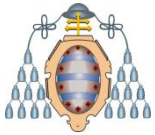


Ontologías y Web Semántica

Jose Emilio Labra Gayo

Departamento de Informática
Universidad de Oviedo
http://labra.weso.es

Jose Labra
Universidad de Oviedo

¿Qué es una Ontología?

Ontología = Formalización de un dominio

- Utiliza: lenguajes formales
- Para: definir **vocabulario** de un dominio
- **Compartir** el significado entre aplicaciones
- **Inferir** nuevo conocimiento a partir de definiciones

Otros términos relacionados:

- Taxonomía: Clasficación jerárquica
- Tesauro: Definiciones de términos

Jose Labra
Universidad de Oviedo

Ejemplos de dominios

Medicina

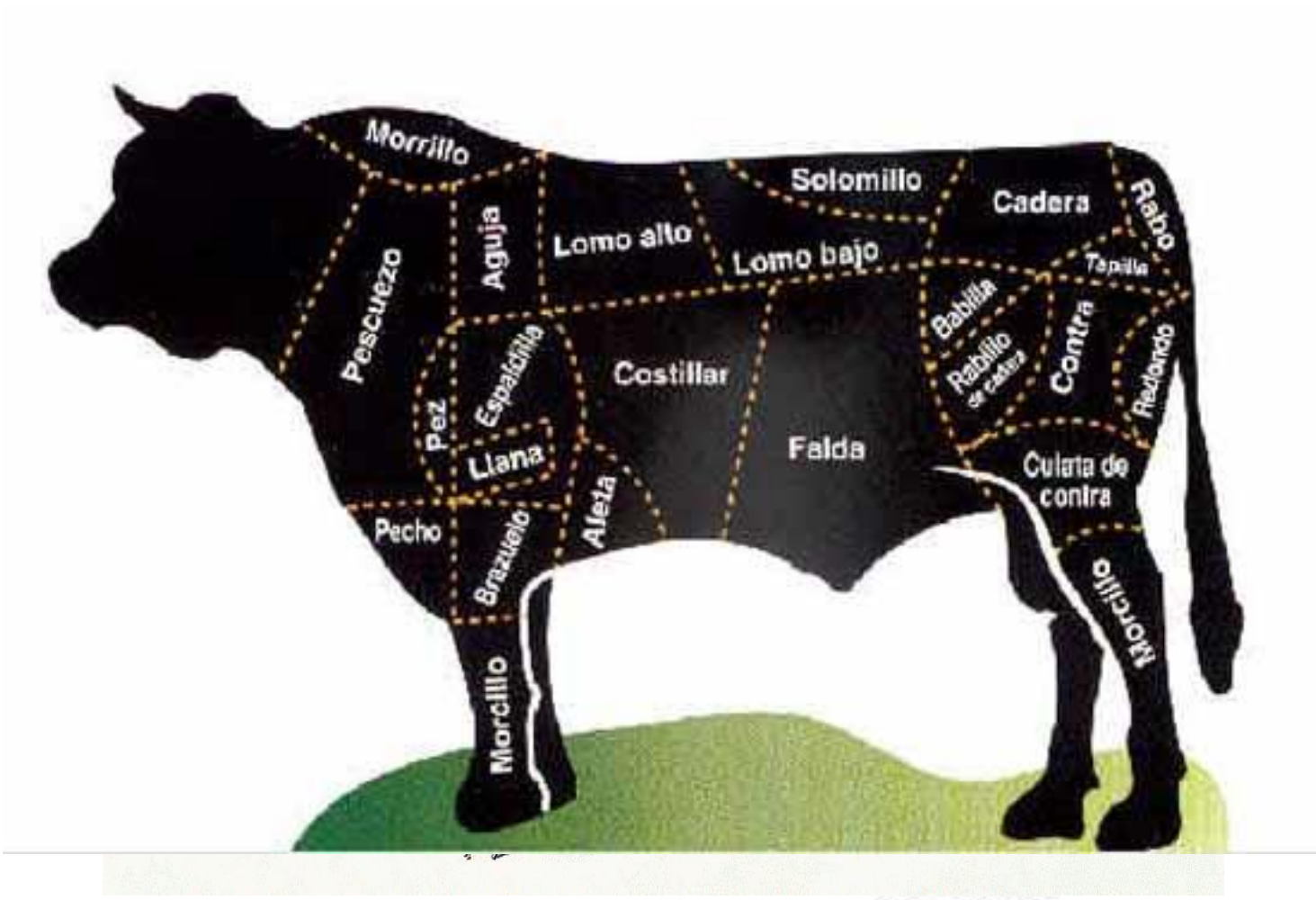
Biología

Aviación

Animales

Comida

…etc



Jose Labra
Universidad de Oviedo

Partes de una ontología

Define conjunto de términos (vocabulario)

Corazón, Sangre, Sistema circulatorio

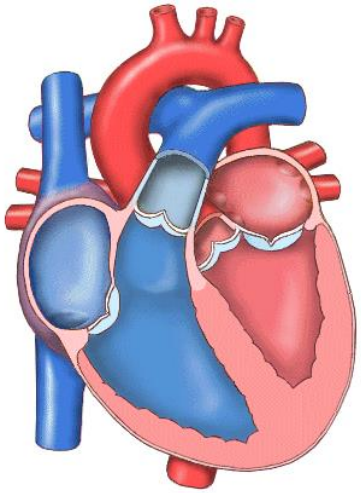
Propiedades entre dichos términos

Ejemplo:

"*el corazón es un órgano muscular que es parte del sistema circulatorio*"

Descrito en un lenguaje formal

Ejemplo (lógica):



$$\forall x(\text{Corazón}(x) \rightarrow \text{OrganoMuscular}(x) \wedge \exists y\ (\text{esParteDe}(x,y) \wedge \text{SistemaCirculatorio}(y))$$

Jose Labra
Universidad de Oviedo

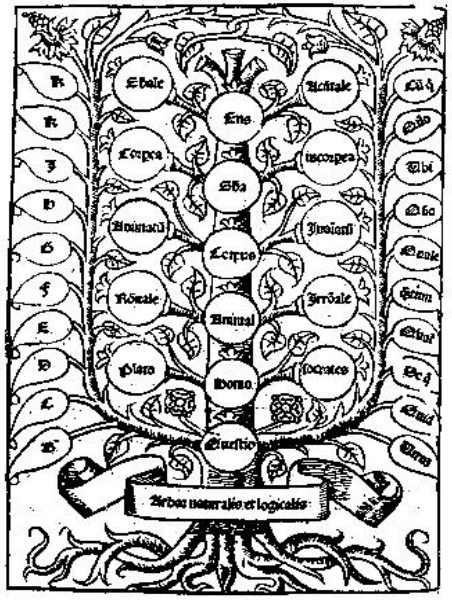
Ontología como rama del conocimiento

Ontología, rama de la metafísica

- Desde Aristóteles (metafisica, IV)
- Onto=ser, logos=estudio de (*estudio del ser*)
- Estudia los entes, sus categorías y relaciones

Representación del conocimiento

- Ontología = formalización de un dominio
- Un vocabulario compartido que describe un determinado dominio
- Un conjunto de declaraciones sobre términos
- Usar lenguaje formal, manipulable automáticamente.



Árbol de la naturaleza
y de la lógica
Ramón Llull (1235-1316)

Jose Labra
Universidad de Oviedo

Lógica

Lógica: Estudio de los razonamientos

- Origen en Aristóteles (-342 a. de C.)
- Desarrollo de la Lógica formal a finales s. XIX (De Morgan, Fregge)
- Lógica computacional (Hilbert, Church, Turing, Herbrand, Tarski, …)
- Varios sistemas de Lógica
  - Lógica proposicional
  - Lógica de predicados
  - Otras lógicas: lógica modal, lógica descriptiva, lógica borrosa, etc.

Jose Labra
Universidad de Oviedo

Lógica Proposicional

Cada frase que puede ser verdadera o falsa es una proposicón o enunciado (p)

Varias conectivas:

Negación: $\neg p$
Conjunción: $p \wedge q$
Disyunción: $p \vee q$
Implicación: $p \rightarrow q$
Equivalencia: $p \leftrightarrow q$

“Si Juan juega al fútbol, se cansa y Juan juega al fútbol
Por tanto: Juan se cansa”

$$\frac{p \rightarrow q \quad p}{q}$$

Jose Labra
Universidad de Oviedo

Lógica proposicional

Existen sistemas de demostración que comprueban si un razonamiento es correcto

Propiedades:

- **Consistente**: todos los razonamientos que se demuestran son correctos
- **Completo**: todos los razonamientos correctos pueden demostrarse
- **Complejidad**: NP (es uno de los problemas NP clásicos)
- **Expresividad:** Muy poca.

  Ejemplo: *"Todos los hombres son mortales, Sócrates es un hombre, luego Sócrates es mortal"*

Jose Labra
Universidad de Oviedo

Lógica de predicados

Extiende la lógica proposicional con predicados, funciones y cuantificadores

Ejemplo de predicado: P(x,y) = x es padre de y

Ejemplo de función: m(x) = “madre de x”

Cuantificadores: Existencial: $\exists x P(x)$

Universal: $\forall x P(x)$

Ejemplo: “*Todos los hombres son mortales, Sócrates es un hombre, luego Sócrates es mortal*”

$$\frac{\forall x(H(x) \rightarrow M(x)) \quad H(s)}{M(s)}$$

Jose Labra
Universidad de Oviedo

Lógica de predicados

Existen varios sistemas de demostración en lógica de predicados

Propiedades:

- **Consistente:** Todo lo que demuestran es correcto
- **Completo:** Todos lo que es correcto es demostrable
- **Semidecidible**: Si una fórmula es correcta, lo detectan, si no lo es, pueden no detectarlo

Para resolver ese problema se han buscado ***subconjuntos de lógica de predicados*** de primer orden que sean decidibles:

- Clausulas Horn
- Lógica descriptiva
- etc…

Jose Labra
Universidad de Oviedo

Complejidad

Los sistemas de demostración pueden ser muy complejos

Jerarquía de clases de complejidad

P ⊆ NP ⊆ PSPACE ⊆ EXPTIME ⊆ NEXPTIME ⊆ EXPSPACE

P = Problemas que resuelve una máquina de Turing determinista en tiempo polinómico

NP= Problemas que resuelve una máquina de Turing no determinista en tiempo polinómico

PSPACE= Problemas que resuelve una máquina de Turing determinista en espacio polinómico

EXPTIME=Problemas que resuelve una máquina de Turing determinista en tiempo $O(2^{p(n)})$

NEXPTIME=Problemas que resuelve una máquina de Turing no determinista en tiempo $O(2^{p(n)})$

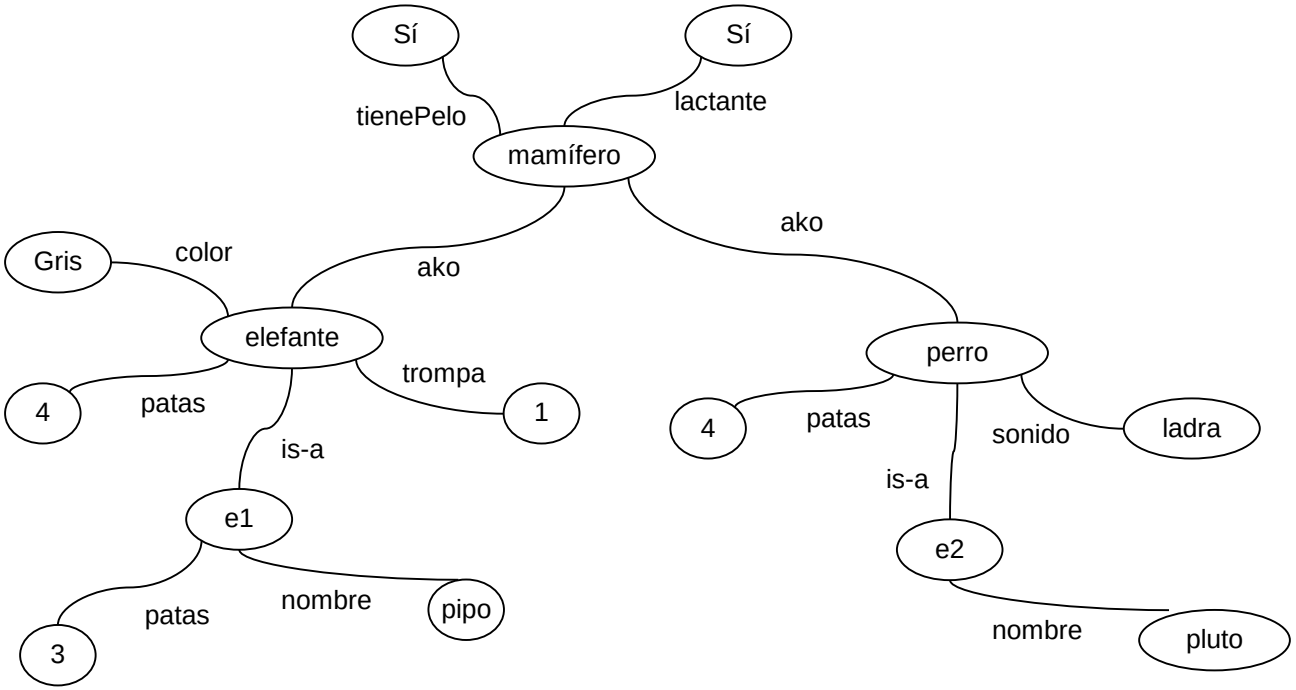
EXPSPACE=Problemas que resuelve una máquina de Turing determinista en espacio $O(2^{p(n)})$

Jose Labra
Universidad de Oviedo

Redes Semánticas

Redes Semánticas (Quillian, 68): Grafos dirigidos donde los vértices son conceptos y los enlaces son relaciones entre conceptos

2 tipos especiales de relaciones: is-a (pertenencia) y ako (inclusión)



Jose Labra
Universidad de Oviedo

Frames

Desarrollados para estructurar el conocimiento de las redes semánticas

Un *frame* o marco = colección de atributos (slots) que describen una entidad

Puede representar un concepto (o clase) y un individuo (o instancia)

```
Clase: Mamífero
         tienePelo: Sí
         lactante: Sí

Clase: Elefante
         ako: Mamífero
          patas: 4
          trompa: 1
         color: gris

Clase: Perro:
         ako: Mamífero
         patas: 4
         sonido: ladra
```

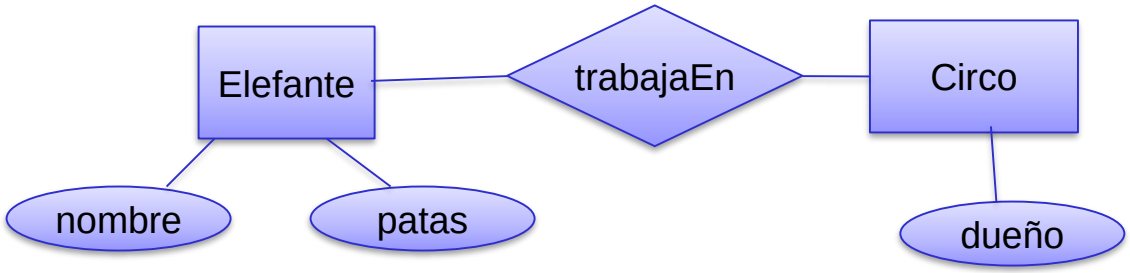
```
Individuo: e1
         isa: Elefante
         patas: 3
         nombre: Pipo


Individuo: e2
         is-a: Perro
         nombre: Pluto
```

Jose Labra
Universidad de Oviedo

Diagramas Entidad-Relación

Diagramas Entidad-Relación (Chen, 1976):
Representaciones gráficas utilizadas para capturar modelos de dominio.

Utilizados en el desarrollo de Bases de Datos



Jose Labra
Universidad de Oviedo

Mapas de Tópicos (Topic Maps)

Mapas de tópicos (http://www.topicmaps.org/)

- Estándar de definición de índices
- XTM es un vocabulario para mapas de tópicos basado en XML

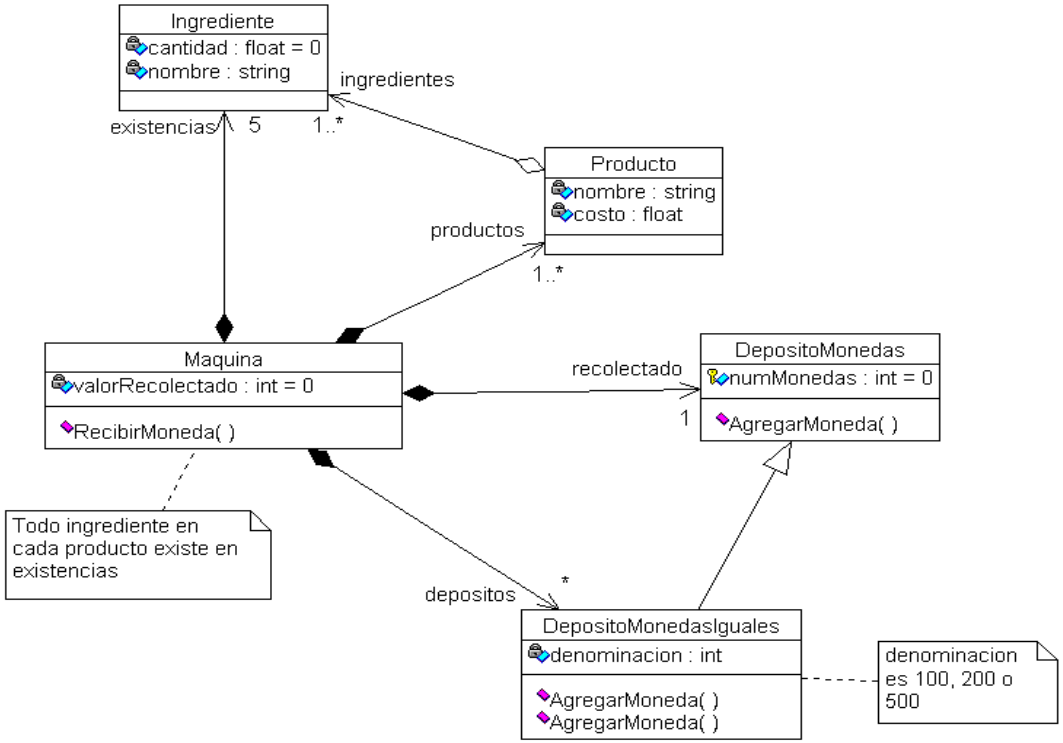
```
<topic id="pizzas"/> ...
 <occurrence>
  <instanceOf>
   <topicRef xlink:href="#barbacoa"/>
  </instanceOf>
  <scope>
    <topicRef xlink:href="#pizza"/>
   </scope>
   <resourceRef xlink:href="barbacoa.jpg"/>
 </occurrence>
 ...
</topic>
```

Jose Labra
Universidad de Oviedo

Modelos Orientados a Objetos

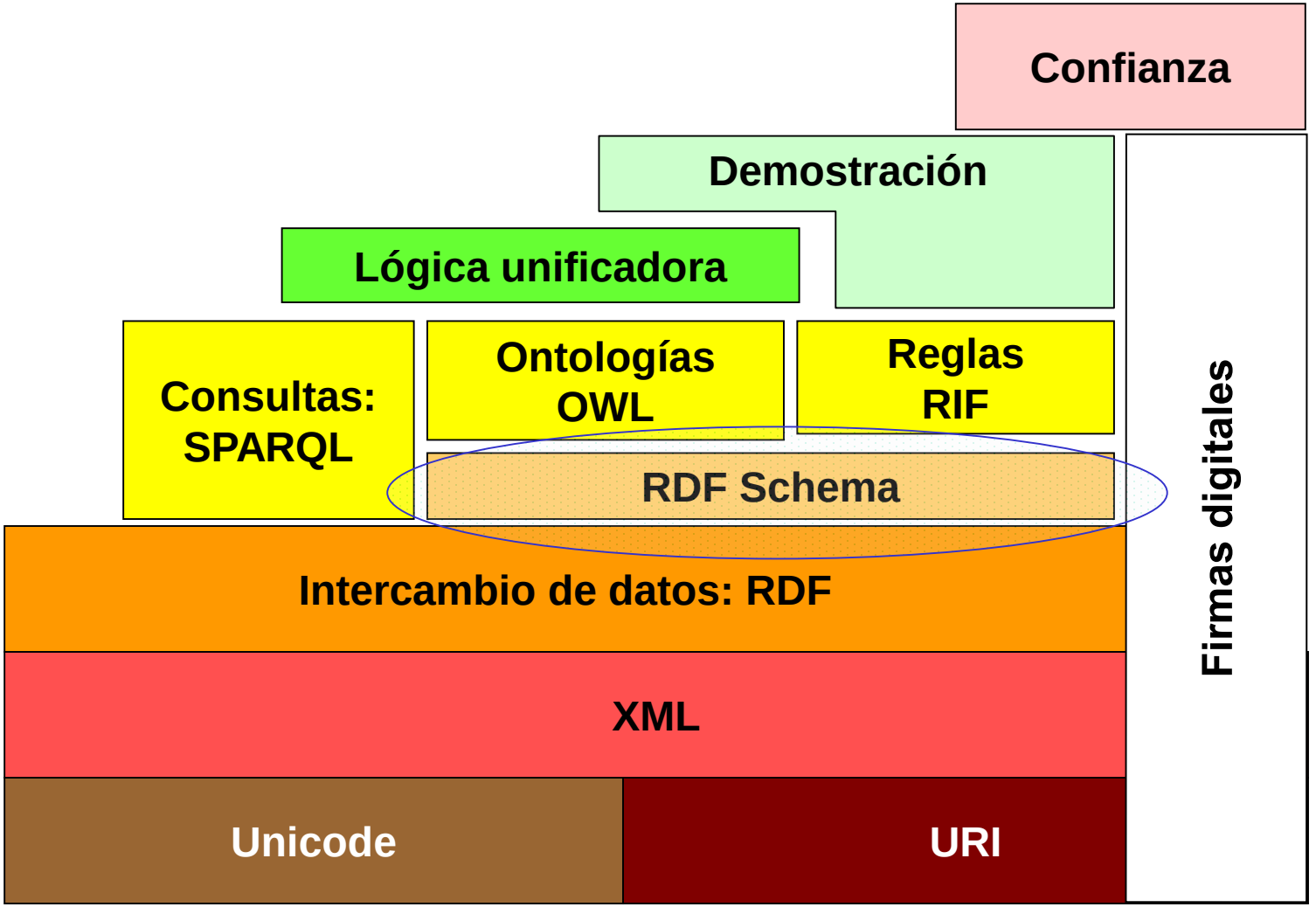
Modelos Orientados a Objetos: Especificación de herencia y jerarquía de objetos

Lenguajes de modelado. UML incluye diagramas de clase que describen la estructura de objetos, atributos, operaciones, etc.



Jose Labra
Universidad de Oviedo

RDF Schema



Jose Labra
Universidad de Oviedo

RDF Schema Motivación

RDF permite establecer propiedades pero no dice nada acerda de las propiedades

RDF Schema tiene varias clases y propiedades predefinidas que permiten definir vocabularios.

Es permite definir:

- Clases y propiedades
- Jerarquías y herencia entre clases
- Jerarquías de propiedades

Jose Labra
Universidad de Oviedo

RDF Schema
Clases e individuos

Hay que distinguir entre:

- Cosas concretas (individuos) del dominio.
  - Ej. "Jose Labra", "Lógica"
- Clases o conceptos = Conjuntos de individuos que comparten algunas propiedades (***rdfs:Class***)
  - Ej. "Profesor", "Asignatura", "Estudiante", ...

rdf:type indica que un individuo pertenece a una clase

rdfs:subClassOf indica que una clase está incluida en otra

Nota
rdf:type = <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
rdfs:Class = <http://www.w3.org/2000/01/rdf-schema#Class>

Jose Labra
Universidad de Oviedo

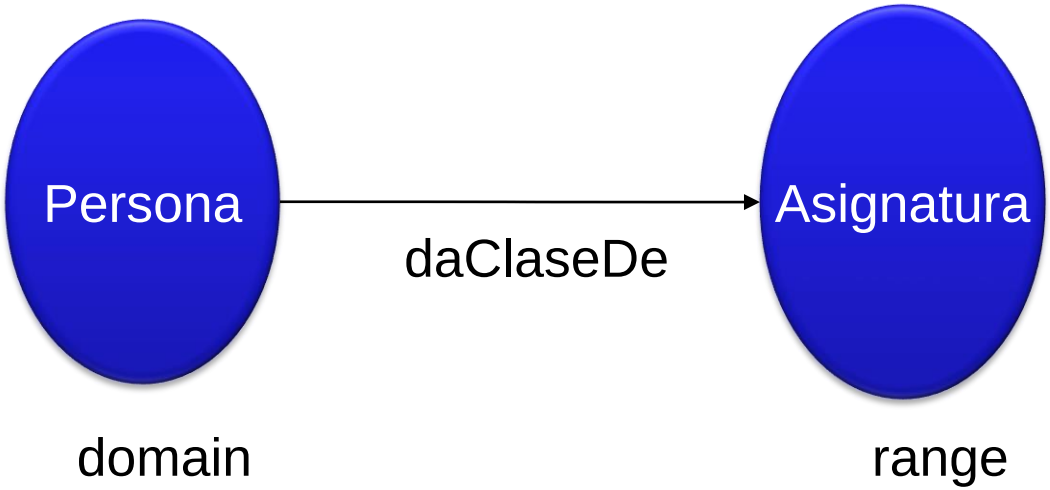
RDF Schema
Rango y Dominio

Pueden declararse restricciones de Rango y Dominio

Ejemplo: daClaseDe

rdfs:domain: Persona

rdfs:range: Asignatura



Jose Labra
Universidad de Oviedo

RDF Schema Jerarquías

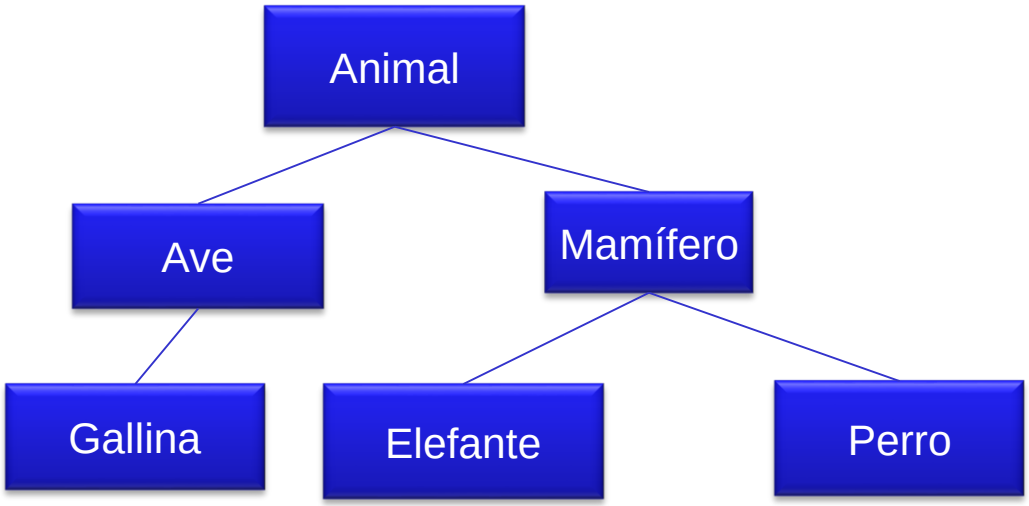
Las clases pueden organizarse en jerarquías

rdfs:subClassOf define que una clase es una subclase de otra

A es una subclase de B si todo individuo de A pertenece a B

Entonces, B es una superclase de A

Una clase puede tener múltiples superclases



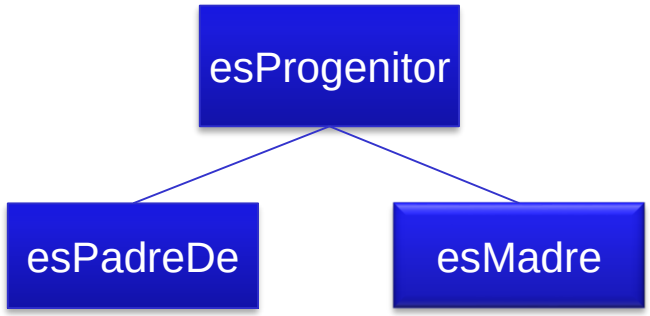
Jose Labra
Universidad de Oviedo

RDF Schema
Jerarquía de Propiedades

Jerarquías entre propiedades ***subPropertyOf***

Ej. Ser padre es una subpropiedad de ser progenitor

P es subpropiedad de Q $\equiv x\ P\ y \Rightarrow x\ Q\ y$




```
:juan :esPadreDe :ana ⇒ :juan :esProgenitorDe :ana
```

Jose Labra
Universidad de Oviedo

RDF Schema: Inferencias

RDF Schema tiene una semántica predefinida que permite inferir nuevas declaraciones a partir de las existentes

En realidad, se genera un grafo nuevo a partir del grafo anterior

Ejemplos:

X rdf:type A $\wedge$ A subClassOf B $\rightarrow$ X rdf:type B

A rdfs:subclassOf B $\wedge$ B rdfs:subClassOf C $\rightarrow$ A rdfs:subClassOf C

P rdfs:domain A $\wedge$ X P Y $\rightarrow$ X rdf :type A

P rdfs:range B $\wedge$ X P Y $\rightarrow$ Y rdf:type B

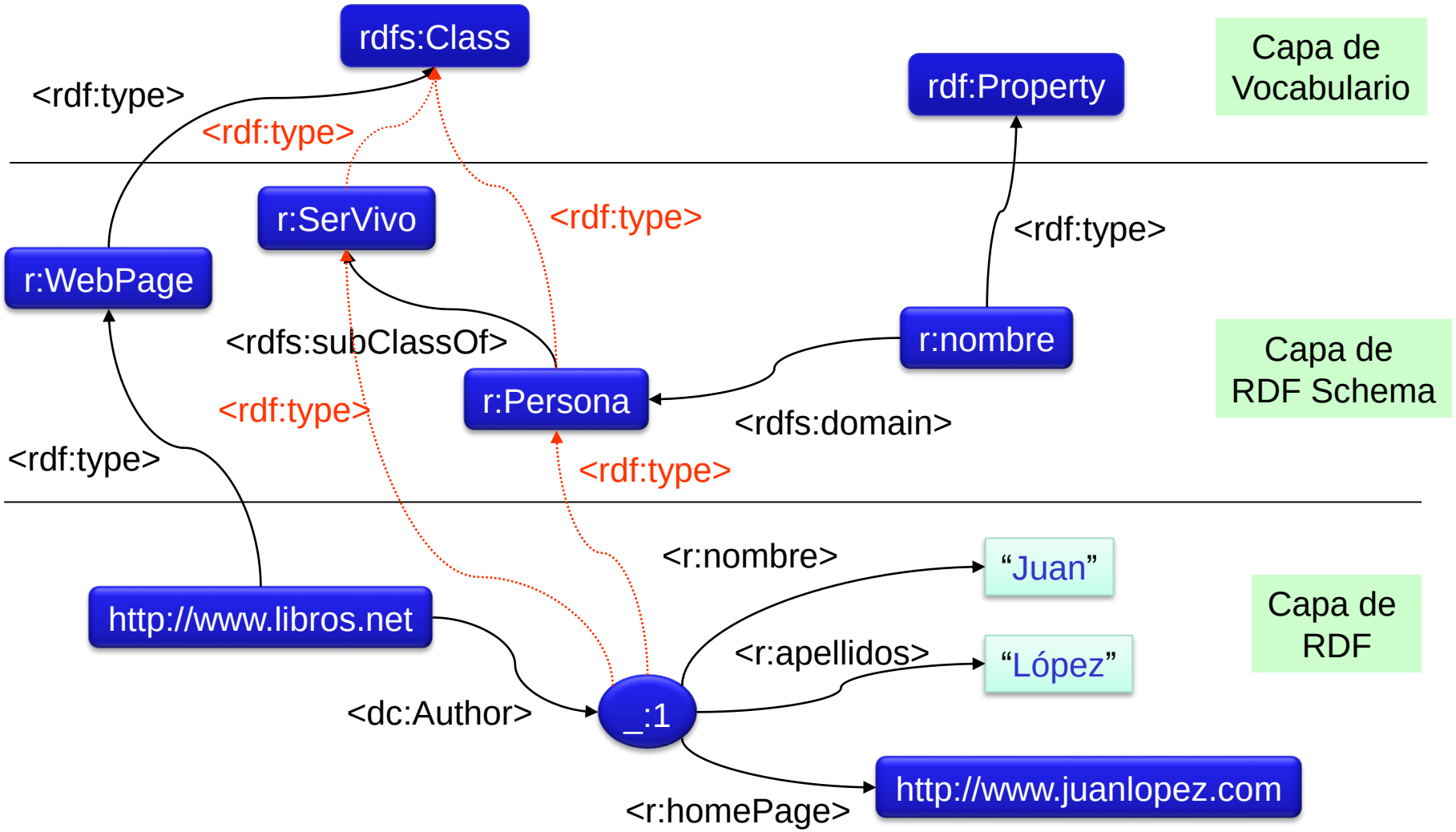
P rdfs:subPropertyOf Q $\wedge$ Q rdfs:subPropertyOf R $\rightarrow$ P rdf s:SubpropertyOf R

P rdfs:subPropertyOf Q $\wedge$ X P Y $\rightarrow$ X Q Y

etc.

Jose Labra
Universidad de Oviedo

RDF Schema: Inferencias



Jose Labra
Universidad de Oviedo

Ejercicio Barcos

Modelar el siguiente conocimiento en RDF

Nombre	Primer Viaje	Proximo Viaje	Fecha Baja	Fecha Hundimiento	Comandante
Pisco	1953		1978		Olmos
Rambo	1979	2012			Gallardo
Titanic	1912			1912	Smith
Sauce	1980	2013			Torre

Expresar el siguiente conocimiento

Un barco está ***fuera de servicio*** si tiene fecha de baja o fecha de hundimiento.

Un barco que tenga previsto un próximo viaje está ***en servicio***.

El comandante de un barco es un **marinero**.

Un marinero es una **persona**

La **fecha fin de servicio** es la fecha de baja o la fecha de hundimiento.

Jose Labra
Universidad de Oviedo

Ejercicio
Dominios y Rangos

En el ejercicio anterior declarar dominios y rangos de propiedades

primerViaje: Barco $\rightarrow$ Fecha

proximoViaje: Barco $\rightarrow$ Fecha

fechaBaja: Barco $\rightarrow$ Fecha

fechaHundimiento: Barco $\rightarrow$ Fecha

tieneComandante: Barco $\rightarrow$ Marinero

. . .

Jose Labra
Universidad de Oviedo

Añadiendo información

Crear una nueva tabla con la siguiente información

Nombre	Cargo	Proximo Viaje	Barco
Olmos	Comandante		Pisco
Gallardo	Comandante	2013	Alba
Ana	Azafata	2012	Rambo
Juan	Camarero	2013	Sauce

Preguntas:

Buscar todos los barcos

Añadir información de que Ana va a viajar en 2013 en Sauce

Buscar todos los próximos viajes de barcos o personas

Jose Labra
Universidad de Oviedo

Limitaciones de RDF Schema

- RDF Schema sólo permite declarar información clases y propiedades
- Es un primer paso hacia las ontologías pero se queda corto
- Carece de expresividad para:
  - Información negativa
    - Los hombres no son mujeres
  - Cuantificadores
    - Para que alguien sea considerado padre debe tener al menos un hijo
  - Cardinalidad
    - Un buen estudiante tiene que tener aprobadas más de 3 asignaturas
  - No permite atributos de propiedades
    - Transitiva, simétrica, inversa, etc.

Jose Labra
Universidad de Oviedo

Problema de RDF Schema

RDF Schema es demasiado liberal permitiendo mezclar clases con individuos y propiedades

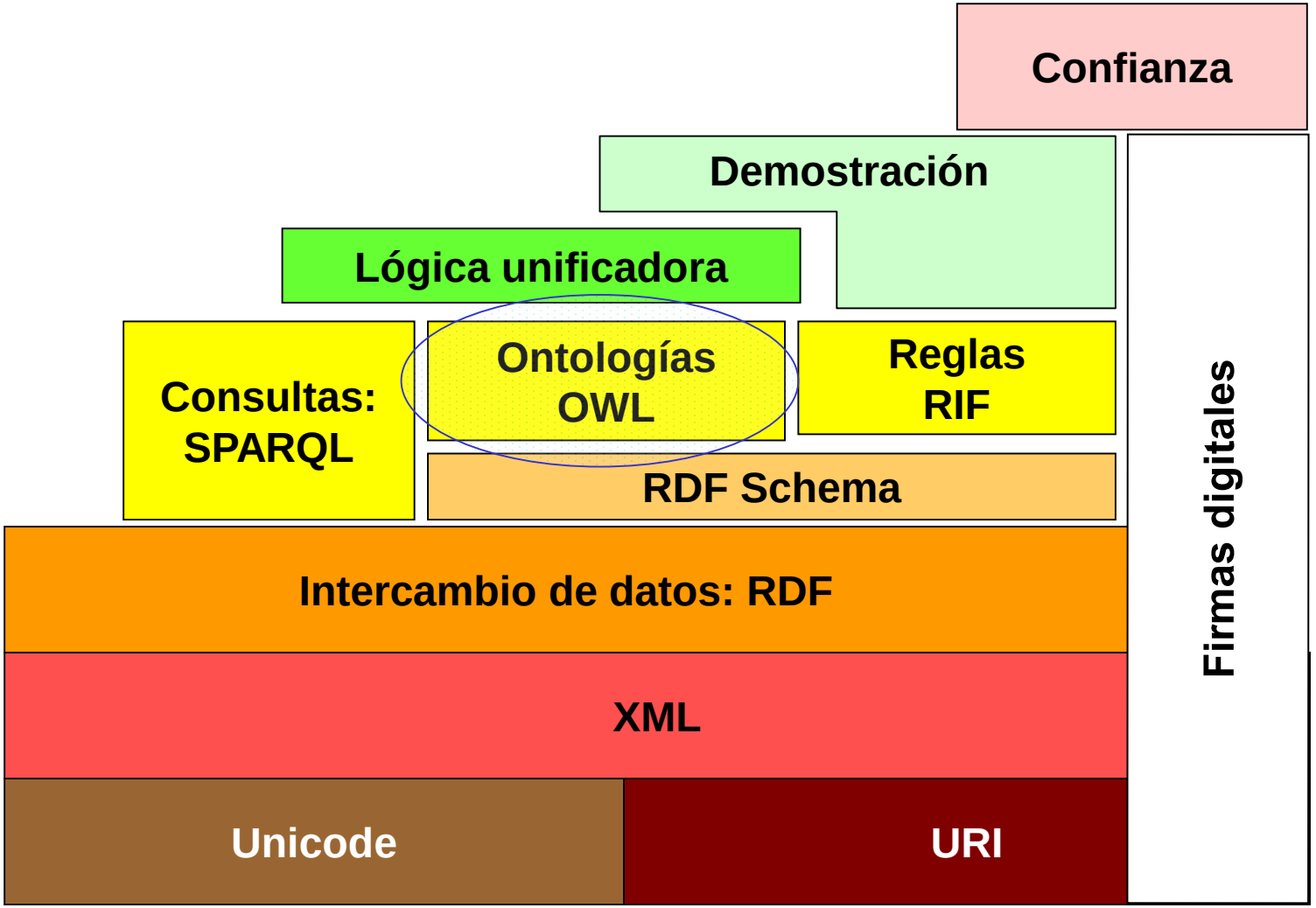
Por ejemplo:

x rdf:type x

Pueden llegar a declararse paradojas en RDF Schema

Jose Labra
Universidad de Oviedo

OWL



Jose Labra
Universidad de Oviedo

OWL

OWL = Web Ontology Language

Desarrollado a partir de iniciativa del W3c

- Antecedentes: DAML, OIL
- Se basa en lógica descriptiva
- 2004 - OWL 1.0 recomendación W3c
- 2009 - OWL 2.0 recomendación W3c

Jose Labra
Universidad de Oviedo

Ejemplos de Ontologías

Cyc (http://www.cyc.com).

- Conceptos de sentido común para Inteligencia Artificial
- Utiliza lógica de predicados mediante lenguaje CycL

Frame Ontology y OKBC Ontology

- Disponibles en Ontolingua (http://www-ksl-svc.stanford.edu/)
- Utiliza KIF (Knowledge Interchange Format)

Ontologías en campos concretos:

- Lingüística: WordNet (http://www.globalwordnet.org/)
- Medicina: GALEN (http://www.opengalen.org/)
- etc.

Jose Labra
Universidad de Oviedo

Ejemplos de Ontologías
Dublin Core

- Dublin Core Metadata Initiative (http://www.dcmi.org)
- Utilizado para la catalogación de documentos
- Espacio de nombres: http://purl.org/dc/elements/1.1/
- Conjunto de elementos básicos cuyo significado es compartido
  - Contenido: Coverage, Description, Type, Relation, Source, Subject, Title
  - Propiedad Intelectual: Contributor, Creator, Publisher, Rights
  - Instanciación: Date, Format, Identifier, Language
- Cada elemento básico admite una serie de cualificadores
  - Refinamiento de elementos
    - Ejemplo: Date.created, Description.tableOfContents
  - Esquema de codificación
    - Ejemplos: Identifier.URI, Date.DCMIPeriod

Jose Labra
Universidad de Oviedo

Evolución de las Ontologías para la Web

- SHOE (Simple HTML Ontology Extensions) Univ. Maryland, 1996
  - Permite definir ontologías en documentos HTML
  - Objetivo = Facilitar búsquedas y anotaciones de documentos
- OIL (Ontology Inference Layer)
  - Sintaxis RDF(S) y primitivas de representación del conocimiento en marcos
  - Se basa en el uso de *description logics*
- DAML (DARPA Agent Markup Language)
  - Proyecto americano de creación de lenguaje para ontologías
- DAML-OIL. Proyecto conjunto que será la base de OWL
- OWL (Web Ontology Language) desarrollado en W3C (2004)

Jose Labra
Universidad de Oviedo

OWL

OWL (Web Ontology Language)

Desarrollado por el consorcio W3C (2004)

3 niveles:

- OWL Full. Unión de sintaxis OWL y RDF (sin restricciones)
  - No se garantiza la eficiencia ni siquiera la decidibilidad
- OWL DL (Description Logics). Limita la expresividad intentando conseguir decidibilidad
- Profiles. Subconjuntos de OWL DL: EL, QL, RL, etc.
  - Más eficientes, menos expresivos

Jose Labra
Universidad de Oviedo

OWL

- OWL DL se basa en Lógica Descriptiva (Description Logics)
  - En realidad equivale al formalismo $\mathcal{SHOIN}(\mathcal{D}n)$
- Características
  - Semántica bien definida
  - Propiedades formales (decidibilidad, complejidad)
  - Algoritmos de razonamiento conocidos
  - Varios razonadores (Pellet, HermiT, Quonto)
  - Incluye tipos de datos primitivos de XML Schema

Jose Labra
Universidad de Oviedo

Lógica Descriptiva

La lógica descriptiva consiste en:

- Conceptos (o clases):
  - Ejemplo: Padre, Madre, Persona
- Propiedades (o roles): Relaciones entre conceptos
  - Ejemplo: tieneHijo, esPadreDe
- Individuos: Elementos del dominio
  - Ejemplo: Juan, Sergio, …

Jose Labra
Universidad de Oviedo

Lógica Descriptiva

La lógica descriptiva es un subconjunto de la lógica de primer orden

Características:

Sólo se usan predicados de un argumento (clases) y dos argumentos (propiedades)

El uso de las variables está restringido

Jose Labra
Universidad de Oviedo

Lógica Descriptiva

La base de conocimiento contiene 2 niveles

Términos (TBox): Descripción de conceptos

$Padre \equiv Persona \cap \exists\ tieneHijo\ Persona$

$Orgulloso \equiv Persona \cap \exists\ tieneHijo\ Recién Nacido$

$Recién Nacido \subseteq Persona$

Aserciones (ABox): Descripción de individuos

Recién Nacido(Sergio)
tieneHijo(Jose,Sergio)
Persona(Jose)

Jose Labra
Universidad de Oviedo

Lógica descriptiva

Las expresiones en lógica descriptiva pueden representarse en lógica de primer orden

Padre $\equiv$ Persona $\cap \exists$ tieneHijo Persona

$\forall x(Padre(x) \leftrightarrow (Persona(x) \wedge \exists y(tieneHijo(x,y) \wedge Persona(y)))$

Orgulloso $\equiv$ Persona $\cap \exists$ tieneHijo ReciénNacido

$\forall x(Orgulloso(x) \leftrightarrow (Persona(x) \wedge \exists y(tieneHijo(x,y) \wedge RecienNacido(y)))$

ReciénNacido $\subseteq$ Persona

$\forall x(RecienNacido(x) \rightarrow Persona(x))$

Jose Labra
Universidad de Oviedo

Lógica Descriptiva
Definición de Conceptos

Definición de conceptos

- Equivalencia: $C \equiv D$
  - Ejemplo: Asturiano $\equiv$ NacidoEnAsturias
- Subclase: $C \subseteq D$ (C está incluido en D ó D subsume a C)
  - Ejemplo: Asturiano $\subseteq$ Español
- Intersección: $C \cap D$
  - Ejemplo: Mujer $\equiv$ Persona $\cap$ Femenino
- Unión: $C \cup D$
  - Ejemplo: Persona $\equiv$ Hombre $\cup$ Mujer
- Complemento: $\neg C$
  - Ejemplo: Masculino $\equiv \neg$ Femenino
- Concepto vacío: $\bot$
- Clases Disjuntas: $C \cap D \equiv \bot$

Jose Labra
Universidad de Oviedo

Lógica Descriptiva
Cuantificadores

Descripción de Propiedades

Existencial ($\exists$ R C)

x pertenece a $\exists$ R C si existe algún valor y $\in$ C tal que R(x,y)

Ejemplo: Madre $\equiv$ Mujer $\cap$ $\exists$ tieneHijo Persona

Universal ($\forall$ R C)

x pertenece a $\forall$ R C si para todo y, si R(x,y) entonces y $\in$ C

Ejemplo: MadreFeliz $\equiv$ Madre $\cap$ $\forall$ tieneHijo Sano

Una Madre es feliz si todos sus hijos están sanos

NOTA: Si no tuviese hijos, también se cumpliría…

Jose Labra
Universidad de Oviedo

Lógica Descriptiva
Cardinalidades

Cardinalidad (P = n)

x pertenece a (P = n) si existen n y $\in$ C tales que R(x,y)

Ejemplo: Elefante $\subseteq$ Animal $\cap$ tienePatas = 4

Cardinalidad máxima (P $\leq$ n)

x pertenece a (P $\leq$ n) si existen n ó menos y $\in$ C tales que R(x,y)

Ejemplo: MalEstudiante $\equiv$ Estudiante $\cap$ tieneAprobada $\leq$ 3

Cardinalidad mínima (P $\geq$ n)

x pertenece a (P $\geq$ n) si existen n ó más y $\in$ C tales que R(x,y)

Ejemplo: BuenEstudiante $\equiv$ Estudiante $\cap$ tieneAprobada $\geq$ 3

$$\forall x(\text{BuenEstudiante}(x) \leftrightarrow \text{Estudiante}(x) \wedge \exists y_1 \exists y_2 \exists y_3 (\text{tieneAprobada}(x, y_1) \wedge \text{tieneAprobada}(x, y_2) \wedge \text{tieneAprobada}(x, y_3) \wedge y_1 \neq y_2 \wedge y_2 \neq y_3 \wedge y_1 \neq y_3))$$

Jose Labra
Universidad de Oviedo

Lógica Descriptiva
Atributos de propiedades

Reflexiva: P es reflexiva $\Rightarrow \forall x P(x,x)$

Ejemplo: viveCon es reflexiva

Irreflexiva: P es irreflexiva $\Rightarrow \forall x \neg P(x,x)$

Ejemplo: esPadreDe es irreflexiva

Simetría. Si P(x,y) entonces P(y,x)

Ejemplo: viveCon

Asimétrica. Si P(x,y) entonces $\neg P(y,x)$

Ejemplo: esPadreDe

Transitividad. Si P(x,y) y P(y,z) entonces P(x,z)

Ejemplo: viveCon

Jose Labra
Universidad de Oviedo

Lógica descriptiva
Relaciones entre propiedades

Inversa: P es inversa de Q $\Rightarrow$ P(x,y) $\Leftrightarrow$ Q(y,x)

Ejemplo: daClaseDe es inversa de tieneProfesor

SubPropiedad: P subpropiedad de Q si P(x,y) $\Rightarrow$ Q(x,y)

Ejemplo: esHijoDe es subpropiedad de esDescendienteDe

Jose Labra
Universidad de Oviedo

Lógica Descriptiva
Funcionalidad

Propiedad Funcional.

P(x,y) y P(x,z) entonces y = z

Ejemplo: edad

Propiedad Funcional inversa.

P(x,y) y P(z,y) entonces x = z

Ejemplo: dni

Claves. similares a las propiedades func. inversas

P(x,y) y P(z,y) entonces x = z.

Se definen en OWL 2 (específicas para una clase)

Ejemplo: dni

Jose Labra
Universidad de Oviedo

Lógica Descriptiva Razonamiento

A partir de una base de conocimiento Σ se ofrecen varios mecanismos de inferencia:

1.- Satisfacibilidad de conceptos: De Σ no se deduce que C $\equiv \bot$

Ejemplo: Orgulloso $\cap$ ReciénNacido

2.- Subsunción: Deducir si un concepto está incluido en otro

$\Sigma \Rightarrow C \subseteq D$

Ejemplo: Orgulloso $\subseteq$ Padre

Padre $\equiv$ Persona $\cap \exists$ tieneHijo Persona Orgulloso $\equiv$ Persona $\cap \exists$ tieneHijo ReciénNacido ReciénNacido $\subseteq$ Persona Padre $\subseteq \neg$ReciénNacido
ReciénNacido(Sergio) tieneHijo(Jose,Sergio) Persona(Jose)

Jose Labra
Universidad de Oviedo

Lógica Descriptiva Razonamiento

3.- Instanciación: $\Sigma \Rightarrow a \in C$

Ejemplo: Orgulloso(Jose)

4.- Recuperación de Información

Dado un concepto C, obtener a tales que $a \in C$

Ejemplo: ? Orgulloso

Jose

5.- Realización/Comprensión (realizability).

Dado un elemento a, obtener concepto más específico C tal que $a \in C$

Ejemplo: ? jose

Orgulloso

Padre ≡ Persona ∩ ∃ tieneHijo Persona Orgulloso ≡ Persona ∩ ∃ tieneHijo ReciénNacido ReciénNacido ⊆ Persona Padre ⊆ ¬ReciénNacido
ReciénNacido(Sergio) tieneHijo(Jose,Sergio) Persona(Jose)

Jose Labra
Universidad de Oviedo

Lenguaje OWL

Jose Labra
Universidad de Oviedo

OWL
Definición de ontologías

Aunque OWL es un lenguaje basado en lógica descriptiva, existen varias sintaxis:

- RDF (N3 ó RDF/XML)
- SIntaxis abstracta
- Sintaxis Manchester

Espacio de nombres:

- http://www.w3.org/2002/07/owl#

Cabecera

- Declarar una URI de la clase owl:Ontology

```
@prefix : <http://ejemplo.org#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

<>    a                    owl:Ontology .
```

Jose Labra
Universidad de Oviedo

OWL
Cabecera

En la cabecera pueden incluirse diversas anotaciones

Anotaciones posibles:

rdfs:label	Etiqueta del recurso
rdfs:comment	Comentario del recurso
owl:versionInfo	Información sobre la versión del recurso
rdfs:seeAlso	Indica otro recurso con más información
rdfs:isDefinedBy	Indica que otro recurso contiene la definición
owl:imports	Indica la URI de una ontología que se va a incorporar a la ontología actual

```
@prefix : <http://ejemplo.org#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.

:    a                    owl:Ontology ;
     rdfs:comment "Ejemplo de Ontologia" ;
     owl:imports     <http://paises.org> .
```

Jose Labra
Universidad de Oviedo

OWL
Definiciones básicas

```
:juan     a     :Alumno  ;
          :nombre "Juan Manuel"  ;
          :apellidos "Gallardo"  ;
          :esAmigoDe  :pepe .

:pepe     a     :Alumno  ;
          :nombre "Jose Luis" ;
          :apellidos "Torres"  ;
          :tieneProfesor :jj .

:jj       a          :Profesor ;
          :nombre "Juan Jose" ;
          :apellidos "Bravo" .
```

Jose Labra
Universidad de Oviedo

OWL
Subclases

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.

:Profesor  rdfs:subClassOf   :Persona .
:Alumno   rdfs:subClassOf   :Persona .
```

Existen 2 clases predefinidas:

owl:Thing - Contiene a todos los individuos

owl:Nothing - representa el conjunto vacío

Jose Labra
Universidad de Oviedo

OWL
Propiedades

2 tipos de propiedades:

ObjectProperty - relaciona individuos

Ejemplo: :daCLaseDe

DatatypeProperty - relaciona un individuo con un valor

Ejemplo: nombre, edad

Jose Labra
Universidad de Oviedo

OWL
Dominio y rango

Las definiciones de dominio y rango se toman de RDF Schema

Dominio: Conjunto inicial

Rango: Conjunto final

```
:daClaseDe  rdfs:domain  :Profesor .
:daClaseDe  rdfs:range     :Asignatura .
```

Nota: Las definiciones de dominio y rango pueden introducir poca flexibilidad en el modelo

Jose Labra
Universidad de Oviedo

Relaciones entre propiedades

rdfs:subPropertyOf: P es subpropiedad de Q, si P(x,y) entonces Q(x,y)

```
:esPadreDe  rdfs:subPropertyOf   :esProgenitor .
```

owl:inverseOf: P es inversa de Q si P(x,y) $\Rightarrow$ Q(y,x)

```
:esPadreDe  owl:inverseOf   :esHijoDe .
```

Jose Labra
Universidad de Oviedo

Propiedades disjuntas

P es disjunta de Q si $P(x,y) \Rightarrow \neg Q(x,y)$

Puede definirse propiedad a propiedad:

```
:tienePadre  owl:propertyDisjointWith   :tieneMadre .
```

...o para varias propiedades a la vez:

```
[]    a owl:AllDisjointProperties ;
      owl:members ( :tienePadre ,
                    :tieneMadre ) .
```

Jose Labra
Universidad de Oviedo

Tipos de propiedades

owl:SymmetricProperty: $P(x,y) \Rightarrow P(y,x)$

Ejemplo: viveCon

owl:ASymmetricProperty: $P(x,y) \Rightarrow \neg P(y,x)$

Ejemplo: esPadreDe

owl:ReflexiveProperty: Para todo x, P(x,x)

Ejemplo: viveCon

owl:IrreflexiveProperty: Para todo x, $\neg$ P(x,x)

Ejemplo: esPadreDe

owl:TransitiveProperty: $P(x,y) \wedge P(y,z) \Rightarrow P(x,z)$

Ejemplo: viveCon

```
:viveCon a owl:SymmetricProperty,
                owl:ReflexiveProperty,
                owl:TransitiveProperty.
```

Jose Labra
Universidad de Oviedo

Propiedades funcionales

owl:FunctionalProperty: P(x,y) y P(x,z) $\Rightarrow$ y = z

Ejemplo: tieneMadre

owl:InverseFunctionalProperty: P(x,y) y P(z,y) $\Rightarrow$ x = z

- Ejemplos: esMadreDe, dni

```
:tieneMadre a owl:FunctionalProperty.
```

owl:hasKey: Define una clave para una clase.

Una clave es un conjunto de valores de propiedades que identifican únicamente a un elemento.

```
:Persona owl:hasKey (:dni :nombre).
```

Jose Labra
Universidad de Oviedo

Restricciones de propiedades Existencial

owl:someValuesFrom: Al menos un valor debe pertenecer a una clase

```
:Padre  owl:equivalentClass  [
   a                          owl:Restriction ;
   owl:onProperty             :tieneHijo ;
   owl:someValuesFrom  :Persona
 ] .
```

Notación en lógica descriptiva: $Padre \equiv Persona \cap \exists\ tieneHijo\ Persona$

Notación en lógica de predicados:

$$\forall x(Padre(x) \leftrightarrow (Persona(x) \wedge \exists y(tieneHijo(x,y) \wedge Persona(y))))$$

Jose Labra
Universidad de Oviedo

Restricciones de propiedades Universal

owl:allValuesFrom: Todos los valores deben pertenecer a una clase

```
[]  a owl:Class ;
    owl:intersectionOf ( :Person  :Feliz) ;
    owl:equivalentClass [ a owl:Restriction ;
                          owl:onProperty  :hasChild ;
                          owl:allValuesFrom  :Feliz ] .
```

Notación en lógica descriptiva:

$$\text{Persona} \cap \text{Feliz} \equiv \forall\ \text{tieneHijo Feliz}$$

Notación en lógica de predicados:

$$\forall x(\text{Persona}(x) \wedge \text{Feliz}(x) \leftrightarrow \forall y(\text{tieneHijo}(x,y) \rightarrow \text{Feliz}(y)))$$

Jose Labra
Universidad de Oviedo

Restricciones de propiedades Valores

owl:hasValue: Debe contener un valor determinado

```
:HijosDeJuan owl:equivalentClass
        [ a owl:Restriction ;
          owl:onProperty  :esHijoDe ;
          owl:hasValue  :Juan
        ].
```

Notación en lógica de predicados:

$$\forall x(HijosDeJuan(x) \leftrightarrow esHijoDe(x,Juan))$$

Jose Labra
Universidad de Oviedo

Cardinalidades

owl:cardinality: N valores exactos

owl:minCardinality: Al menos N valores

owl:maxCardinality: N valores como mucho

```
:Persona owl:equivalentClass
        [ a owl:Restriction ;
          owl:onProperty  :esHijoDe ;
          owl:cardinality   "2"^^xsd:nonNegativeInteger ;
        ].
```

Jose Labra
Universidad de Oviedo

Cardinalidades cualificadas

owl:qualifiedCardinality: N valores exactos de una clase

owl:minQualifiedCardinality: Al menos N valores de una clase

owl:maxQualifiedCardinality: N valores como mucho de una clase

```
:DosHijosDoctores owl:equivalentClass
        [ a owl:Restriction ;
          owl:onProperty  :esPadreDe ;
          owl:qualifiedCardinality  "2"^^xsd:nonNegativeInteger ;
          owl:onClass :Doctor
        ].
```

Jose Labra
Universidad de Oviedo

Conjuntos por enumeración

owl:oneOf define un conjunto de valores

```
:Estacion a owl:Class ;
   owl:OneOf (
          :Primavera
          :Otonio
          :Verano
          :Invierno
    ) .
```

Jose Labra
Universidad de Oviedo

Operaciones de conjuntos

owl:unionOf define la unión de varias clases

```
:Progenitor owl:equivalentClass
   [ a owl:Class ;
     owl:unionOf ( :Padre :Madre )
   ] .
```

owl:intersectionOf define la intersección de unas clases

```
:Madre owl:equivalentClass    [ a owl:Class ;
        owl:intersectionOf ( :Progenitor :Mujer )
   ] .
```

owl:complementOf

```
:PersonaSinHijos owl:equivalentClass
   [ a owl:Class ;
     owl:intersectionOf ( :Persona [ owl:complementOf :Progenitor] )
   ] .
```

Jose Labra
Universidad de Oviedo

Clases disjuntas

owl:disjointWith

```
:Persona     owl:disjointWith :Asignatura .
:Asignatura owl:disjointWith :Ciudad .
:Ciudad       owl:disjointWith :Asignatura .
```

owl:allDisjointClasses

```
[] a owl:AllDisjointClasses ;
  owl:members (:Persona :Asignatura :Ciudad)
```

Jose Labra
Universidad de Oviedo

Equivalencia

owl:sameAs establece que 2 individuos son el mismo

owl:equivalentClass establece que 2 clases son la misma

owl:equivalentProperty establece que 2 propiedades son la misma

Jose Labra
Universidad de Oviedo

Individuos diferentes

owl:differentFrom

```
:Juan owl:differentFrom :Pepe .
:Juan owl:differentFrom :Luis .
:Luis owl:differentFrom :Pepe .
```

owl:AllDifferent

```
[] a owl:AllDifferent ;
   owl:distinctMembers (:Juan :Pepe :Luis) .
```

Jose Labra
Universidad de Oviedo

Declaraciones negativas

owl:negativePropertyAssertion permite declarar que no se cumple una propiedad

Hay que indicar la propiedad, el sujeto y el objeto.

Ejemplo: Declara que Juan no es padre de Ana

```
[] a owl:NegativePropertyAssertion ;
   owl:sourceIndividual :Juan ;
   owl:assertionProperty :esPadreDe ;
   owl:targetIndividual :Ana .
```

Jose Labra
Universidad de Oviedo

Cadenas de propiedades

owl:propertyChainAxiom define una cadena de propiedades

```
:esAbueloDe owl:propertyChainAxiom
      ( :esPadreDe
        :esPadreDe
      ) .

:esTioDe owl:propertyChainAxiom
      ( :esHermanoDe
        :esPadreDe
      ) .
```

Jose Labra
Universidad de Oviedo

Restricciones self

Define la clase de todos los individuos que se relacionan consigo mismos mediante una propiedad

```
:Autonomo a owl:Class;
    owl:equivalentClass [
       a owl:SelfRestriction ;
       owl:onProperty :alimentaA
    ].
```

Jose Labra
Universidad de Oviedo

OWL
Sintaxis XML

Jose Labra
Universidad de Oviedo

OWL
Sintaxis XML

OWL se basa en RDF (utiliza sintaxis XML de RDF)

También existen otras formas sintácticas más sencillas

Las ontologías comienzan por owl:Ontology

```
<owl:Ontology rdf:about="http://www.uniovi.es/ontologia_1.1">
    <rdfs:comment>Ejemplo de Ontología</rdfs:comment>
    <owl:priorVersion
        rdf:resource="http://www.uniovi.es/ontologia_1.0"/>
    <owl:imports
        rdf:resource="http://www.uniovi.es/personas"/>
    <rdfs:label>Ontología de la Universidad</rdfs:label>
</owl:Ontology>
```

owl:imports es una propiedad transitiva

Jose Labra
Universidad de Oviedo

Clases en OWL

Las clases se definen mediante owl:Class

owl:Class es una subclase de rdfs:Class

Clases equivalentes mediante equivalentClass

```
<owl:Class rdf:ID="Profesor">
  <owl:equivalentClass rdf:resource="#PersonalDocente"/>
</owl:Class>
```

owl:Thing es la clase más general

owl:Nothing es la clase vacía

Las clases disjuntas se definen mediante owl:dijointWith

```
<owl:Class rdf:about="#ProfesorAsociado">
  <owl:disjointWith rdf:resource="#catedrático"/>
  <owl:disjointWith rdf:resource="#titular"/>
</owl:Class>
```

Jose Labra
Universidad de Oviedo

Propiedades en OWL

2 tipos de propiedades

Propiedades de Objetos relacionan un objeto con otro objeto. ej. "esHijoDe"

```
<owl:ObjectProperty rdf:ID="esHijoDe">
   <owl:domain rdf:resource="#Persona"/>
   <owl:range rdf:resource= "#Persona"/>
     <rdfs:subPropertyOf rdf:resource="#esDescendienteDe"/>
</owl:ObjectProperty>
```

Propiedades de tipos de datos relacionan un objeto con valores de tipos de datos (enteros, literales, etc.), ej. "edad"

Habitualmente, se utilizan los tipos de datos de XML Schema

```
<owl:DatatypeProperty rdf:ID="edad">
  <rdfs:range
      rdf:resource="http://www.w3.org/2001/XLMSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

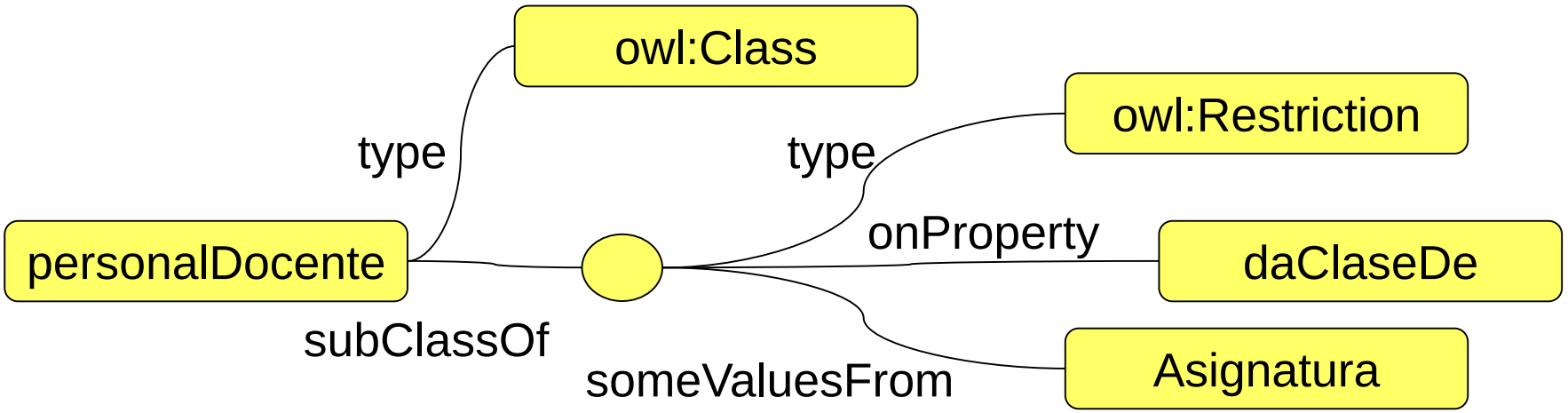
Jose Labra
Universidad de Oviedo

Definición de Clases

Clases como restricciones de propiedades

```
<owl:Class rdf:about="#personalDocente">
    <rdfs:subClassOf>
      <owl:Restriction>
          <owl:onProperty rdf:resource="#daClaseDe"/>
          <owl:someValuesFrom rdf:resource="#Asignatura"/>
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

personalDocente $\subseteq \exists$ daClaseDe Asignatura



Jose Labra
Universidad de Oviedo

Propiedades en OWL Restricciones

allValuesFrom (∀) indica que todos los valores deben ser de un tipo

NOTA: Los que no tiene ningún valor, también cumplen la condición

someValuesFrom (∃) Al menos un valor de la propiedad debe tener un tipo

Ejemplo: Un estudiante es una persona que cursa al menos una asignatura

hasValue Al menos uno de los valores tiene un valor

minCardinality, maxCardinality restringen el número máximo/mínimo de valores

Jose Labra
Universidad de Oviedo

Propiedades en OWL
Combinaciones booleanas

Combinaciones booleanas

complementOf, unionOf, intersectionOf

```
<owl:Class rdf:ID="personasUniversidad">
        <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#personalDocente"/>
                <owl:Class rdf:about="#estudiantes"/>
                <owl:Class rdf:about="#PAS"/>
        </owl:unionOf>
</owl:Class>
```

Jose Labra
Universidad de Oviedo

Propiedades en OWL
Enumeraciones

oneOf permite realizar enumeraciones

```
<owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Lunes"/>
        <owl:Thing rdf:about="#Martes"/>
        <owl:Thing rdf:about="#Miércoles"/>
        <owl:Thing rdf:about="#Jueves"/>
        <owl:Thing rdf:about="#Viernes"/>
        <owl:Thing rdf:about="#Sábado"/>
        <owl:Thing rdf:about="#Domingo"/>
</owl:oneOf>
```

Jose Labra
Universidad de Oviedo

Individuos en OWL

Se declaran igual que en RDF

```
<rdf:Description rdf:ID="jose">
    <rdf:type rdf:resource= "#profesor"/>
</rdf:Description>
```

```
<personalDocente rdf:ID="jose">
   <uni:edad rdf:datatype="&xsd;integer">35<uni:edad>
</personalDocente>
```

Jose Labra
Universidad de Oviedo

OWL – Web semántica
No asume nombres únicos

Web = modelo abierto

- Información incompleta
- 2 URIs diferentes podrían identificar el mismo objeto
- No soporta UNA (Unique name assumption)
- Permite **inferir** que 2 elementos son iguales
- No está pensado para validar modelos

Ejemplo

$Persona \subset tienePadre = 1$

```
tienePadre(luis,jose)
tienePadre(luis,pepe)
Persona(luis)
```

No indica error en el modelo

Infiere que “pepe” y “jose” son iguales

Jose Labra
Universidad de Oviedo

OWL – Web Semántica
Asumción de mundo abierto

Web = Sistema abierto

Sistemas tradicionales usaban *closed world assumption*

En OWL se usa *open world assumption*

Ejemplo

$Soltero \subseteq \neg \exists\ estaCasadoCon\ Persona$
$Casado \subseteq \exists\ estaCasadoCon\ Persona$

Persona(pepe)
Persona(Maria)
Persona(luis)
estaCasadoCon(maria,pepe)
Casado(luis)

El sistema infiere que María está casada

El sistema no infiere que pepe esté casado ni soltero

El sistema infiere que luis Está casado con alguien… pero no sabe con quién.

Jose Labra
Universidad de Oviedo

OWL Herramientas

Herramientas para manipulación de documentos OWL

Protègè (http://protege.stanford.edu) es una herramienta para creación de ontologías desarrollada en Stanford (se basa en Frames)

Arquitectura que facilita el desarrollo de plugins

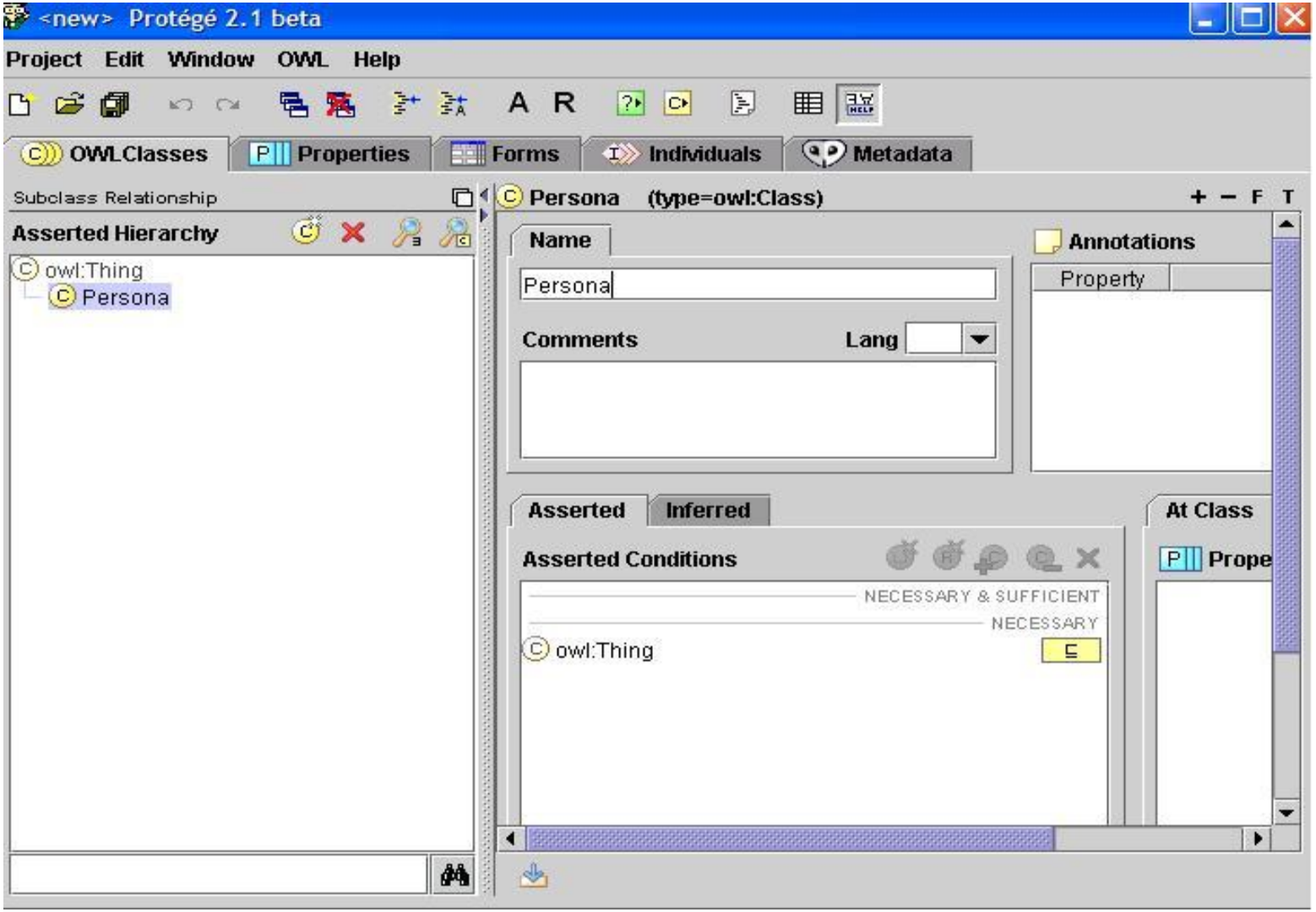
Plugin para edición de documentos OWL

Swoop: Herramienta inspirada en un visualizador web con la posibilidad de editar ontologías

TopBraid (Comercial)

Jose Labra
Universidad de Oviedo

OWL Herramientas



Jose Labra
Universidad de Oviedo

Ejercicio. Resolver el Problema de Einstein en OWL

Problema propuesto por Einstein y traducido a varios idiomas conservando su lógica. Einstein aseguraba que el 98% de la población mundial sería incapaz de resolverlo.

Condiciones iniciales:

- **Tenemos cinco casas, cada una de un color.**
- **Cada casa tiene un dueño de nacionalidad diferente.**
- **Los 5 dueños beben una bebida diferente, fuman marca diferente y tienen mascota diferente.**
- **Ningún dueño tiene la misma mascota, fuma la misma marca o bebe el mismo tipo de bebida que otro.**

Datos:

1. **El noruego vive en la primera casa, junto a la casa azul.**
2. **El que vive en la casa del centro toma leche.**
3. **El inglés vive en la casa roja.**
4. **La mascota del Sueco es un perro.**
5. **El Danés bebe té.**
6. **La casa verde es la inmediata de la izquierda de la casa blanca.**
7. **El de la casa verde toma café.**
8. **El que fuma PallMall cría pájaros.**
9. **El de la casa amarilla fuma Dunhill.**
10. **El que fuma Blend vive junto al que tiene gatos.**
11. **El que tiene caballos vive junto al que fuma Dunhill.**
12. **El que fuma BlueMaster bebe cerveza.**
13. **El alemán fuma Prince.**
14. **El que fuma Blend tiene un vecino que bebe agua.**

¿Quién tiene peces por mascota?

Jose Labra
Universidad de Oviedo

OWL
Sistemas de Inferencia

HermiT (Java) razonador OWL 2

Pellet (Java) incluye razonador para OWL 2

- Soporte en línea de comandos o mediante interfaz DIG

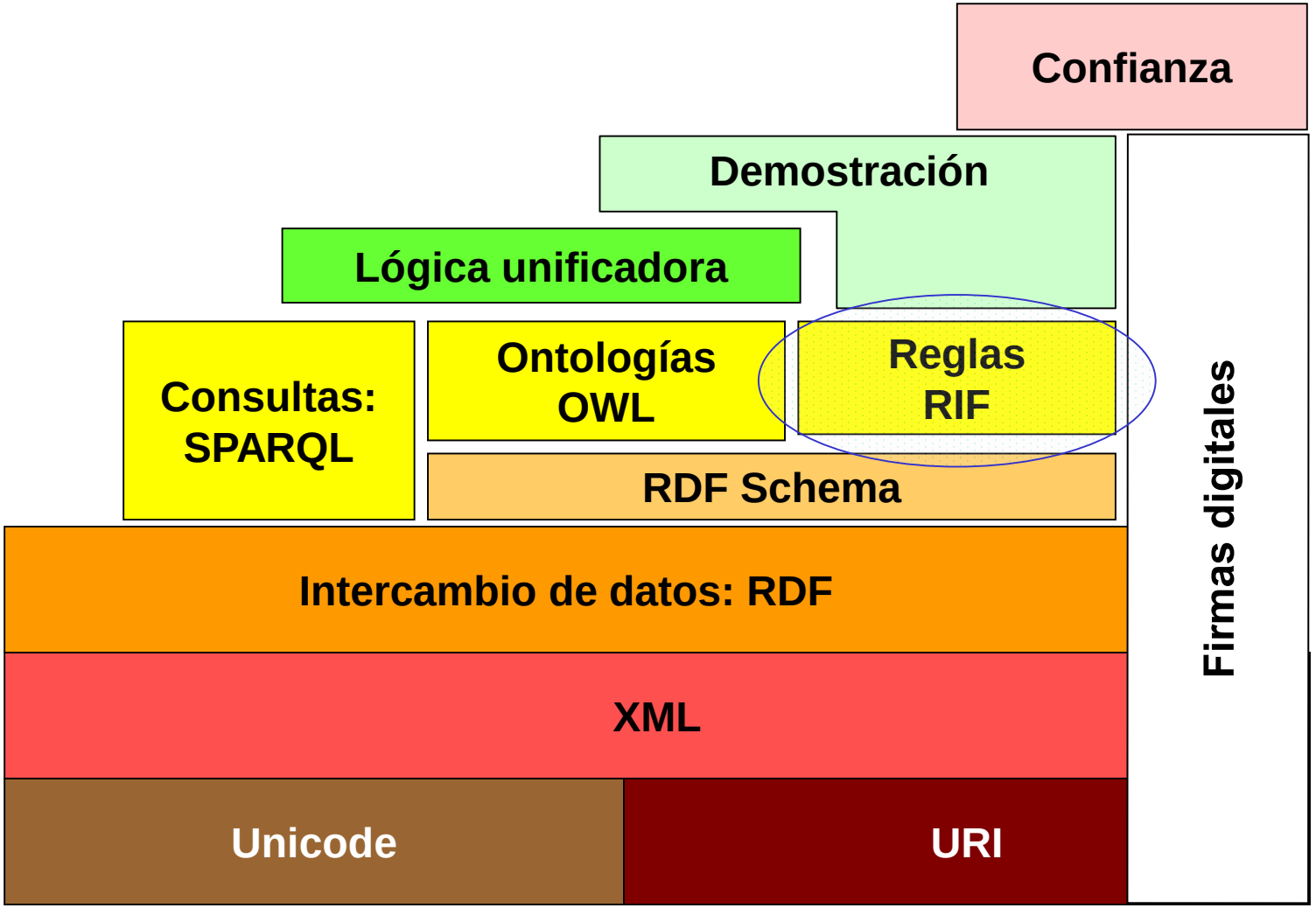
Fact++ (C++) razonador

RACER. Sistema de inferencia implementado en Lisp

JENA. API Java para RDF. Incluye sistema de inferencia

Jose Labra
Universidad de Oviedo

RIF: Reglas



Jose Labra
Universidad de Oviedo

Ampliaciones Reglas

Las Cláusulas Horn son la base de la programación lógica

SWRL (Semantic Web Rule Language) es una propuesta de creación de un lenguaje de reglas que añada reglas tipo Prolog a OWL

Orígenes: RuleML (Iniciativa internacional)

RIF (Rule Interchange Format) = Intercambio de reglas

$$\textbf{hermano(x,y)} \wedge \textbf{progenitor(y,z)} \rightarrow \textbf{tío(x,z)}$$

$$\textbf{hermano . progenitor} \subseteq \textbf{tío}$$

Problema: Indecidible al unirse con OWL

Jose Labra
Universidad de Oviedo

Monotonicidad y Reglas

Problema: negación por fallo

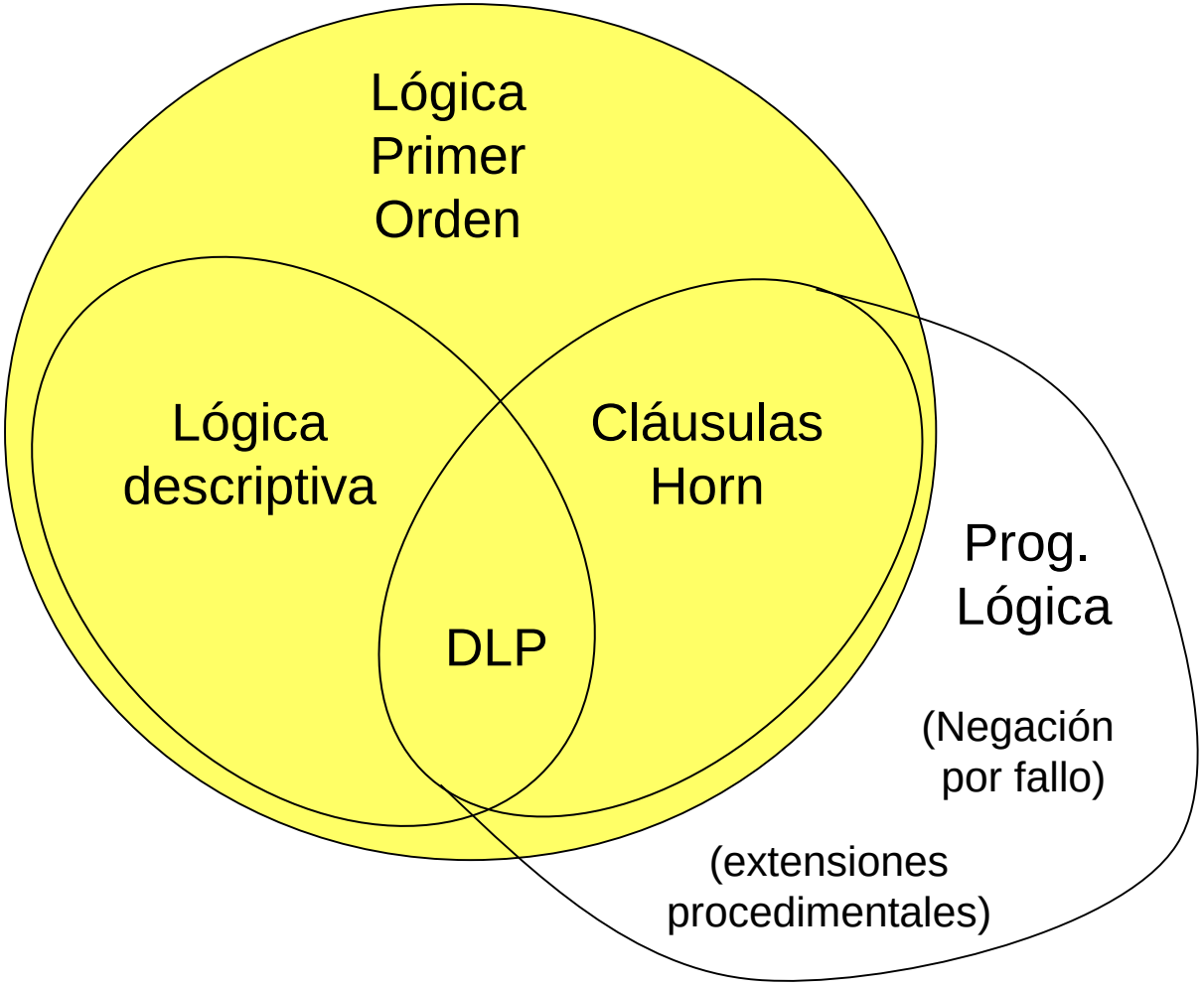
Lógica de primer orden es monótona

Negación por fallo no es monótona

Programación lógica con negación por fallo no es un subconjunto de lógica de primer orden

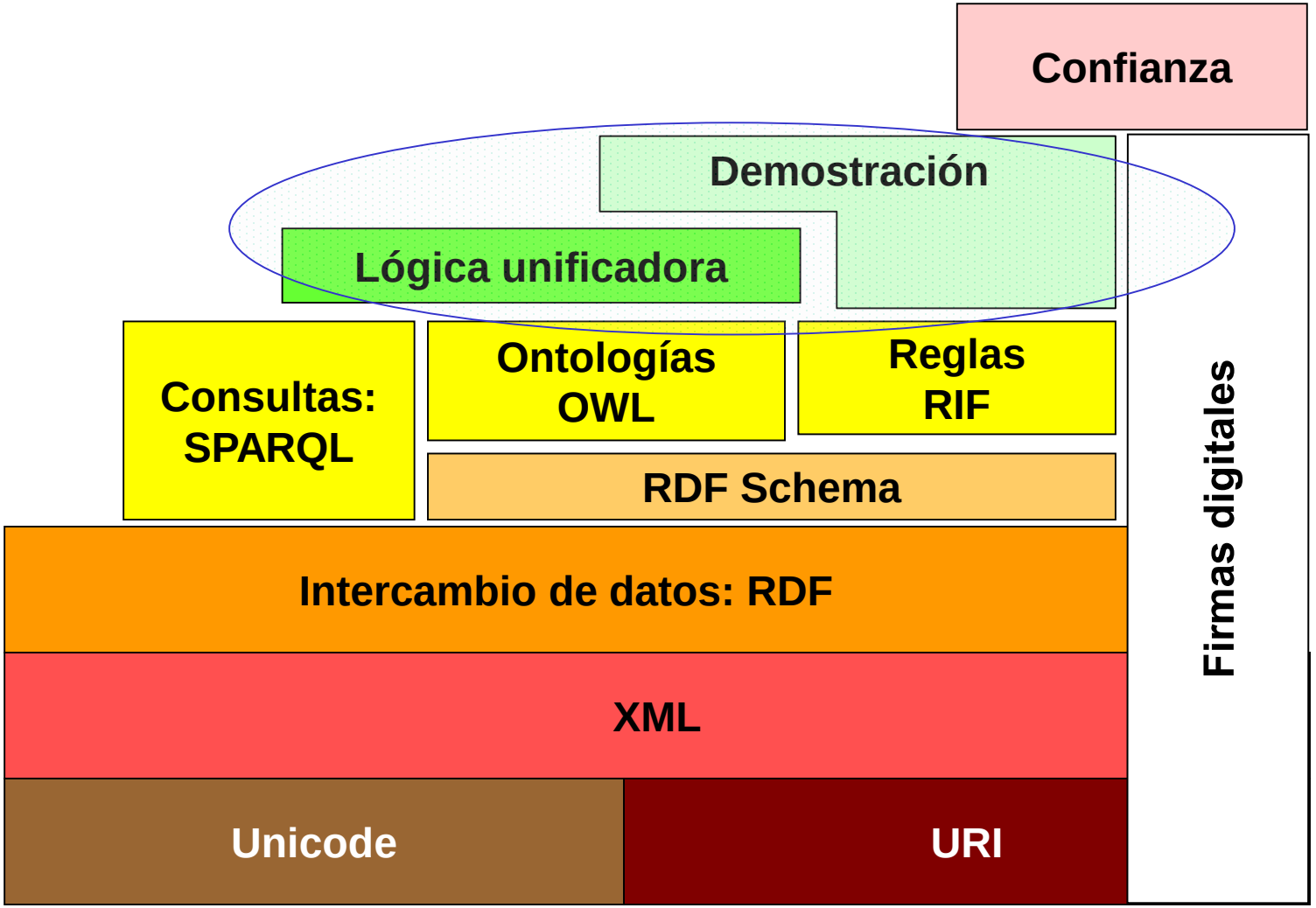
Jose Labra
Universidad de Oviedo

Interacción entre programación lógica y lógica descriptiva



Jose Labra
Universidad de Oviedo

Lógica y Demostración



Jose Labra
Universidad de Oviedo

Sistema de Inferencia Semantic Tableaux

- Semantic Tableaux
  - Detecta si es insatisfacible
  - Va buscando modelos que cumplan las condiciones
  - Va generando un árbol de posibles modelos
  - Cierra las ramas cuando encuentra inconsistencias (clash)
- Expresividad vs. Decidibilidad
  - Sopa de letras

http://www.cs.man.ac.uk/~ezolin/logic/complexity.html

Jose Labra
Universidad de Oviedo

Semantic Tableaux
Ejemplo

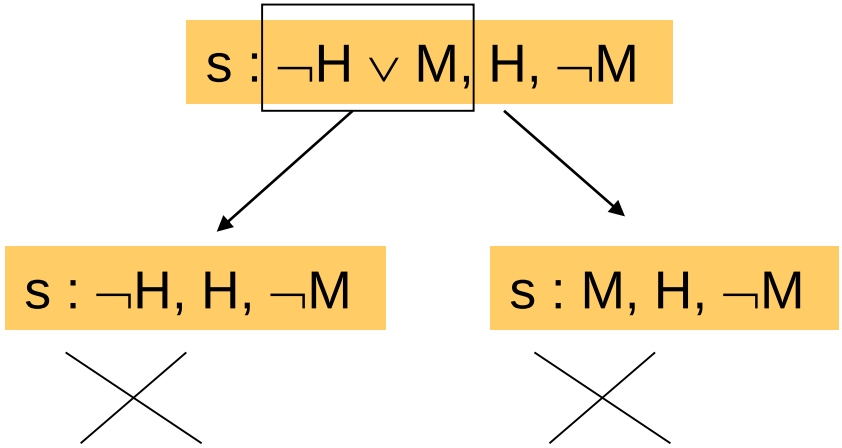
Hombre $\subseteq$ Mortal

~~Hombre(Sócrates)~~

Mortal(Sócrates)

Razonamiento: $\{ H \subseteq M, H(s) \} \Rightarrow M(s)$

Forma normal: $\{ \neg H \vee M, H(s), \neg M(s) \}$



Regla de inferencia:
Si A contiene x : C $\vee$ D entonces
A' = A $\cup$ x : C
A' = A $\cup$ x : D

Jose Labra
Universidad de Oviedo

Semantic Tableaux
Algunas Reglas de inferencia

Si A contiene x : C $\vee$ D entonces
A' = A $\cup$ x : C
A' = A $\cup$ x : D

Si A contiene x : C $\wedge$ D entonces
A' = A $\cup$ x : C, D

Si A contiene x : $\exists$ R C y $\nexists$z tal que R(x,z) y z: C entonces
A' = A $\cup$ { y : C, R(x,y) } para un y $\notin$A

Si A contiene x : $\forall$ R C y R(x,y) pero no contiene y : C
A' = A $\cup$ y : C

Jose Labra
Universidad de Oviedo

Semantic Tableaux
Otro ejemplo

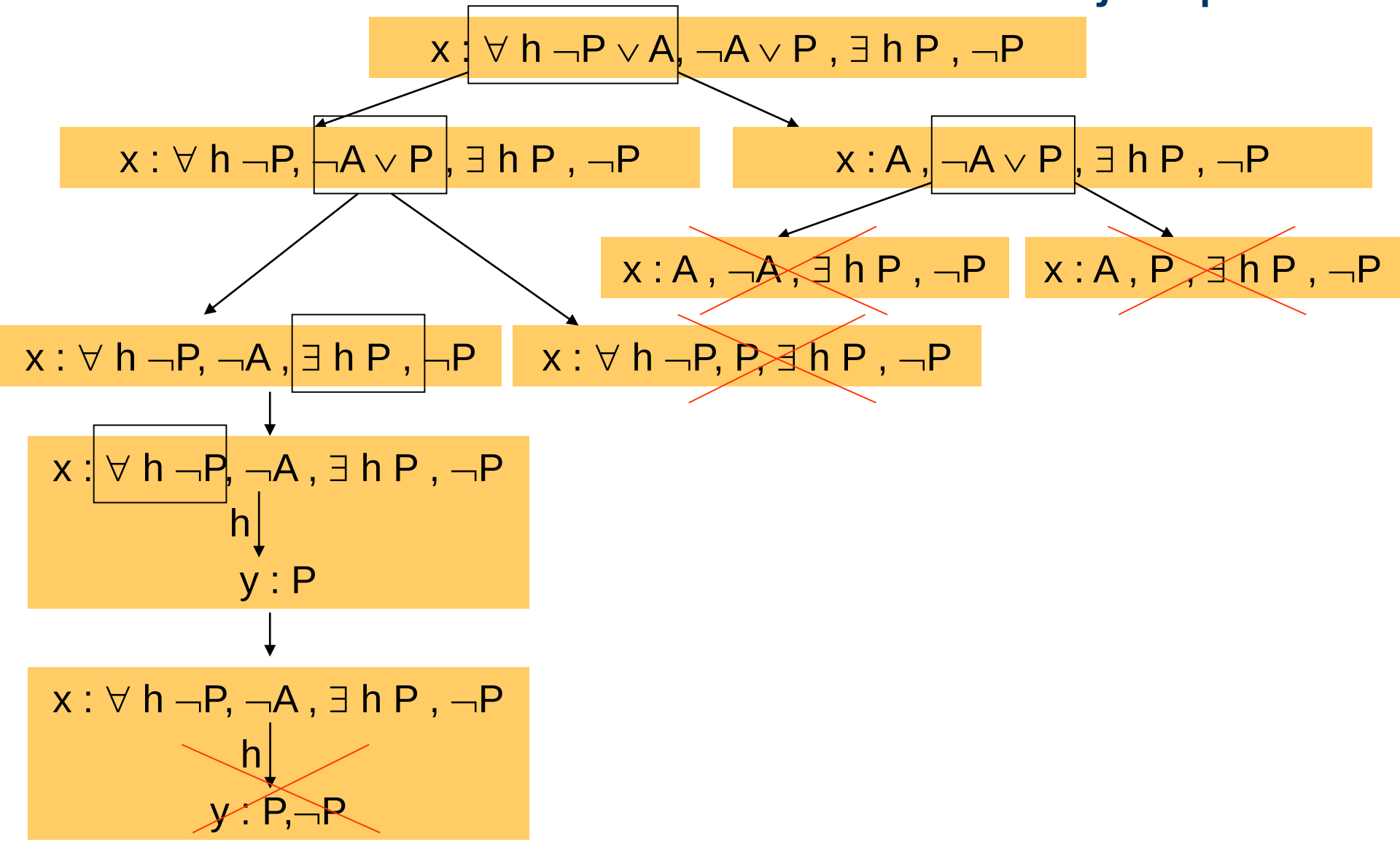
$\{ \exists \text{ hijo Persona} \subseteq \text{Padre}, \text{Padre} \subseteq \text{Persona} \} \Rightarrow \exists \text{ hijo Persona} \subseteq \text{Persona}$

Cambiando nombres: $\{ \exists \text{ h P} \subseteq \text{A}, \text{A} \subseteq \text{P} \} \Rightarrow \exists \text{ h P} \subseteq \text{P}$

Forma nornal: $\{ \forall \text{ h } \neg\text{P} \vee \text{A}, \neg\text{A} \vee \text{P} , \exists \text{ h P} , \neg\text{P} \}$

Jose Labra
Universidad de Oviedo

Semantic Tableaux
Otro ejemplo



Jose Labra
Universidad de Oviedo

Semantic Tableaux
Otro ejemplo

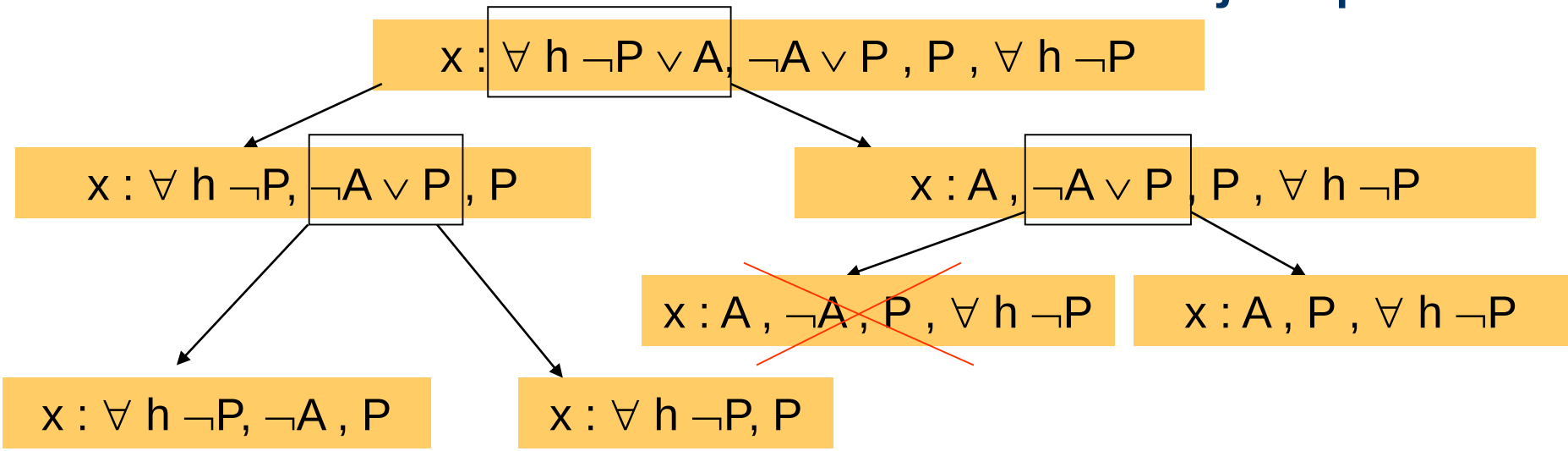
$\{ \exists \text{ hijo Persona} \subseteq \text{Padre}, \text{Padre} \subseteq \text{Persona} \} \Rightarrow \text{Persona} \subseteq \exists \text{ hijo Persona}$

Cambiando nombres: $\{ \exists \text{ h P} \subseteq \text{A}, \text{A} \subseteq \text{P} \} \Rightarrow \text{P} \subseteq \exists \text{ h P}$

Forma nornal: $\{ \forall \text{ h } \neg\text{P} \vee \text{A}, \neg\text{A} \vee \text{P} , \text{P} , \forall \text{ h } \neg\text{P} \}$

Jose Labra
Universidad de Oviedo

Semantic Tableaux
Otro ejemplo



Se encuentra un modelo ⇒ No se cumple

Otros tutoriales

OWL2 and SWRL Tutorial

http://dior.ics.muni.cz/~makub/owl/

Jose Labra
Universidad de Oviedo

Fin de la Presentación

NOTA: *Esta presentacion se difunde unicamente con fines divulgativos.*
Las imagenes utilizadas pueden pertenecer a terceros y por tanto son propiedad de sus autores.