Más o menos

Semantic Web: Lógica

DATA:

LOGIC: “$(b$,capital,$a)$ $\rightarrow$ $(a$,partOf,$b)$”

“$(a$,partOf,$b)$, $(b$,partOf,$c)$ $\rightarrow$ $(a$,partOf,$c)$”

QUERY: “$(x$,partOf,$y)$?”

OUTPUT: $\{(x \mapsto \text{Ireland}, y \mapsto \text{Europe}),$
$(x \mapsto \text{Dublin}, y \mapsto \text{Ireland}),$
$(x \mapsto \text{Dublin}, y \mapsto \text{Europe})\}$

¿Cómo capturar la lógica?

LOGIC: “$(b,\text{capital},a) \rightarrow (a,\text{partOf},b)$”

“$(a,\text{partOf},b),\ (b,\text{partOf},c) \rightarrow (a,\text{partOf},c)$”

Web Semántica dice: Schema/Ontologías

- En lugar de reglas, usamos RDF
- Define relaciones entre clases y propiedades

¿Qué tipo de relaciones pueden ser útiles definir entre las siguientes clases y propiedades?

ex:Town

ex:hasCapitalCity

ex:City

ex:partOf

ex:Country

foaf:familyName

ex:hasCity

ex:Place

foaf:Person

ex:geographicallyPartOf

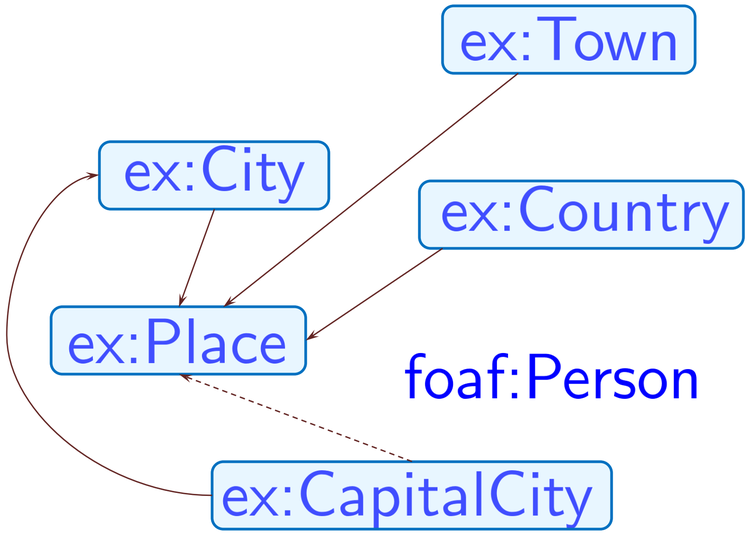
ex:CapitalCity

Jerarquías de clases

- Clase c es sub-class de Clase d
  - Si (x, rdf:type, c) entonces (x, rdf:type, d),

Ejemplo: si ex:CapitalCity *es sub-class de* ex:City
y si (ex:Dublin, rdf:type, ex:CapitalCity)
entonces (ex:Dublin, rdf:type, ex:City)

¿Cuáles clases deben ser sub-classes de cuales?



Jerarquía de propiedades

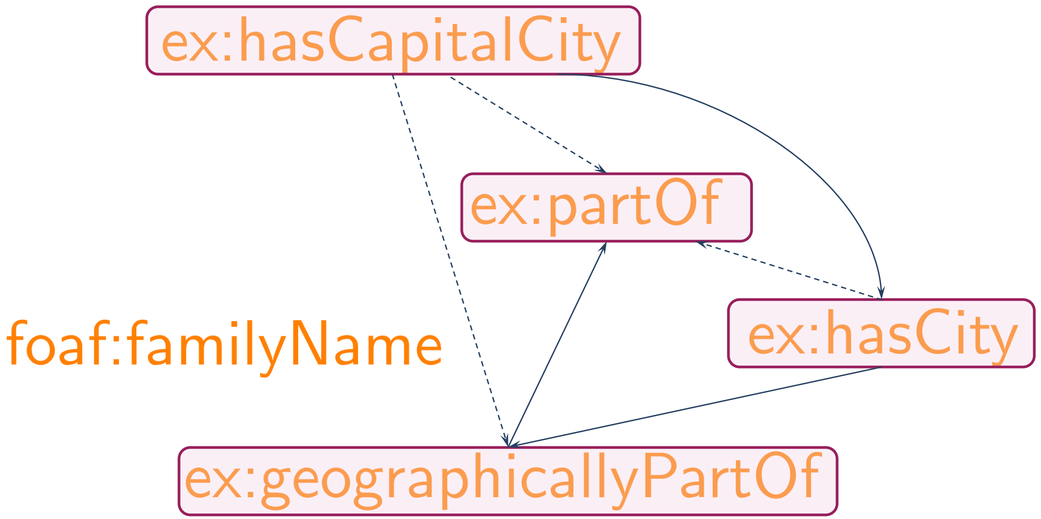
- La Propiedad p es una sub-property de q
  - Si (x,p,y) entonces (x,q,y)

Ejemplo: si ex:hasCapitalCity *es sub-property de* ex:hasCity

Y si (ex:Ireland,ex:hasCapitalCity,ex:Dublin)

entonces (ex:Ireland,ex:hasCity,ex:Dublin)

¿Qué propiedades deben ser sub-properties de cuales?

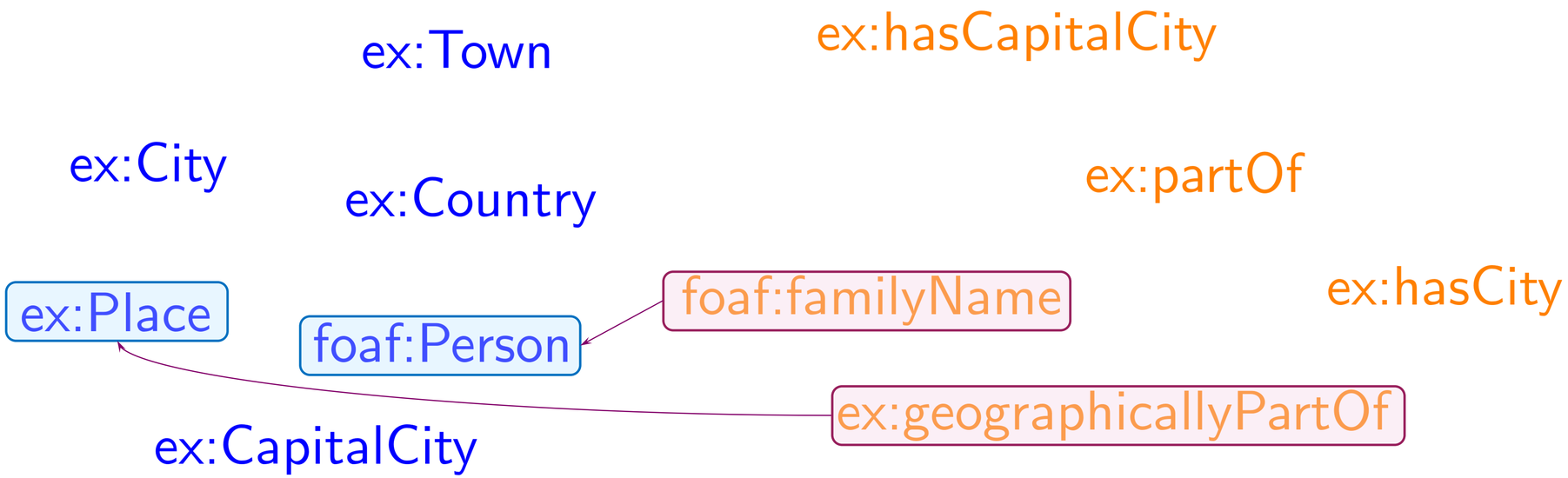


Dominio en las propiedades

- La Propiedad p tiene domain a la clase c
  - Si (x,p,y) entonces (x,rdf:type,c)

Ejemplo: si `foaf:familyName` *tiene domain* `foaf:Person`
Y si `(ex:Diego,foaf:familyName,"Torres")`
entonces `(ex:Diego,rdf:type,foaf:Person)`

¿Cuáles propiedades vinculan con las clases como domain?



Rango de las propiedades

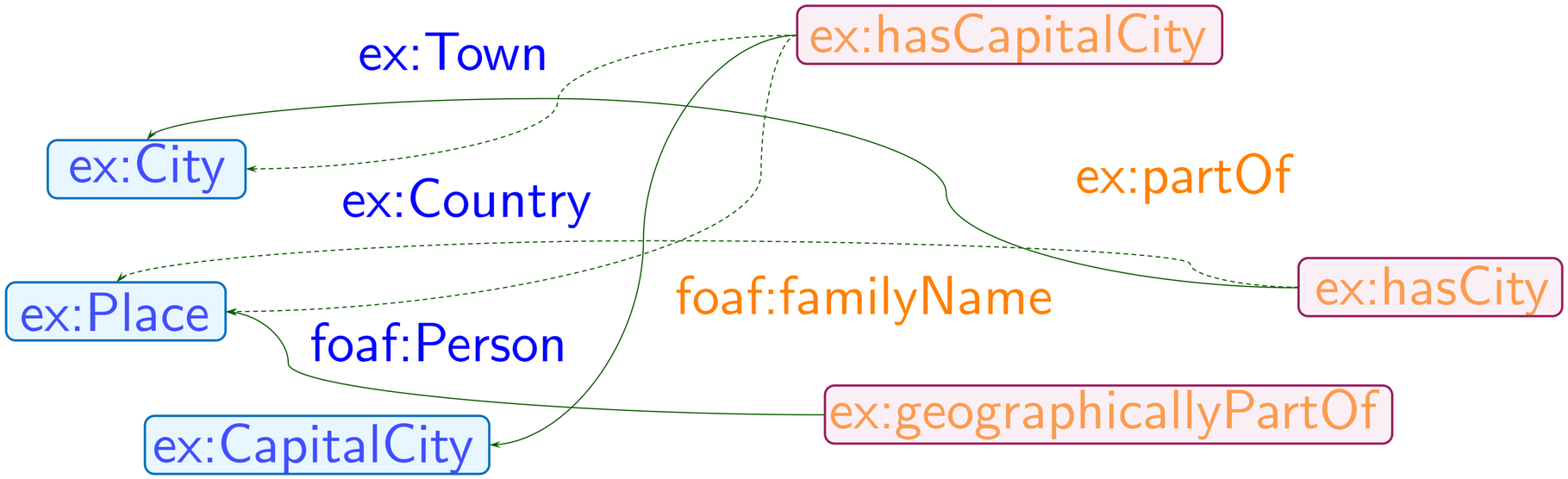
- La propiedad p tiene de range a la clase c
  - si (x,p,y) entonces (y,rdf:type,c)

Ejemplo: si ex:hasCity *has range* ex:City

Y si (ex:Ireland,ex:hasCity,ex:Dublin)

entonces (ex:Dublin,rdf:type,ex:City)

¿Cuales propiedades vincularin con las clases para su range?



Trade-off: Más especifico / Menos Reusable

- Más específico → más conclusiones
- Menos específico → más reusable

Ejemplo: `ex:hasCapitalCity` has domain `ex:Country`

PRO: Conocer que quien tenga una ciudad capital es un páis.

CON: No se puede usar para capitals de estados, regions, etc

Trade-off: Más específico / Menos Reusable

- Otro ejemplo:
  - `ex:Mayor` sub-class of `foaf:Person`

Bosco the dog

Mayor of Sunol, California

1981–1994

R.I.P.

Trade-off: Más específico / Menos Reusable

- Otro ejemplo:
  - `ex:spouse` has domain/range `foaf:Person`

Cuidado con las definiciones ocultas

FOAF Vocabulary Specification 0.99

Namespace Document 14 January 2014 - *Paddington Edition*

Property: foaf:img

image - An image that can be used to represent some thing (ie. those depictions which are particularly representative of something, eg. one's photo on a homepage).
Status: testing
Domain: having this property implies being a Person
Range: every value of this property is a Image

¿Detectan algun problema potencial?

`(ex:Dublin,foaf:img,ex:Dublin_night.jpg)`

Elijan nombres de clases/propiedades adecuados!

RDFS: RDF Schema

RDFs y Datatypes

Diego Torres

Caso disparador 1

De libro

- Agricultura por tipo (7 cat, 17 págs.)
  - Agricultura urbana (11 págs.)
  - Agricultura ecológica (1 cat, 62 págs.)
    - Abono orgánico (7 págs.) → Compost
  - Fruticultura (3 cat, 13 págs.)
    - Árboles frutales (2 cat, 74 págs.)
      - Cerezas (2 cat, 11 págs.)
      - Pomología (12 págs.)
    - Frutas (21 cat, 222 págs.)
    - Portainjertos (10 págs.)
  - Horticultura (10 cat, 88 págs.)
    - Horticultores (2 cat)
    - Drenaje (12 págs.)

Desafío

¿Qué debería hacer para conocer todas las categorías a las que pertenece Compost?

Caso disparador

De libro ….

`ex:GabrielGarcíaMarquez` `ex:escribioNovela` `ex:CienAñosDeSoledad`

Contamos con:

- Propiedades `ex:autorDe`
- Clases: `ex:Novelista, ex:Persona, ex:Escritor, ex:TrabajoLiterario`

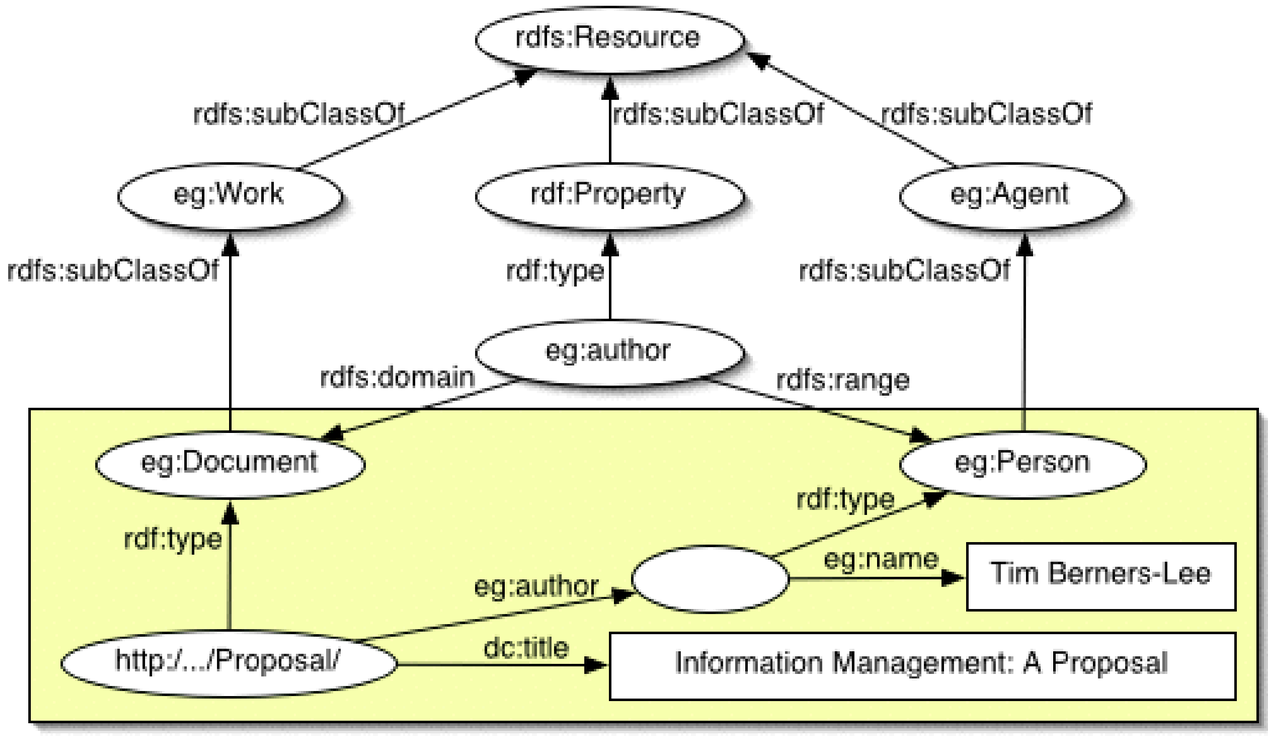
Desafío

¿Qué podríamos deducir con nuestro sentido común?

```
ex:GabrielGarcíaMarquez  rdf:Type ex:Escritor
ex:GabrielGarcíaMarquez  rdf:Type ex:Novelista
ex:GabrielGarcíaMarquez  rdf:Type ex:Persona
ex:CienAñosDeSoledad rdf:Type ex:TrabajoLiterario
ex:GabrielGarcíaMarquez  ex:autorDe ex:CienAñosDeSoledad
```

La necesidad de tener esquemas

- Definir términos que se pueden utilizar.
- Restricciones a aplicar
- Relaciones extras que aparecen en el contexto.
- Esto es RDFs: Lenguaje de descripción del vocabulario RDF

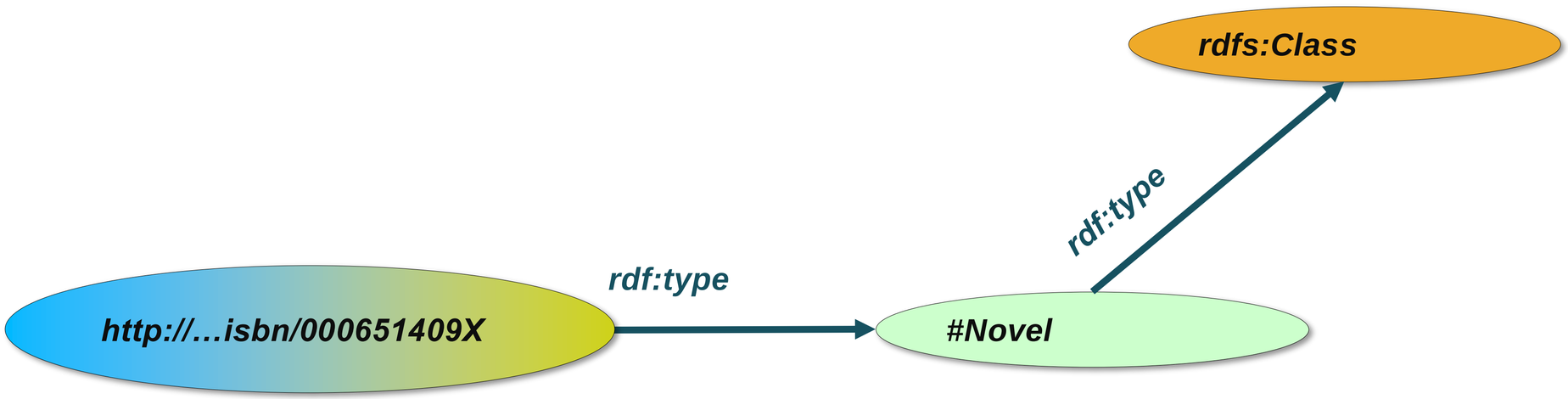


RDFs
RDF Schema

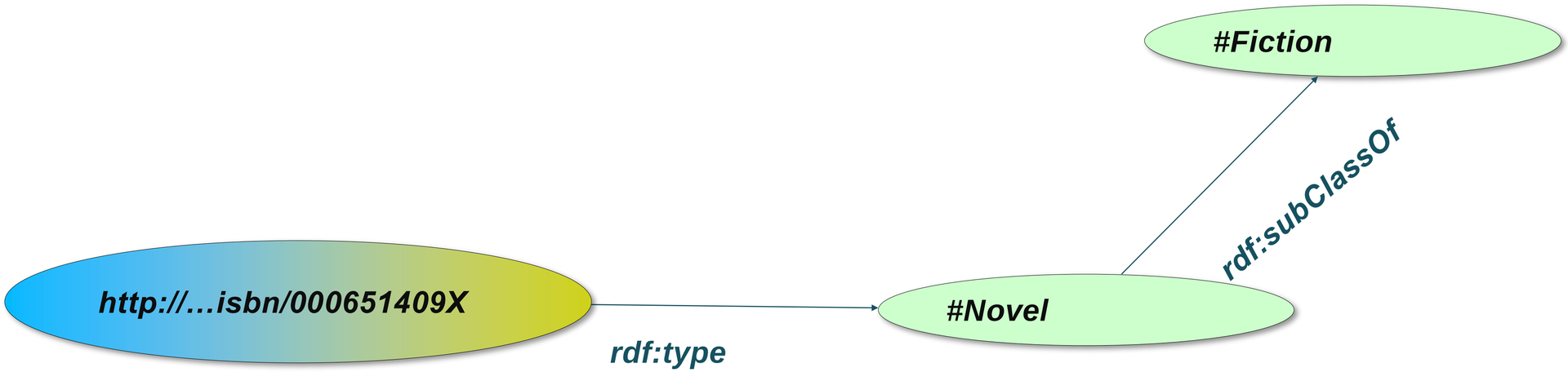
- Vocabulario
  - Sub-class
  - Sub-property
  - Domain
  - Range

SUB-CLASS

- RDFS define el significado de los términos
  - (todas estas son URIs especiales en las cuales estamos utilizando un abreviación para su espacio de nombres) abbreviation)



INFERENCIA DE PROPIEDADES




```
<http://…/isbn/000651409X> rdf:type #Fiction
```

- no era parte del RDF
- …pero puede obtenerse a través de las reglas de RDFS
- los ambientes RDFS retornan también la ultima tripleta

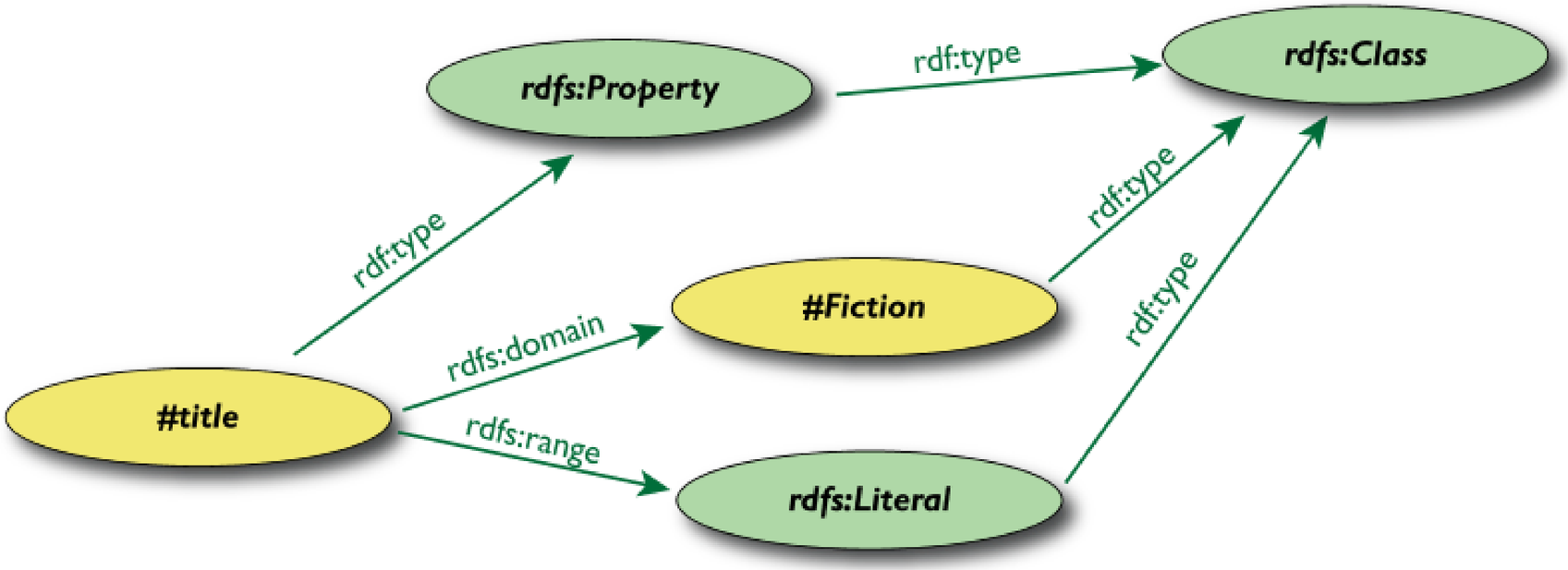
PROPIEDADES

- Property es una clase especial (rdf:Property)
  - Las propiedades también se identifican con una URI.
- Existe la posibilidad de definir sub-propiedades
  - todos los recursos ligados por la “sub” también son ligados a la otra.
- Se define el rango y el dominio de los elementos que relaciona
  - Qué tipo de recurso puede ser el objeto y cuales el sujeto

PROPIEDADES (CONT.)

- También son recursos
- Por lo cual, propiedades entre propiedades también se pueden expresar como propiedades RDF
- Por ejmplo, (P rdfs:domain C) significa:
  - P es una propiedad
  - C es una clase
  - Cuando uso P, puede inferir que el sujeto es de tipo C

EJEMPLO DE ESPECIFICACIÓN DE UNA PROPIEDAD




```
#title rdf:type  rdfs:Property;
#title  rdfs:domain #Fiction;
#titel  rdfs:range  rdfs:Literal.
```

¿QUÉ SIGNIFICA ESTO?

- Se pueden inferir nuevas relaciones ….

```
:title
   rdf:type      rdf:Property;
   rdfs:domain   :Fiction;
   rdfs:range    rdfs:Literal.
```

```
<http://…/isbn/000651409X> :title "One hundred years of solitude"
```

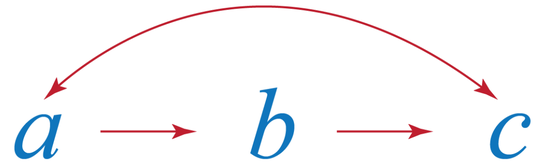
el sistema puede inferir

```
<http://…/isbn/000651409X> rdf:type :Fiction .
```

Sub property

tieneIngrediente rdfs:subPropertyOf contiene

Indicamos que cualesquiera dos recursos que están relacionados por la primera propiedad deben estar relacionados también por la última.



rdfs:subPropertyOf es transitiva

LemonPie tieneIngrediente Limón

De las dos sentencias podemos llegar a la conclusión que

LemonPie contiene Limón

tieneAderezo rdfs:subPropertyOf tieneIngredientes
tieneIngredientes rdfs:subPropertyOf contiene

tieneAderezo rdfs:subPropertyOf contiene

Anotaciones

- **rdfs:label:** es una propiedad que relaciona un **recurso** con una etiqueta legible por personas.
- **rdfs:comment**: es una propiedad que relaciona a un **recurso** con un comentario legible por personas.
- **rdfs:seeAlso** es una propiedad que relaciona un **recurso** con una ubicación en la Web que tiene información sobre él.
- **rdfs:isDefinedBy** es una propiedad que relaciona un **recurso** con una ubicación en la Web en la que se da una definición sobre él.

Anotaciones
Ejemplos reales

```
dbr:Argentina  rdf:type        dbo:Country


dbr:Argentina rdfs:label      "Argentine"@fr ,
dbr:Argentina rdfs:label      "Argentina"@en ,
dbr:Argentina rdfs:label      "Argentina"@es ,
dbr:Argentina rdfs:comment    "Argentini\u00EB (Spaans: Argentina), officieel de
Argentijnse Republiek ..."@nl


dbr:Ushuaia    rdfs:seeAlso   dbr:Argentina
```

```
<rdfs:Class rdf:ID="ResourceAccessRule">
  <rdfs:label xml:lang="en">Access Rule</rdfs:label>
  <rdfs:comment>An assertion of access privileges to a rdf:resource.</rdfs:comment>
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2001/02/acls/ns#"/>
</rdfs:Class>
```

OTROS CONTENEDORES

- Tambien están definidos en RDF
  - rdf:Bag
    - un bag general, no poseen semántica asociada
  - rdf:Alt
    - semántica convenida: sólo uno de los elementos puede ser considerado válido
- Estos contenedores son semánticamente incompletos, es mejor NO usarlos, en su lugar
  - usar predicados repetidos en lugar de bags
  - usar listas en lugar de secuencias

OWL
Web Ontology Language

Motivación

De libro

```
ex1:NLP1983              ex1:awardedTo        ex1:WilliamGolding
ex1:NLP1983
ex1:NLP1983
ex1:WilliamGolding
ex1:WilliamGolding
```

Intuitivamente podemos imaginar que la consulta pide los nombres de las personas que lucharon en la Segunda Guerra Mundial y son premios Nobel de Literatura

```
ex2:WGGolding
ex2:WGGolding            ex2:fought in        ex2:NormandyInvasion
```

```
ex3:NormandyInvasion     rdf:type             ex3:Conflict
ex3:NormandyInvasion     ex3:partOf           ex3:OperationOverlord
ex3:OperationOverlord    ex3:partOf           ex3:WorldWarII
```

Desafío

```
?x                       rdf:type             ex4:WWIIVeteran
?x                       rdf:type             ex4:NobelLaureateLit
?x                       rdfs:label           ?y
```

Con RDFs podemos agregar lo siguiente

ex1:Novelist	rdfs:subClassOf	ex2:Author
ex2:Author	rdfs:subClassOf	ex2:Person
ex2:foughtIn	rdfs:domain	ex2:Person
ex2:foughtIn	rdfs:range	ex3:Conflict
ex2:foughtIn	rdfs:subPropertyOf	ex2:involvedIn

ex1:NLP1983	ex1:awardedTo	ex1:WilliamGolding
ex1:NLP1983	rdf:type	ex1:NobelPrizeLit
ex1:NLP1983	ex1:year	"1982"^^xsd:integer
ex1:WilliamGolding	rdf:type	ex1:Novelist
ex1:WilliamGolding	rdfs:label	"William Golding"

ex2:WGGolding	rdf:type	ex2:Author
ex2:WGGolding	ex2:foughtIn	ex2:NormandyInvasion

ex3:NormandyInvasion	rdf:type	ex3:Conflict
ex3:NormandyInvasion	ex3:partOf	ex3:OperationOverlord
ex3:OperationOverlord	ex3:partOf	ex3:WorldWarII

Enriquece mucho, pero aun no nos deja responder la consulta

?x	rdf:type	ex4:WWIIVeteran
?x	rdf:type	ex4:NobelLaureat
?x	rdfs:label	?y

Sería bueno poder indicar:

Son el mismo recurso

ex1:WilliamGolding y ex2: WGGolding

ex2:NormandyInvasiony ex3: NormandyInvasion

ex1:hasAward **sea la inversa de** ex1:awardedTo

La clase ex4:WWIIVeteran sea la clase de todos los recursos con el valor ex3:WolrdWarII para la propiedad ex2:foughtIn

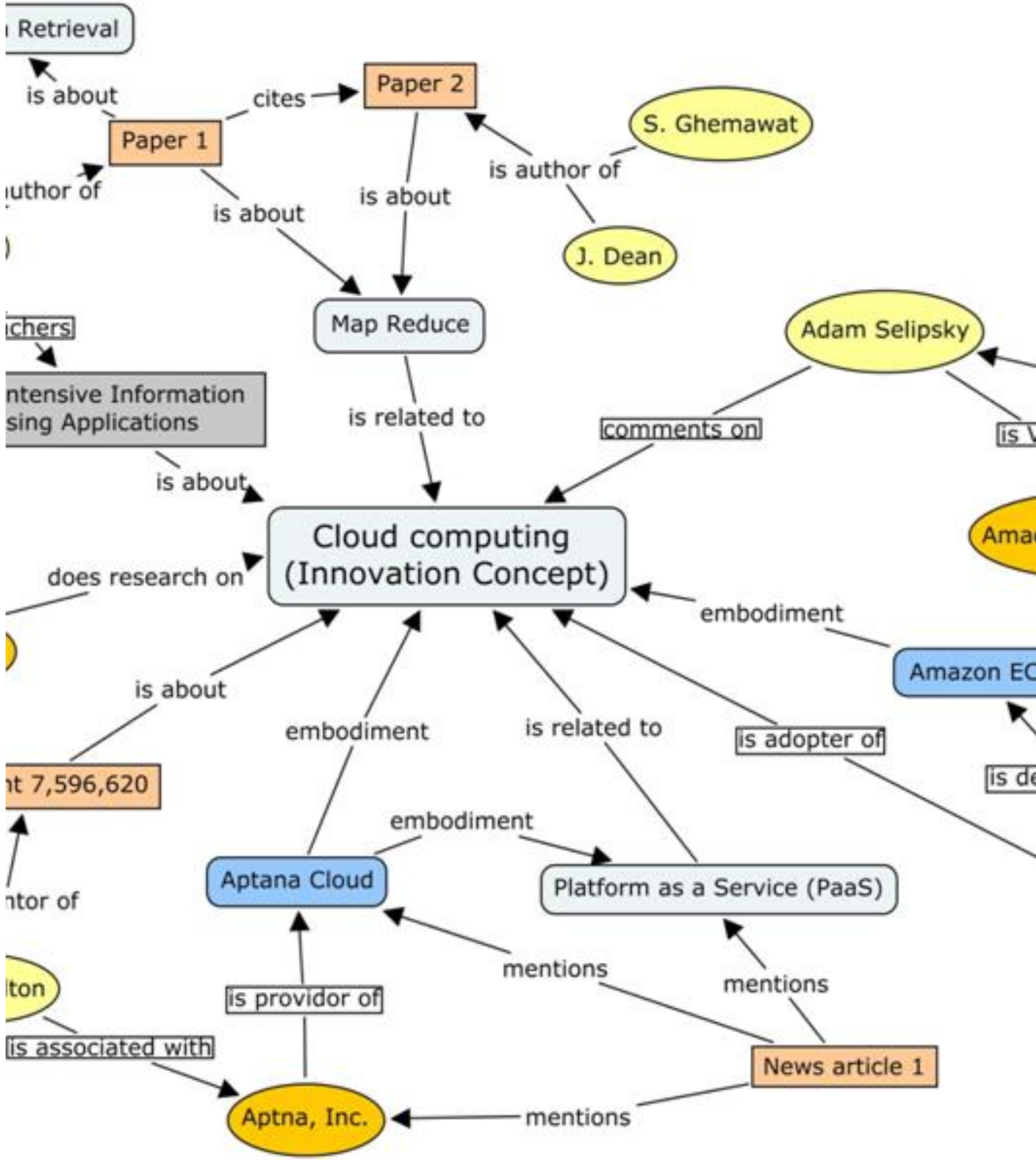
ex3:partOf **sea transitiva**

Si x ex2:foughtIn y - y ex3:partOf z , entonces x ex2:foughtIn z

La clase ex4:NobelLaureateLit sea la clase de todos los recursos con el valor de la clase ex1:NobelPrizeLit para la propiedad ex1: hasAward

Ontología

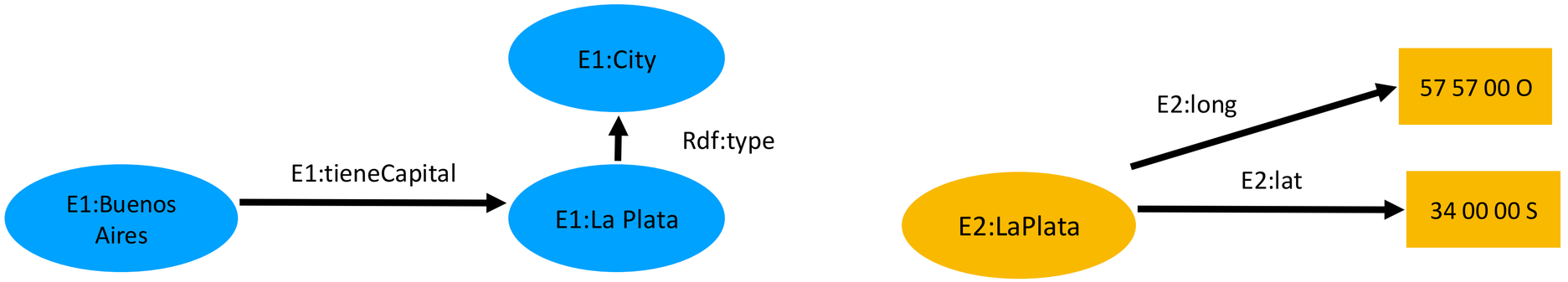
- En informática, una ontología es una representación formal del conocimiento que constituye una conceptualización compartida de un dominio determinado.
- Buscamos que se puedan realizar inferencias a partir de ellas.



OWL - Vocabulario

Extensión de RDFS

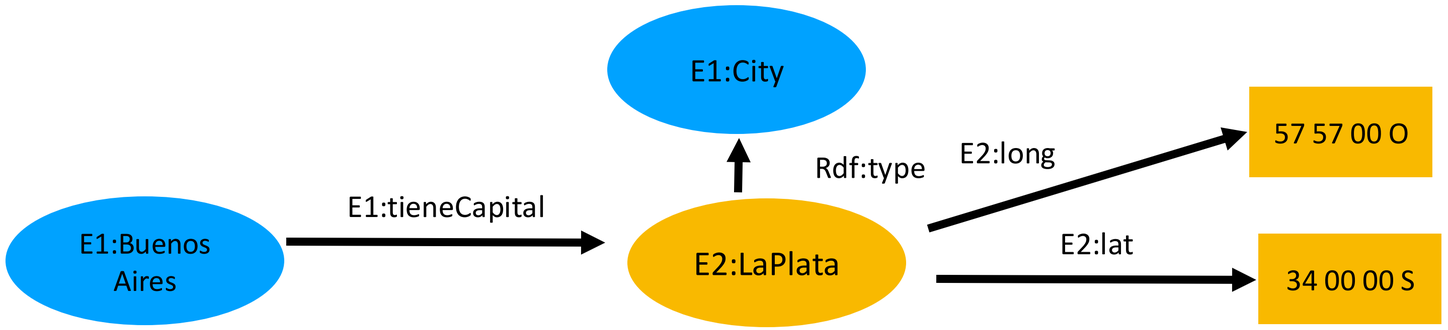
- owl:sameAs dos recursos rdf refieren al mismo recurso. Son co-referentes.
- owl:differentFrom dos recursos rdf refieren a recursos diferentes.



E1:La Plata owl:sameAs E2:LaPlata

OWL - Vocabulario
sameAs y differentFrom

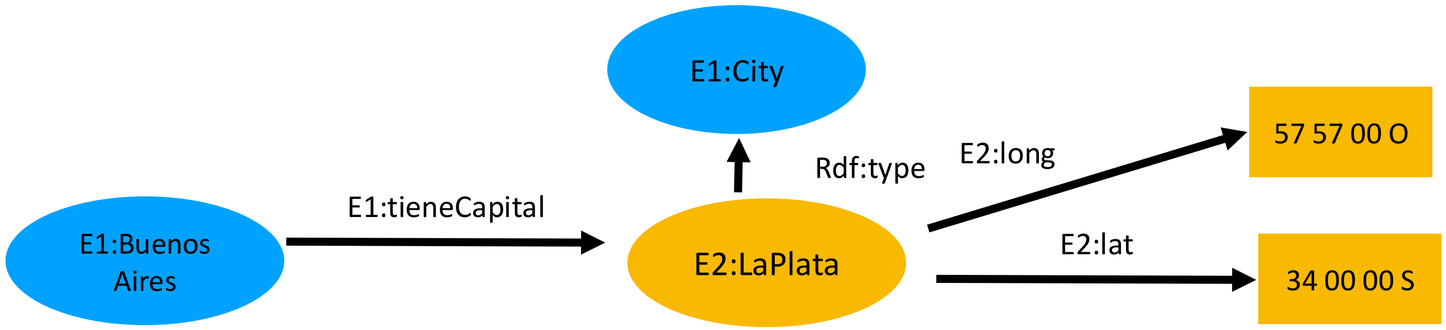
- owl:sameAs dos recursos rdf refieren al mismo recurso. Son co-referentes.
- owl:differentFrom dos recursos rdf refieren a recursos diferentes.



E1:La Plata owl:sameAs E2:LaPlata

OWL - Vocabulario
dataTypes y objectProperties

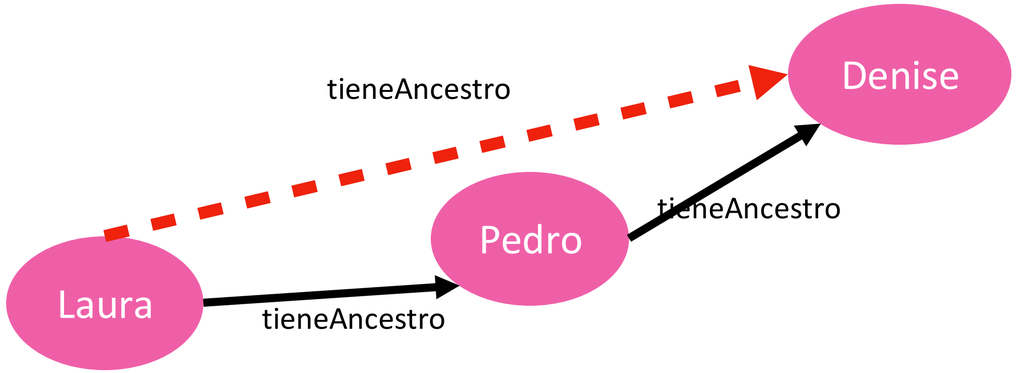
- owl:ObjectProperty es la clase de las propiedades que relaciona instancias con instancias.
- owl:DatatypeProperty es la clase de las propiedades que relacionan instancias con datatypes (números, texto, etc).



E1:tieneCapital rdf:type owl:ObjectProperty

owl:TransitiveProperty

- Si **p** es un miembro de la clase de propiedades **transitivas**, entonces p es una propiedad, y si p relaciona x1 con x2 y x2 con x3, entonces p también relaciona x1 con x3

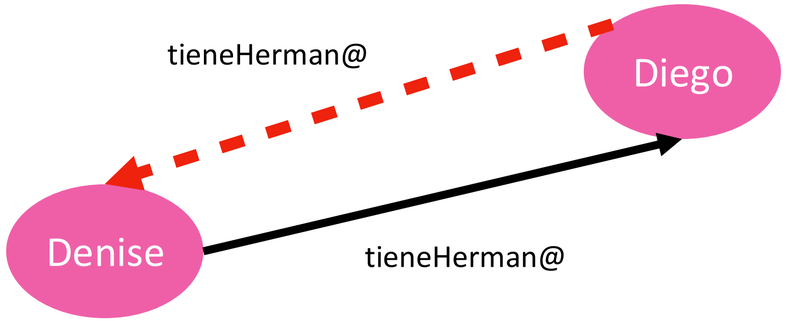


Ejemplo

- Si tieneAncestro es transitiva, y:
  - Laura tieneAncestro Pedro.
  - Pedro tieneAncestro Denise.
  - **Entonces**
    - Laura tieneAncestro Denise

owl:SymmetricProperty

- Si **p** es un miembro de la clase de propiedad **simétrica**, entonces p es una propiedad, y **si p relaciona x1 con x2**, entonces **p también relaciona x2 con x1.**

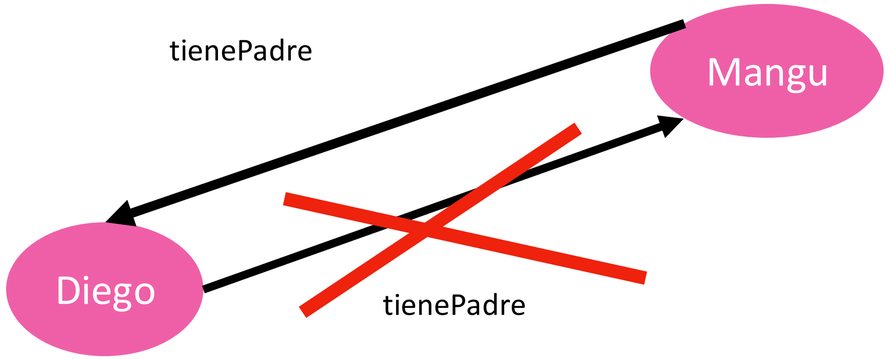


Ejemplo

- Si tieneHerman@ es simétrica, y:
  - Diego tieneHerman@ Denise.
  - **Entonces**
    - Denise tieneHerman@ Diego

owl:AsymmetricProperty

- Si **p** es un miembro de la clase de **propiedad asimétrica**, entonces p es una propiedad, y **si p relaciona x1 con x2**, entonces **p no puede relacionar x2 con x1**; en otras palabras, **si p relaciona x1 con x2 y x2 con x1, surge una contradicción**.

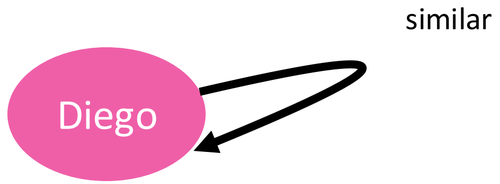


Ejemplo

- Si tienePadre es asimétrica, y:
  - Diego tienePadre Mangu.
  - **Entonces**
    - ~~Mangu tienePadre Diego~~

owl:ReflexiveProperty

- Si **p** es un miembro de la clase de **propiedad reflexiva**, entonces p es una propiedad que **relaciona cada recurso x ∈ IR consigo mismo.**

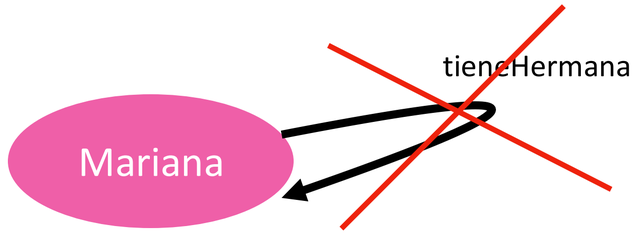


Ejemplo

- Si similar es reflexiva:
  - Todo elemento es similar a si mismo.

owl:IrreflexiveProperty

- Si **p** pertenece a la clase de **propiedades irreflexivas**, entonces **p** es una propiedad que **no puede relacionar ningún recurso consigo misma**; si p relaciona un recurso x consigo mismo, surge una contradicción.

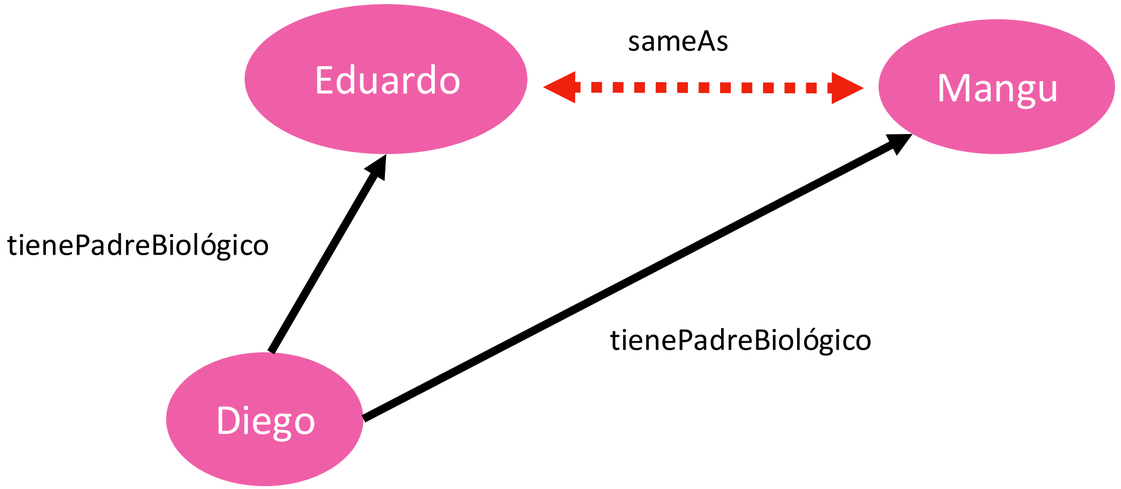


Ejemplo

- Si tieneHermana es irreflexiva:
  - Mariana tieneHermana Mariana, debe generar una contradicción.

owl:FunctionalProperty

- Si **p** es un miembro de la clase de **propiedades funcionales**, entonces p es una propiedad, y si **relaciona x con y1 y x con y2, concluimos que y1 es igual a y2.**

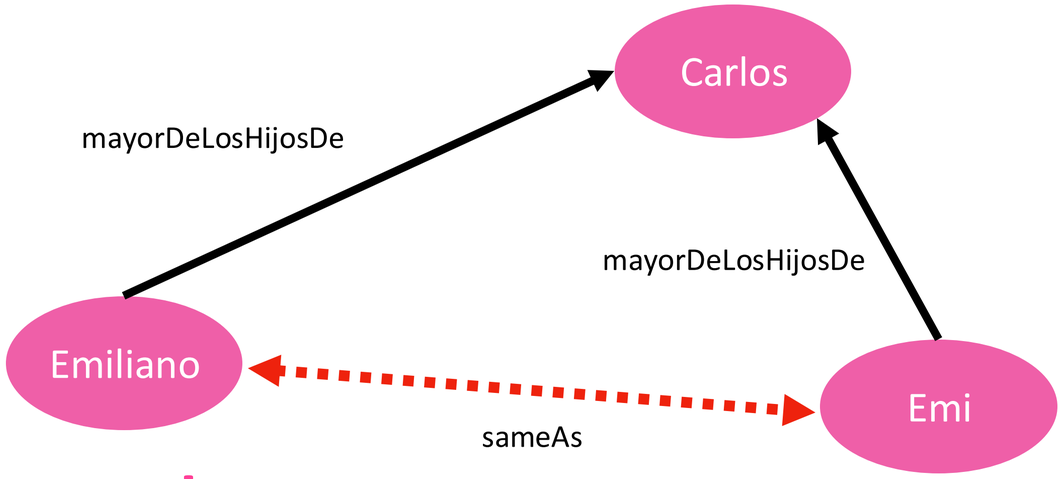


Ejemplo

- Si decimos que tienePadreBiológico es funcional, y sabemos
  - Diego tienePadreBiológico Mangu
  - Diego tienePadreBiológico Eduardo
- Entonces
  - Mangu es igual que Eduardo

owl:InverseFunctionalProperty

- Si **p es** un miembro de la clase de propiedades **funcionales inversas**, entonces p es una propiedad, y **si relaciona x1 con y** y **x2 con y,** debemos concluir que **x1 es igual a x2**



Ejemplo

- si mayorDeLosHijosDe es una propiedad funcional inversa, y
  - Emiliano mayorDeLosHijosDe Carlos
  - Emi mayorDeLosHijosDe Carlos
- **entonces**
  - Emi es igual a Emiliano.

Relaciones entre propiedades

- Owl:equivalentProperty
  - Relacionan a los mismo recursos de igual forma.
- Owl:inverseOf
  - hijoDe con progenitorDe
- Owl:propertyDisjointWith

- Owl:propertyChainAxiom
  - Una secuencia de propiedades.
    - hijoDe dos veces es nietoDe

Axiomas de clases

- Owl:equivalentClass
- Owl:disjointWith
- Owl:intersectionOf
- Owl:unionOf
- Owl:complementOf

```
f:Marriage rdf:type owl:Class .
_:x14 rdf:type owl:Class ;
   owl:unionOf (f:CivilMarriage f:ReligiousMarriage) .
f:Marriage owl:equivalentClass _:x14 .
```

```
:LivingPerson owl:equivalentClass
   [ owl:intersectionOf (
        :Person
        [ owl:complementOf :Deceased ]
     ) ] .
```

VOCABULARIOS

- RDFS y OWL hacen posible la definición de vocabularios:
  - colecciones de propiedades y clases
  - relaciones entre estos y términos de otros vocabularios
- Ejemplos
  - Dublin Core: creador, fecha, publicacion…
  - FOAF: caracterización de personas y relaciones de amistad
  - Good Relations: eCommerce
  - Creative Commons: clases para definir copyright, relaciones de licencias
  - schema.org: eventos, organizaciones, lugares, reviews

Metodología básica

Guía para crear la primera ontología

- Determine domain
- reusing existing ontologies
- enumerate important terms
- de-fines classes
- define properties
- define domain and ranges
- create instances.

Ontology Development 101: A Guide to Creating Your First Ontology

Natalya F. Noy and Deborah L. McGuinness
Stanford University, Stanford, CA, 94305
noy@smi.stanford.edu and dlm@ksl.stanford.edu

1 Why develop an ontology?

In recent years the development of ontologies—explicit formal specifications of the terms in the domain and relations among them (Gruber 1993)—has been moving from the realm of Artificial-Intelligence laboratories to the desktops of domain experts. Ontologies have become common on the World-Wide Web. The ontologies on the Web range from large taxonomies categorizing Web sites (such as on Yahoo!) to categorizations of products for sale and their features (such as on Amazon.com). The WWW Consortium (W3C) is developing the Resource Description Framework (Brickley and Guha 1999), a language for encoding knowledge on Web pages to make it understandable to electronic agents searching for information. The Defense Advanced Research Projects Agency (DARPA), in conjunction with the W3C, is developing DARPA Agent Markup Language (DAML) by extending RDF with more expressive constructs aimed at facilitating agent interaction on the Web (Hendler and McGuinness 2000). Many disciplines now develop standardized ontologies that domain experts can use to share and annotate information in their fields. Medicine, for example, has produced large, standardized, structured vocabularies such as SNOMED (Price and Spackman 2000) and the semantic network of the Unified Medical Language System (Humphreys and Lindberg 1993). Broad general-purpose ontologies are emerging as well. For example, the United Nations Development Program and Dun & Bradstreet combined their efforts to develop the UNSPSC ontology which provides terminology for products and services (www.unspsc.org).

An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them.

Why would someone want to develop an ontology? Some of the reasons are:

To share common understanding of the structure of information among people or software agents

To enable reuse of domain knowledge

To make domain assumptions explicit

To separate domain knowledge from the operational knowledge

To analyze domain knowledge

https://protege.stanford.edu

Ejemplo para interesante: https://protege.stanford.edu/ontologies/pizza/pizza.owl