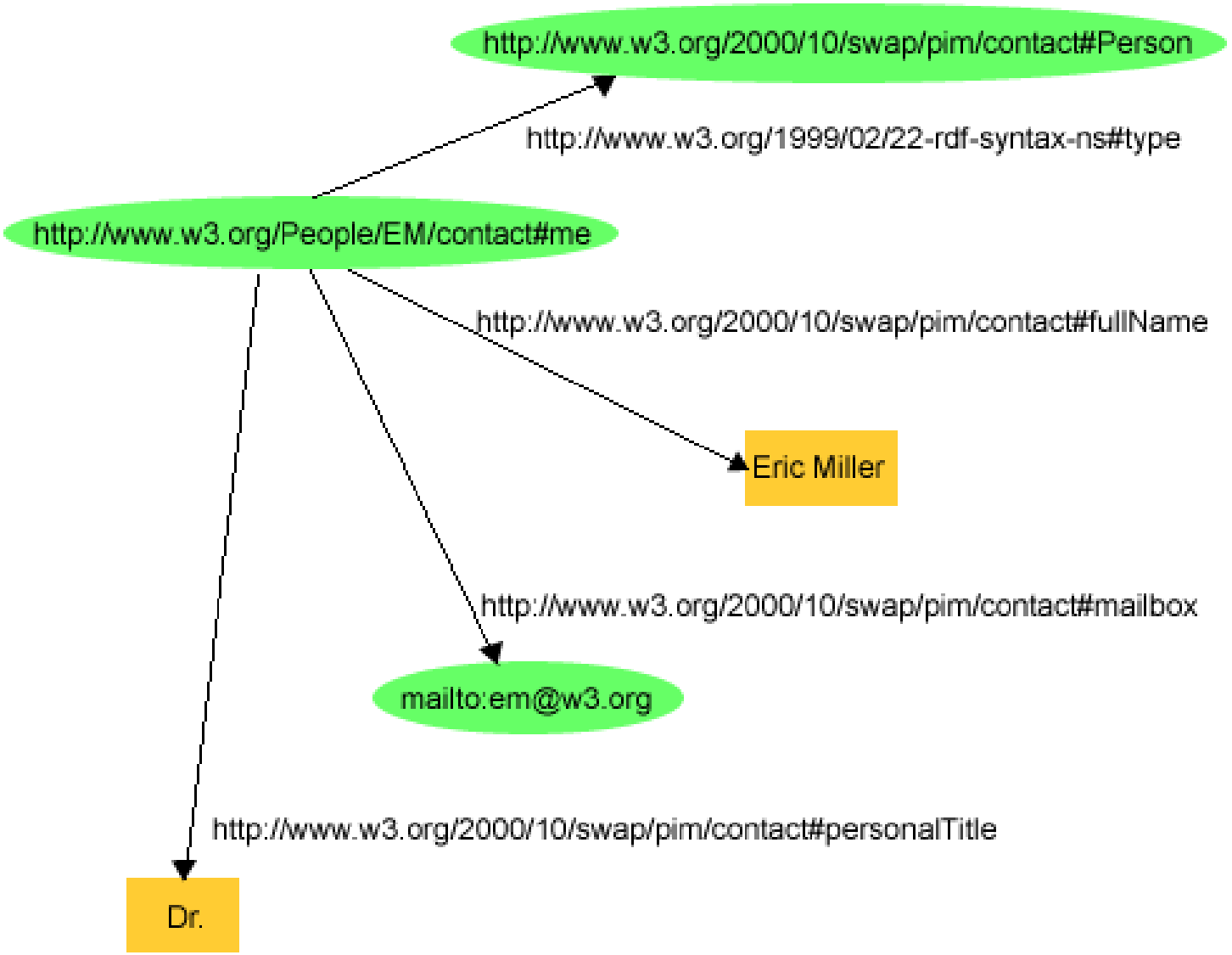
Aim

- Defined to represent “metadata” about web resources
- Useful to represent any kind of “knowledge base”
- Basic reference: http://www.w3.org/TR/rdf-primer/ READ THIS ONE
- http://www.w3.org/TR/rdf-concepts/
- http://www.w3.org/TR/rdf-syntax-grammar/
- http://www.w3.org/TR/rdf-mt/: semantics

The data model

- “Resources”, identified by an IRI (nodes)
  - IRI: internationalized RI
    - Used to be URI
- Subject-predicate-object triples:
  - Subject: a resource, identified by an IRI
  - Predicate: a property name, identified by an IRI
  - Object: the property value, either:
    - Another resource
    - A simple value
- A set of triples can be seen as a graph

Example


http://www.w3.org/2000/10/swap/pim/contact#Person
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://www.w3.org/People/EM/contact#me
http://www.w3.org/2000/10/swap/pim/contact#fullName
Eric Miller
http://www.w3.org/2000/10/swap/pim/contact#mailbox
mailto:em@w3.org
http://www.w3.org/2000/10/swap/pim/contact#personalTitle
Dr.

The same example as N-Triples

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/.../22-rdf-syntax-ns#type>
<http://www.w3.org/2000/10/swap/pim/contact#Person>.

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#fullName>
"Henry Miller".

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#mailbox>
<mailto:em@w3.org>.

<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#personalTitle> "Dr.".

N-Triples

<http://example.org/bob#me>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://xmlns.com/foaf/0.1/Person> .

<http://example.org/bob#me>
<http://xmlns.com/foaf/0.1/knows>
<http://example.org/alice#me> .

<http://example.org/bob#me>
<http://schema.org/birthDate>
"1990-07-04"^^<http://www.w3.org/2001/XMLSchema#date> .

<http://example.org/bob#me> <http://xmlns.com/foaf/0.1/topic_interest>
<http://www.wikidata.org/entity/Q12418> .

<http://www.wikidata.org/entity/Q12418> <http://purl.org/dc/terms/title>
"Mona Lisa" .

<http://www.wikidata.org/entity/Q12418> <http://purl.org/dc/terms/creator>
<http://dbpedia.org/resource/Leonardo_da_Vinci> .

Turtle

```
BASE   <http://example.org/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX schema: <http://schema.org/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX wd: <http://www.wikidata.org/entity/>

<bob#me>
        a foaf:Person ;
        foaf:knows <alice#me> ;
        schema:birthDate "1990-07-04"^^xsd:date ;
        foaf:topic_interest wd:Q12418 .

wd:Q12418
        dcterms:title "Mona Lisa" ;
        dcterms:creator <http://dbpedia.org/resource/Leonardo_da_Vinci> .
```

XML syntax for RDF: less used now

```
<rdf:RDF xmlns:rdf="http//www.w3.org/.../22-rdf-syntax-ns#“
  xmlns:contact="http://www.w3.org/2000/.../pim/contact#">
  <contact:Person
          rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>
</rdf:RDF>
```

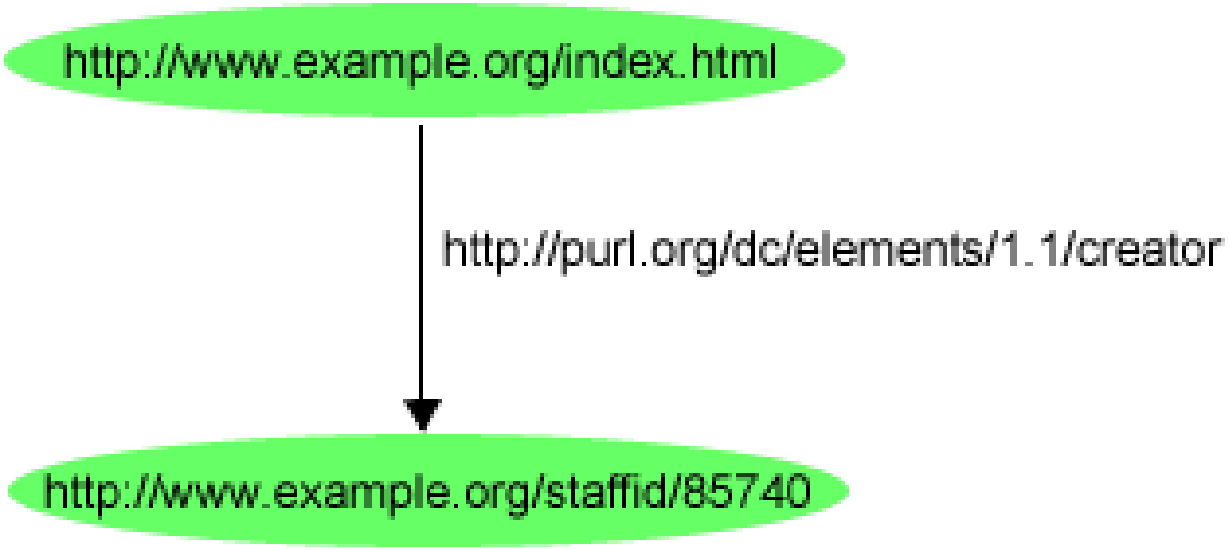
```
<rdf:RDF xmlns:...>
  <tipo rdf:about=soggetto>
    <predicato>valoreOggetto</predicato>
    ...
    <predicato rdf:resource="IRIoggetto"></predicato>
  </tipo>
</rdf:RDF>
```

RDF

- Very simple model: resource-property-value
- Properties are chosen from an extensible vocabulary
- The rdf:type property
- Simple values can be typed

IRIs

- URI: identifies an entity through ASCII chars
- URI: ***scheme***:[//***authority***][/.../.../...][?...][#...]
- IRI: international URI
- RDF uses IRIs to identify both entities and properties



IRI’s

- IRI’s and URL: even if a IRI may look like an URL, it does not necessarily refers to a web page
- Uniqueness of IRI’s: every organization uses a name space that it owns: “http://www.di.unipi.it/teaching/bd2#”

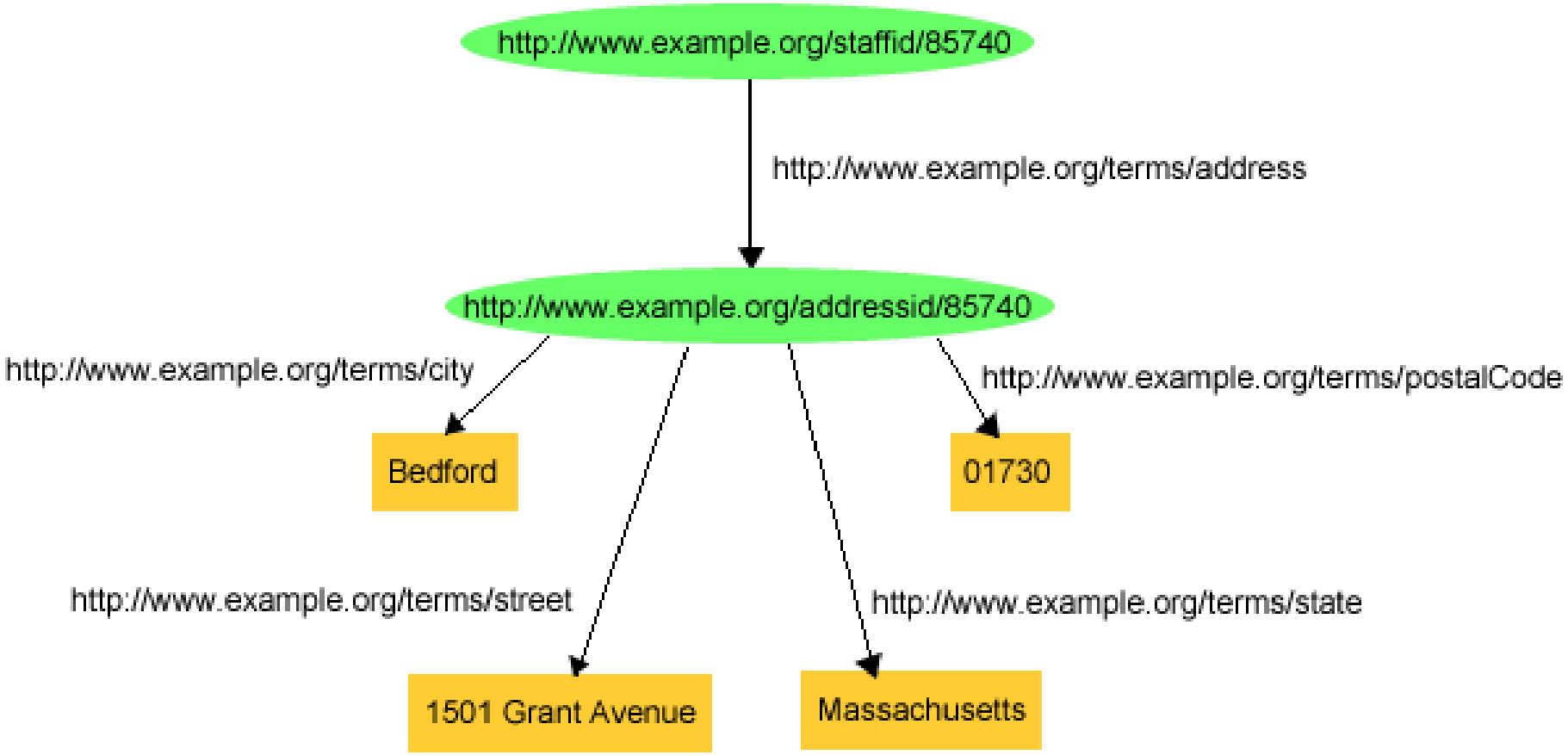
Properties of IRI's

- Are visible, can be transmitted as strings – by computer, or by voice…
- Avoiding conflicts
- Sharing the meaning of terms:
  - dc:creator

Constants (literals)

- Literals can only be used as objecs in s-p-o triples
- They are usually typed "literal"^^type
- RDF has no specific type system, but a foreign type system (xsd, typically) can be used, if its types are identified by IRIs
- Strings may be written with no type, and number with neither quotes nor the type
- Structured values must be represented through artificial nodes

An address


http://www.example.org/staffid/85740
http://www.example.org/terms/address
http://www.example.org/addressid/85740
http://www.example.org/terms/city
http://www.example.org/terms/postalCode
Bedford
01730
http://www.example.org/terms/street
http://www.example.org/terms/state
1501 Grant Avenue
Massachusetts

Anonymous nodes

- A node which needs not to be shared can be “anonymous”

  exstaff:85740 exterms:address _:johnaddress .

  _:johnaddress exterms:street "1501 Gr. Avenue" .

  _:johnaddress exterms:city "Bedford" .

- A “_:johnaddress” node in a different graph creates no conflict
- Turtle syntax needs no name, just a [ ] :

  exstaff:85740 exterms:address [

  exterms:street "1501 Gr. Avenue";

  exterms:city "Bedford"].

Anonymous nodes and identity

- Instead of:
  - ex2terms:book784 ex2terms:author "Jane Smith“
- We should say:
  - ex2terms:book784 ex2terms:author _:author784 .
  - _:author784 rdf:type ex2terms:Person .
  - _:author784 ex2terms:name "Jane Smith" .
- This means: exists x such that the author of book784 is x and the name of x is Jane Smith

Typed literals

- A literal is a value-type pair:
  - exstaff:85740 exterms:age "27"^^xsd:integer

RDF/XML

ex:index.html exterms:creation-date "August 16, 1999"

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/.../22-rdf-syntax-ns#"
  xmlns:exterms="http://www.example.org/terms/">
    <rdf:Description
      rdf:about="http://www.example.org/index.html">
        <exterms:creation-date>August 16, 1999
        </exterms:creation-date>
    </rdf:Description>
</rdf:RDF>
```

Reification

- RDF defines a class with three predicates to describe triples. A triple:
  - exproducts:item10245
    exterms:weight "2.4"^^xsd:decimal .
- Can be expressed (reified) as:
  - **exproducts:triple12345** rdf:type **rdf:Statement** .
  - **exproducts:triple12345** rdf:subject
    exproducts:item10245 .
  - **exproducts:triple12345** rdf:predicate exterms:weight .
  - **exproducts:triple12345** rdf:object "2.4"^^xsd:decimal .
  - **exproducts:triple12345 dc:creator exstaff:85740** .

RDF Schema

- The rdf:type property:
  - ex:plate736421 **rdf:type** ex:Truck
- Declaring classes and subclasses
  - ex:MotorVehicle **rdf:type rdfs:Class** .
  - ex:Van **rdf:type rdfs:Class** .
  - ex:Truck **rdf:type rdfs:Class** .
  - ex:Van **rdfs:subClassOf** ex:MotorVehicle .
  - Ex:Truck **rdfs:subClassOf** ex:MotorVehicle .
- (Use of rdf: rather than rdfs: is historical accident)

Domain and range and of properties

- ex:Person **rdf:type** **rdfs:Class** .
- ex:authorOf **rdf:type** **rdf:Property** .
- ex:authorOf **rdfs:domain** ex:Person .
- ex:authorOf **rdfs:range** ex:Book .
- Means:
  - If “x ex:authorOf y” then “x rdfs:type Person” and “y rdfs:type Book”
  - ex:authorOf $\subseteq$ (ex:Person $\times$ ex:Book)

Two classes with the same property?

- ex:weight rdf:type rdf:Property .
- ex:weight rdfs:range xsd:Integer .
- ex:weight rdfs:domain ex:Car .
- ex:weight rdfs:domain ex:Book .
- What does that mean?

Sharing properties - the right way

- ex:weight rdf:type rdf:Property .
- ex:carweight rdf:type rdf:Property .
- ex:bookweight rdf:type rdf:Property .
- ex:carweight rdfs:subPropertyOf ex:weight .
- ex:bookweight rdfs:subPropertyOf ex:weight .
- ex:weight rdfs:range xsd:Integer .
- ex:carweight rdfs:domain ex:Car .
- ex:bookweight rdfs:domain ex:Book .

Discussion

- Properties are not local to a class
- Once a property has an associated domain, it should not be associated to a different domain
- A property may have no associated domain
- No property is mandatory

Predefined properties

- rdfs:comment
- rdfs:label : a “readable” version of the resource name
- rdfs:seeAlso
- rdfs:isDefinedBy

Missing features

- Cardinality constraints
- Disjointness constraints
- Key constraints
- Identification of two different resources
- Declared disjunction of two different resources
- Closed collection
- Defining new classes as union/intersection of existing classes
- Transitivity / simmetry / reflexivity / … of properties
- …

Formal semantics of RDF

- Needed to disambiguate some notions
- Defines logical implication:
  - T1 |= T2
    iff for any M M |= T1 $\Rightarrow$ M |= T2
- For example:
  - :john :marriedTo :jane |= :john rdf:type :Person ?

Formal semantics of RDF

- Defines the notion of interpretation I for a set of names (a vocabulary)
- Every name is mapped by I to one resource; specifically, properties and types are mapped to resources (not to sets)
- A different function IEXT maps a property to a set of pairs
- Each literal (atomic value) is mapped by I onto itslef
- s p o is true in I/IEXT iff <I(s),I(o)> $\in$ IEXT(I(p))
- Two different names can be mapped to the same entity

Formal semantics of blank nodes

- <e:me> <e:hasDegree> _:x .
- <e:me> <e:hasDegree> _:y .
- _:x <e:subject> CS .
- _:y <e:subject> Eng .
- Has ‘me’ got two distinct degrees?

Formal semantics of blank nodes

- _:xxx <ex:p> <ex:b> .
- <ex:a> <ex:p> _:xxx .
- Means:
  - ∃x. x <ex:p> <ex:b>
  - <ex:a> <ex:p> x
- Hence, the second triple below is redundant:
  - <ex:a> <ex:p> <ex:b> .
  - <ex:a> <ex:p> _:xxx .

Formal semantics of blank nodes

- <e:me> <e:hasDegree> _:x .
- <e:me> <e:hasDegree> _:y .
- _:x <e:subject> CS .
- _:y <e:subject> Eng .
- I may have only one degree, with two subjects
- Remember that <e:me> may be the same as <e:you>

RDF-interpretations

- Must respect the semantics of:
  - rdf:type
  - rdf:Property
  - rdf:XMLLiteral, rdf:nil, rdf:List, rdf:Statement, rdf:subject, rdf:predicate, rdf:object, rdf:first, rdf:rest, rdf:Seq, rdf:Bag, rdf:Alt, rdf:_1, rdf:_2, ..., rdf:value
- For example:
  - p rdf:type P holds then p in P:
    - $I(p) \in ICEXT(P)$
    - In detail: $\langle I(p), I(P) \rangle \in IEXT(I(rdf:type)) \Rightarrow I(p) \in ICEXT(P)$

RDFS-interpretations

- Fixes the meaning of: rdf:type, rdfs:domain, rdfs:range, rdfs:Resource, rdfs:Literal, rdfs:Datatype, rdfs:Class, rdfs:subClassOf, rdfs:subPropertyOf
  - $x \in ICEXT(y)$ IFF $\langle x,y \rangle \in IEXT(I(rdf:type))$
  - If $\langle p,a \rangle \in IEXT(I(rdfs:domain))$ and $\langle u,v \rangle \in IEXT(p)$ then $u \in ICEXT(a)$
  - …

Implication

- S rdfs-implies G iff every rdfs-interpretation that satisfies S satisfies G
  - A rdfs:subClassOf B .
    vvv rdf:type A .

    |= vvv rdf:type B .

Implication

- S implies G iff every interpretation that satisfies S satisfies G
  - <e:a> <e:p> <e:b> |= <e:a> <e:p> _:x
  - <e:a> <e:p> <e:b> |=/= rdf:type rdf:type rdf:Property
- S rdf-implies G iff every rdf-interpretation that satisfies S satisfies G
  - <e:a> <e:p> <e:b> |= <e:a> <e:p> _:x
  - <e:a> <e:p> <e:b> |= rdf:type rdf:type rdf:Property

Consistency

- A theory G is consisent iff it has a model
- Every rdf theory is consistent
- Intuitively, a theory is inconsistent when it implies both P and Not P, for some P. In RDF there is not way neither to express ‘Not P’ nor to deduce ‘Not P’