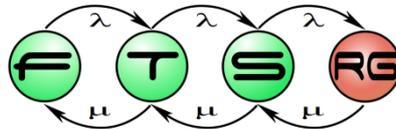




Ontologies and Semantic Technologies

Izsó Benedek

Bergmann Gábor



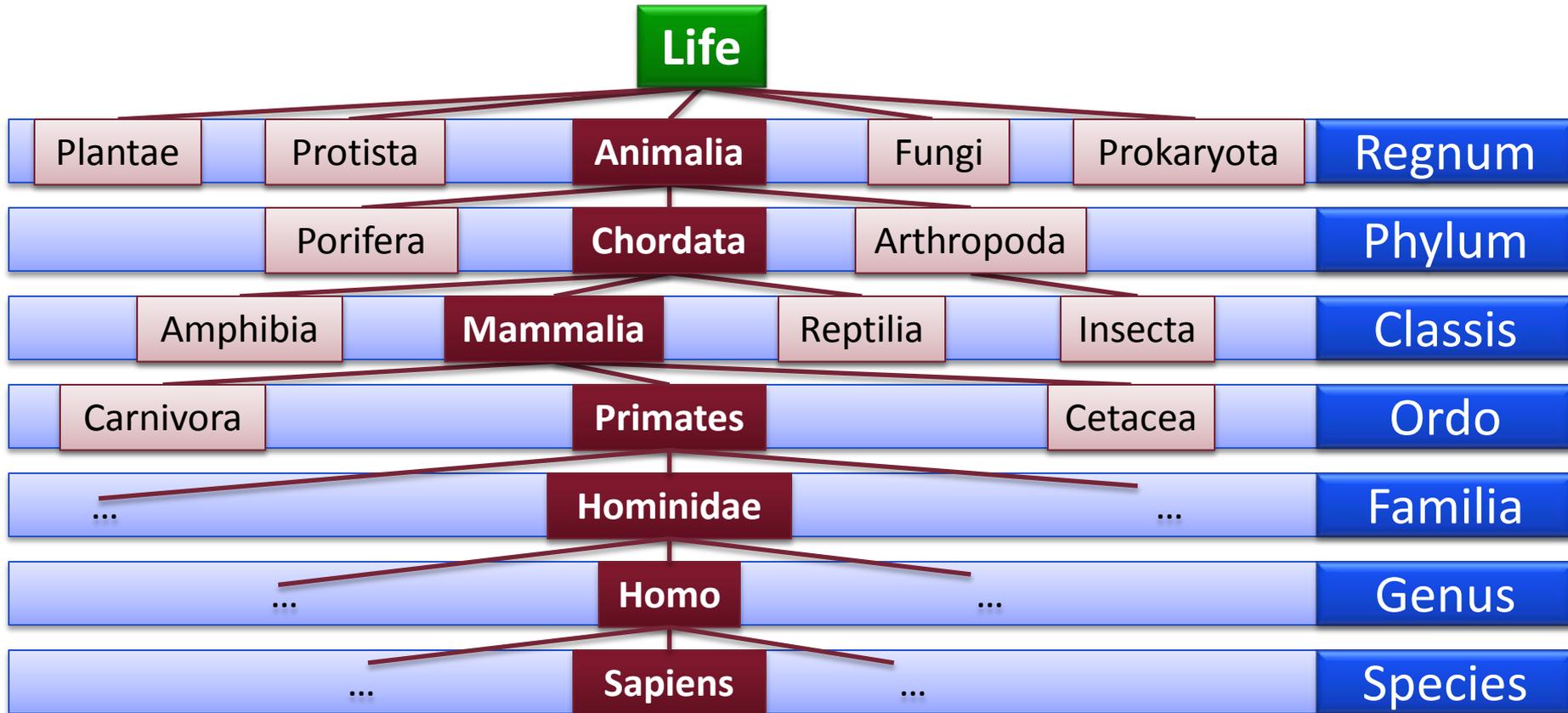
Agenda

- Ontologies
 - Resource Description Framework (RDF)
 - Web Ontology Language (OWL)
- Semantic Technologies and Resources
- Modeling approaches

ONTOLOGIES

Taxonomy

- **Taxonomy** = hierarchy of domain concepts



- **Ontology** \cong taxonomy + relationships + definitions

Ontology

- **Ontology** = „the study of existence”
- Computer representation of **domain knowledge**
 - Identifying **concepts** to categorize **individuals**
 - **Relationships** that can hold between individuals
 - **Axioms** on concepts and their relationships
 - Including **taxonomy** of domain concepts (supertypes)
- Created by
 - **Domain experts, knowledge engineers**
 - People using annotation tools
 - Natural language text analysis tools

Domain Ontologies

- Open Biological and Biomedical Ontologies (OBO)
 - Chemical information
 - Cells, cell types, proteins, etc.
 - Anatomy (Upper/Human/...)
 - Medical software, imaging methods, spectrometry etc.
- National Center for Biomedical Ontology (NCBO)
 - National Drug File
 - International Classification of Diseases
 - SNOMED Clinical Terms
- data.gov (public access to US government data)
 - Documents categorized in an ontology

Open World Assumption

Can we enumerate all diseases?

- Traditional databases have Closed World Assumption
 - E.g. if not explicitly listed as a disease, then not a disease
- Most ontologies: **Open World Assumption (OWA)**
 - Not proven true/false → not treated as false or true
 - Why? Ontologies can never be complete
 - Examples
 - E.g. if not listed / implied as a viral disease, still can be one
 - Patient 42 has lepers. Does Patient 42 have a flu? Unknown!
 - Patient 2501 died of lepers. Did she die of flu? No!
(by multiplicity 1 of cause of death)

Unique Name Assumption

- Can two identifiers correspond to the same thing?
 - Patient 42 carries skin disease31.
 - Disease5 of patient 42 was found to be of viral nature.
 - Are they two different diseases? (→ unknown)
- Two things can be the same, unless contradicted
 - disjoint classes (hereditary and viral disease)
 - Patient 42 has a hereditary disease
 - explicit control: owl:sameAs, owl:differentFrom
- Usually **NO Unique Name Assumption**
- Why? Distributed knowledge gathering

Unique Name Assumption

- Can two identifiers correspond to the same thing?
 - Patient 42 carries skin disease31.
 - Disease5 of patient 42 was found to be of viral nature.
 - Are they two different diseases? (→ unknown)
- Two things can be the same, unless contradicted
 - disjoint classes
 - Patient 42 has
 - explicit contradiction
- Usually **NO Unique Name Assumption**
- Why? Distributed knowledge gathering

John works at KeyWest.

Jill phoned Jack.

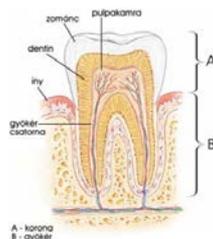
RESOURCE DESCRIPTION FRAMEWORK (RDF)

Metadata

- **Metadata:** description of data,
 - For people
 - For machines
- **Example: image metadata**
 - Generated partly automatically
 - „on this picture: John Doe, Jean-Baptiste Grenouille”
- **Example: text document metadata**
 - Author, literary category, year of publishing, etc.
- **Metadata-based search**

Syntactic Interpretation

- Can machines understand what we mean?
 - Textual / syntactic services can not
- Example: show me pictures depicting „fog”!



- Example: show me *poems by female authors*!
- Semantic solution
 - Machines should process the *meaning*, not the form
 - Use standardized concepts „fog”, „female”, „author” ...
 - Refer to it in metadata and queries

Resource Description Framework

- W3C: Resource Description Framework (**RDF**)
- Graph based structure
 - Node: **rdf:Resource** → something we talk about
 - e.g. a document, this photo, a table or “something”
 - Edge: **rdf:Property** → relation type between resources
 - e.g depicts, taken_in, its_type_is etc.
- Node and relation type name: **IRI**
(Internationalized Resource Identifier)
- Open World Assumption
- No unique names
- RDF Schema (RDFS): extension of the RDF

RDF Statements

- RDF statement = triple:
 - (subject, predicate, object)
 - subject is a Resource
 - predicate is a Property
 - object: can be a Resource or a Literal
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG“)
 - (this_photo, depicts, John Doe)
 - (this_photo, rdf:type, Photo)
 - (rdf:type, rdf:type, rdf:Property)

RDF Statements

- RDF statement = triple:
 - (subject, predicate, object)
 - Subject is an `rdf:Resource`
 - predicate is a Property
 - object can be a Resource or a Literal
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG“)
 - (this_photo, depicts, John Doe)
 - (this_photo, `rdf:type`, Photo)
 - (`rdf:type`, `rdf:type`, `rdf:Property`)

RDF Statements

- RDF statement = triple:
 - (subject, predicate, object)
 - subject is a Resource
 - Predicate is an `rdf:Property`
 - object: can be a Resource or a Literal
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG“)
 - (this_photo, depicts, John Doe)
 - (this_photo, `rdf:type`, Photo)
 - (`rdf:type`, `rdf:type`, `rdf:Property`)

RDF Statements

- RDF statement = triple:
 - (subject, predicate, object)
 - subject is a Resource
 - predicate is a Property
 - object: can be 
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG”)
 - (this_photo, depicts, John Doe)
 - (this_photo, rdf:type, Photo)
 - (rdf:type, rdf:type, rdf:Property)



Object is an rdf:Literal

RDF Statements

- RDF statement = triple:

- (subject, predicate, object)
- subject is a Resource
- predicate is a Property
- object: can be a Resource or a Literal

Resource and Property is identified by Internationalized Resource Identifiers (IRI)

- Example triples

- (this_photo, taken_in, Hungary)
- (this_photo, file_name, „DSC0001.JPG“)
- (this_photo, depicts, John Doe)
- (this_photo, rdf:type, Photo)
- (rdf:type, rdf:type, rdf:Property)

RDF Statements

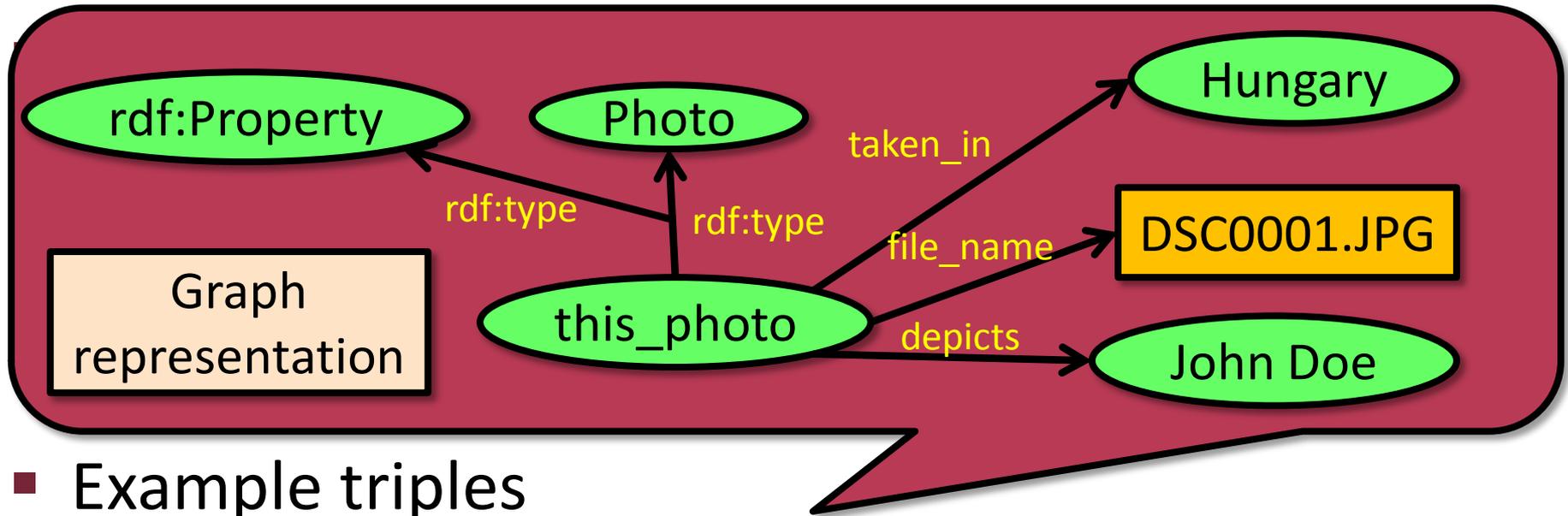
- RDF statement = triple:
 - (subject, predicate, object)
 - subject is a Resource
 - predicate is a Property
 - object: can be a Resource or a Literal
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG”)
 - (this_photo, depicts, John Doe)
 - (this_photo, **has_type**, Photo)
 - (rdf:type, rdf:type, rdf:Property)

RDF Statements

- RDF statement = triple:
 - (subject, predicate, object)
 - subject is a Resource
 - predicate is a Property
 - object: can be a Resource or a Literal
- Example triples
 - (this_photo, taken_in, Hungary)
 - (this_photo, file_name, „DSC0001.JPG“)
 - (this_photo, depicts, John Doe)
 - (this_photo, rdf:type, Photo)
 - (rdf:type, rdf:type, rdf:Property)

rdf:type is defined
in the RDF standard

RDF Statements



■ Example triples

- (this_photo, taken_in, Hungary)
- (this_photo, file_name, „DSC0001.JPG”)
- (this_photo, depicts, John Doe)
- (this_photo, rdf:type, Photo)
- (rdf:type, rdf:type, rdf:Property)

RDF Concrete Syntaxes

■ RDF+XML

```
<rdf: RDF xml ns: rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml ns: geo="http://www.geonames.org/ontology#"
  xml ns: rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xml ns: foaf="http://xmlns.com/foaf/0.1/">

<foaf: Person rdf: about="https://www.inf.mit.bme.hu/members/izso">
  <foaf: firstName>Benedek</foaf: firstName>
  <foaf: surname>Izso</foaf: surname>
  <foaf: homepage rdf: resource="https://www.inf.mit.bme.hu/members/izso"/>
  <foaf: depiction rdf: resource="https://www.inf.mit.bme.hu/.../photos/picture-578.png"/>

  <foaf: based_near>
    <geo: Feature>
      <geo: name>Hungary</geo: name>
    </geo: Feature>
  </foaf: based_near>

  <foaf: knows>
    <foaf: Person rdf: about="http://www.okkam.org/ens/idb89852b2-2b8d-43ac-ac5a-e2d00b65d0df">
      <foaf: surname>Herman</foaf: surname>
      <foaf: firstName>Ivan</foaf: firstName>
      <foaf: homepage rdf: resource="http://www.w3.org/People/Ivan"/>
    </foaf: Person>
  </foaf: knows>
</foaf: Person>

</rdf: RDF>
```

RDF Concrete Syntaxes

■ RDF+XML

```
<rdf:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:geo="http://www.geonames.org/ontology#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">

  <foaf:Person rdf:resource="https://www.inf.mit.bme.hu/members/izso">
    <foaf:firstName>Izso</foaf:firstName>
    <foaf:surname>Izso</foaf:surname>
    <foaf:homepage rdf:resource="https://www.inf.mit.bme.hu/members/izso"/>
    <foaf:depiction rdf:resource="https://www.inf.mit.bme.hu/.../photos/picture-578.png"/>

    <foaf:knows rdf:resource="https://www.inf.mit.bme.hu/members/izso">
      <geo:location>
        <geo:lat>47.516222</geo:lat>
        <geo:long>19.046944</geo:long>
      </geo:location>
    </foaf:knows>
  </foaf:Person>

  <foaf:Person rdf:resource="https://www.inf.mit.bme.hu/members/izso">
    <foaf:knows>
      <foaf:Person>
        <foaf:homepage rdf:resource="http://www.w3.org/People/Ivan/" />
      </foaf:Person>
    </foaf:knows>
  </foaf:Person>

</rdf:RDF>
```

`xmlns:foaf=http://xmlns.com/foaf/0.1/`
Organize concepts and relation types into namespaces

RDF Concrete Syntaxes

■ RDF+XML

```
<rdf:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:geo="http://www.geonames.org/ontology#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">
```

```
<foaf:Person rdf:about="https://www.okkam.org/ens/idb89852b2-2b8d-43ac-ac5a-e2d00b65d0df">
  <foaf:firstName>Benedek</foaf:firstName>
  <foaf:surname>Izso</foaf:surname>
  <foaf:homepage rdf:resource="http://www.w3.org/People/Ivan/" />
  <foaf:depiction rdf:resource="http://www.w3.org/People/Ivan/izso.jpg" />
```

(e2d00b65d0df, rdf:type, foaf:Person)
(e2d00b65d0df, foaf:firstName, "Ivan")

The semantics of these concepts and relations are well understood, these means the same for everyone.

```
<foaf:based_near>
  <geo:Feature>
    <geo:name>Hungary</geo:name>
  </geo:Feature>
</foaf:based_near>
```

```
<foaf:knows>
  <foaf:Person rdf:about="http://www.okkam.org/ens/idb89852b2-2b8d-43ac-ac5a-e2d00b65d0df">
    <foaf:surname>Herman</foaf:surname>
    <foaf:firstName>Ivan</foaf:firstName>
    <foaf:homepage rdf:resource="http://www.w3.org/People/Ivan/" />
  </foaf:Person>
</foaf:knows>
</foaf:Person>

</rdf:RDF>
```

RDF Concrete Syntaxes

■ RDF+XML

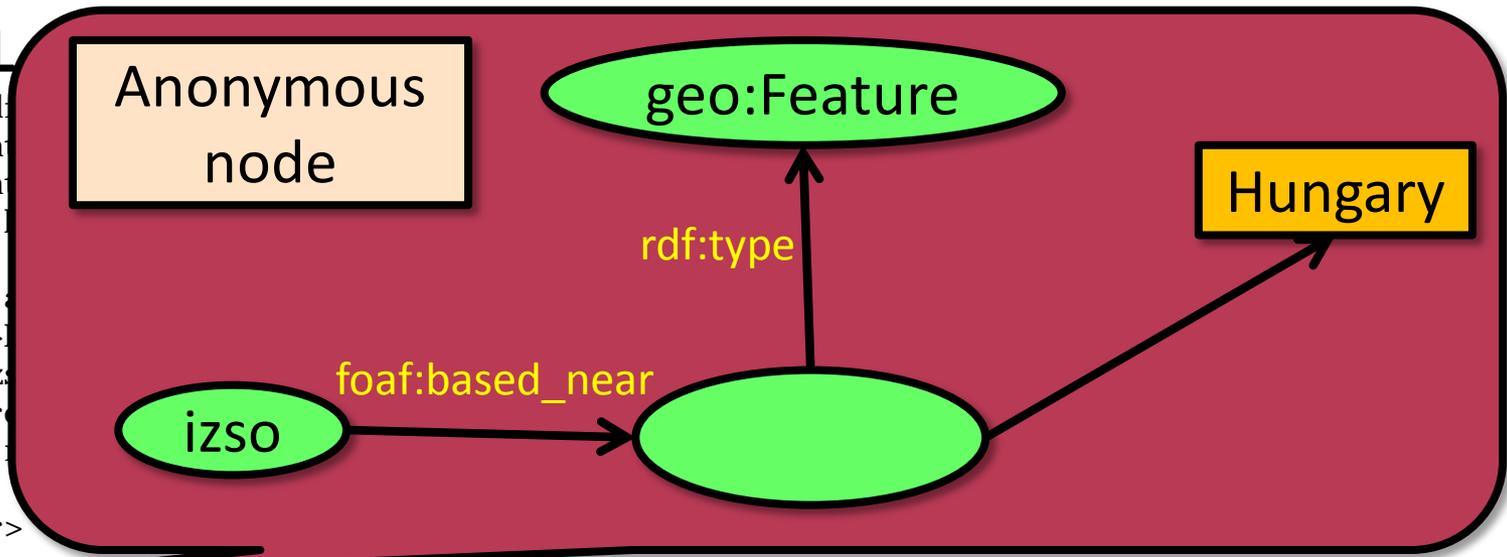
```
<rdf: RDF xml ns: rd
xml ns: geo="h
xml ns: rdf="h
xml ns: foaf="
```

```
<foaf: Person rdf:
<foaf: fi rstName>
<foaf: surname>Iz
<foaf: homepage r
<foaf: depi cti on
```

```
<foaf: based_near>
<geo: Feature>
  <geo: name>Hungary</geo: name>
</geo: Feature>
</foaf: based_near>
```

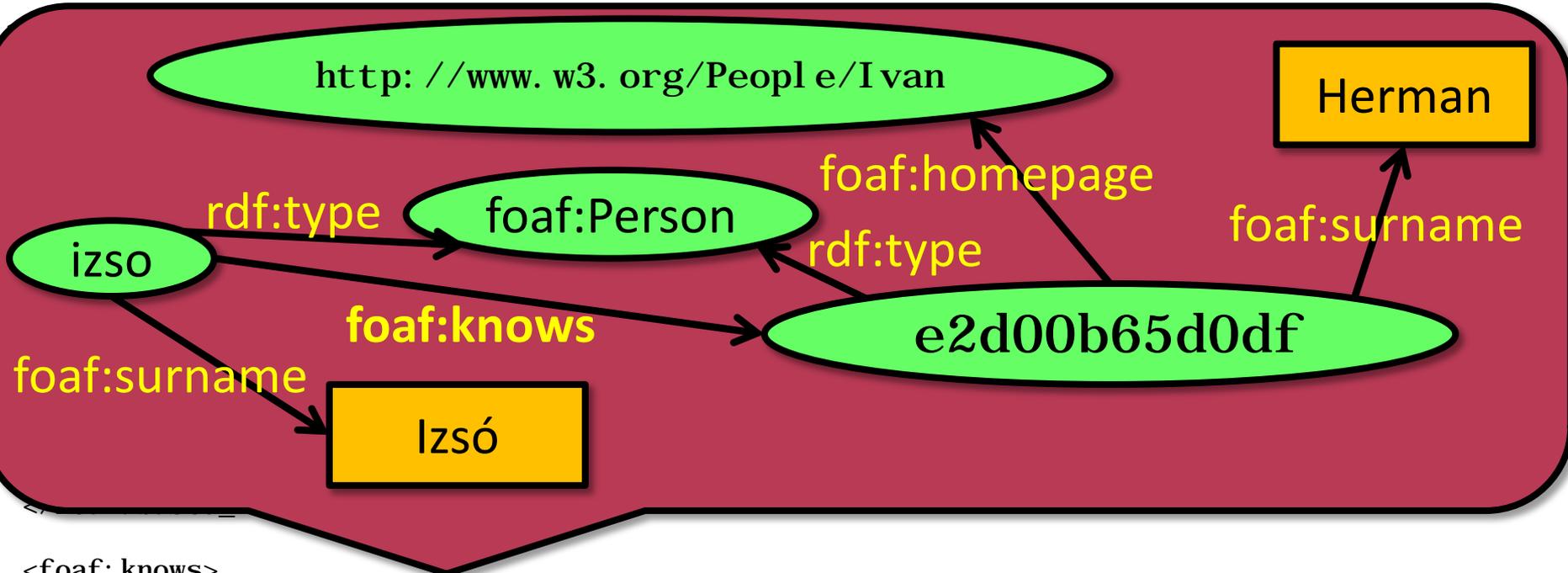
```
<foaf: knows>
<foaf: Person rdf: about="http: //www. okkam. org/ens/i db89852b2- 2b8d- 43ac- ac5a- e2d00b65d0df" >
  <foaf: surname>Herman</foaf: surname>
  <foaf: fi rstName>Ivan</foaf: fi rstName>
  <foaf: homepage rdf: resource="http: //www. w3. org/Peopl e/Ivan/" />
</foaf: Person>
</foaf: knows>
</foaf: Person>

</rdf: RDF>
```



RDF Concrete Syntaxes

■ RDF+XML



```
<foaf: knows>  
<foaf: Person rdf: about="http://www.okkam.org/ens/idb89852b2-2b8d-43ac-ac5a-e2d00b65d0df">  
  <foaf: surname>Herman</foaf: surname>  
  <foaf: firstName>Ivan</foaf: firstName>  
  <foaf: homepage rdf: resource="http://www.w3.org/People/Ivan/" />  
</foaf: Person>  
</foaf: knows>  
</foaf: Person>  
  
</rdf: RDF>
```

RDF Concrete Syntaxes

- RDF+XML

- **RDFa**: tag a HTML webpage with RDF attributes

```
<div xmlns:v="http://rdf.data-vocabulary.org/#" typeof="v:Person">
```

```
  My name is <span property="v:name">Bob Smith</span> ,
```

```
  but people call me <span property="v:nickname">Smithy</span> .
```

```
  Here is my homepage:
```

```
  <a href="http://www.example.com" rel="v:url">www.example.com</a> .
```

```
</div>
```

- **Turtle** (Terse RDF Triple Language)

```
<ftsrg:izso> rdf:type          owl:NamedIndividual ,
```

```
                        foaf:Person ;
```

```
    foaf:firstName "Benedek" ;
```

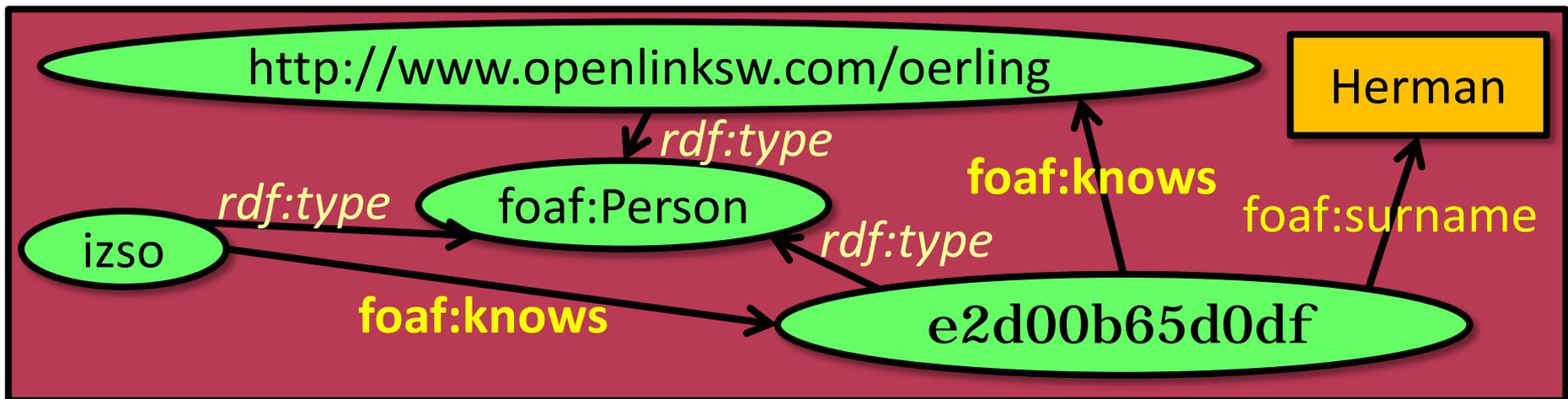
```
    foaf:homepage <https://www.inf.mit.bme.hu/members/izso> .
```

RDF Tools

- Sesame, Jena: RDF APIs for editing
- SPARQL: Graph Pattern based RDF Query Language

```
SELECT ?persA ?persB
WHERE {
  ?persA rdf:type foaf:Person .
  ?persB rdf:type foaf:Person .
  ?persA foaf:knows ?persB .
}
```

Give me pairs,
where
the first entry is a Person
as well as the second one,
and the first knows the second.



RDF Tools

- Sesame, Jena: RDF APIs for editing
- SPARQL: Graph Pattern based RDF Query Language

```
SELECT ?persA ?persB
```

```
WHERE {
```

```
  ?persA rdf:type foaf:Person .
```

```
  ?persB rdf:type foaf:Person .
```

```
  ?persA foaf:knows ?persB .
```

```
}
```

Give me pairs,

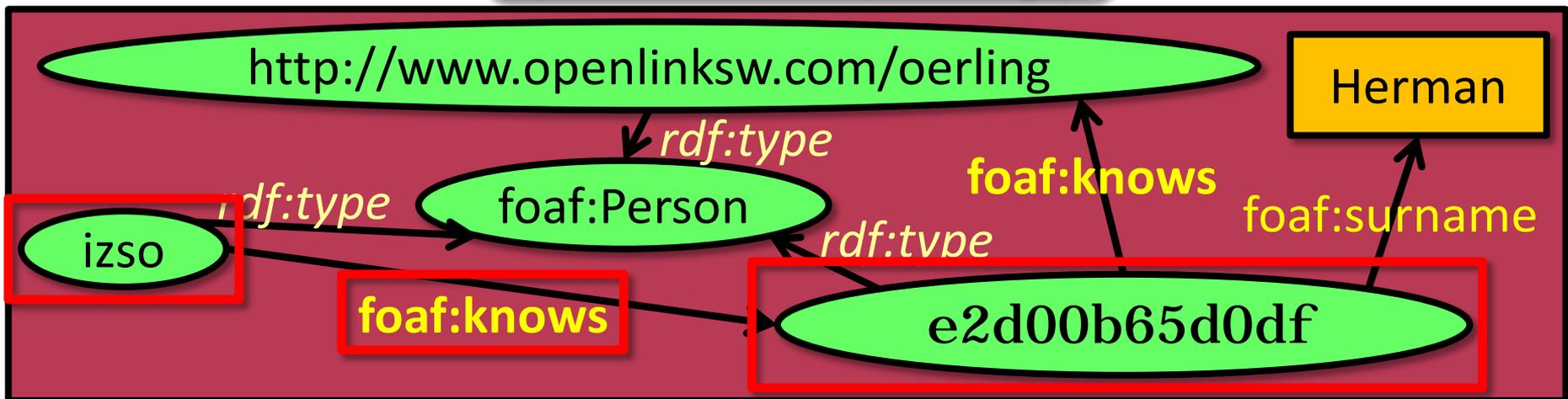
where

the first entry is a Person

as well as the second one,

and the first knows the second.

{(izso, e2d00b65d0df),
(e2d00b65d0df, oerling)}



WEB ONTOLOGY LANGUAGE (OWL 2)

Formal Background

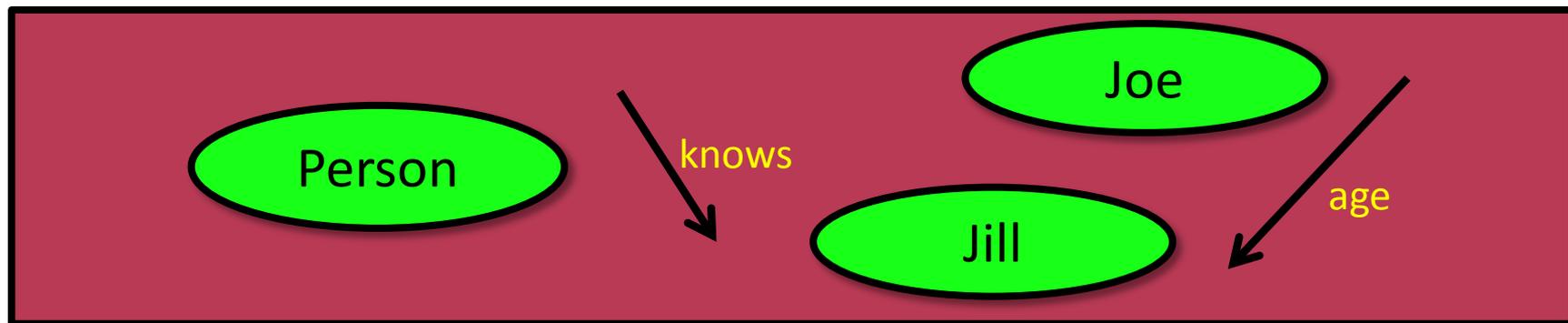
■ Axiomatic language

- Declaration (for tools): concepts, roles, individuals
- A-Box (instance model level): typify individuals, data and object role assertions
- T-Box (metamodel level): axioms between class expressions, which may include logical and bool operators

■ OWL Dialects:

- OWL 2 RDF-Based: multi-level metamodeling, not decidable

Class: **Person**
ObjectProperty: **knows**
DatatypeProperty: **age**
Individual: **Joe, Jill**



Formal Background

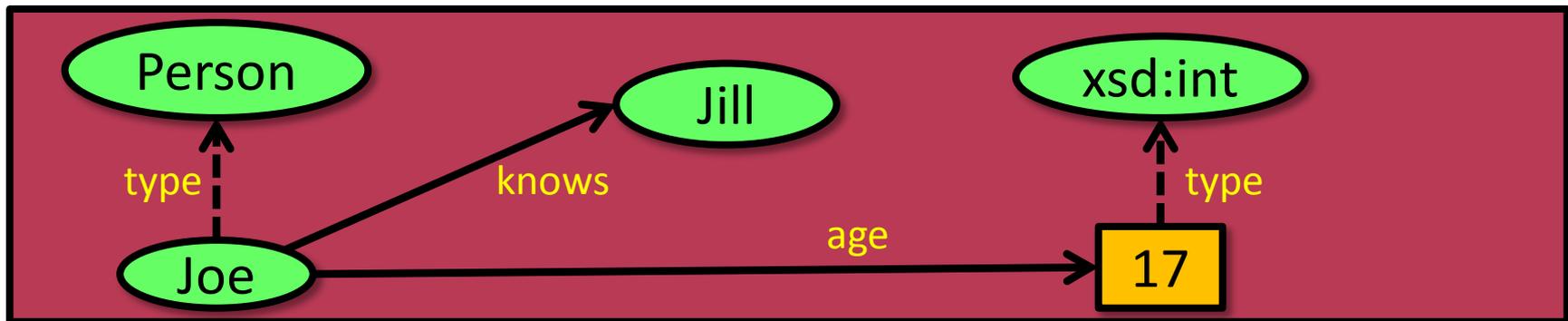
■ Axiomatic language

- Declaration (for tools): concepts, roles, individuals
- A-Box (instance model level): typify individuals, data and object role assertions
- T-Box (metamodel level): subsumption, role inclusions, expressions, which use role navigation

■ OWL Dialects:

- OWL 2 RDF-Based Semantics: support for metamodeling, not decidable

Individual: **Joe**
Types: **Person**
Facts:
knows Jill,
age "17"^^xsd:int



Formal Background

- Axiomatic language
 - Declaration (for tools): concepts, roles, individuals
 - A-Box (instance model level): typify individuals, data and object role assertions
 - T-Box (metamodel level): subsumption between class expressions, which use role navigations and bool operators
- OWL Dialects:

○ Every person is an animal:

$\text{Person} \sqsubseteq \text{Animal}$

○ The intersection of males and females is the empty set:

$\text{Male} \sqcap \text{Female} \equiv \perp$ // \perp sign means: empty set

○ Single child is a person, whose all parents have only one child:

$\text{SingleChild} \equiv \text{Person} \sqcap \forall \text{parent}.=1\text{child.Person}$

Formal Background

■ Axiomatic language

- Declaration (for tools): concepts, roles, individuals
- A-Box (instance model level): typify individuals, data and object role assertions
- T-Box (metamodel level): subsumption between class expressions, which use role expressions and constructors

■ OWL Dialects:

○ Every person is an animal:

$\text{Person} \sqsubseteq \text{Animal}$

○ The intersection of males and females is the empty set:

$\text{Male} \sqcap \text{Female} \equiv \perp$ // \perp sign means: empty set

○ Single child is a person, whose all parents have only one child:

$\text{SingleChild} \equiv \text{Person} \sqcap \forall \text{parent}.=1\text{child.Person}$

OWL defines the semantics of some new relation types, e.g. owl:intersectionOf or owl:cardinality

Formal Background

- Axiomatic language
 - Declaration (for tools): concepts, roles, individuals
 - A-Box (instance model level): typify individuals, data and object role assertions
 - T-Box (metamodel level): subsumption between class expressions, which use role navigations and bool operators
- OWL Dialects:
 - OWL 2 RDF-Based Semantics: supports multi-level metamodeling, not decidable
 - OWL 2 Direct Semantics: $\mathcal{SROIQ}(\mathcal{D})$ description logic (DL)
 - decidable **reasoning** and **consistency checking**
 - OWL 2 profiles: $(\mathcal{EL}, \mathcal{QL}, \mathcal{RL})$
 - reduced expressivity \rightarrow more efficient algorithms

Formal Background

■ Axiomatic language

○ S : Mother $\equiv \exists \text{child. Person}$ // have child who is a person

○ R : ObjectProperty knows // everybody knows oneself

Characteristics: Reflexive

○ O : Visibility $\equiv \{\text{private, public}\}$ // enumeration

○ I : child $\equiv \text{parent}^{-1}$ // Inverse

○ Q : FootballCoach $\sqsubseteq \geq 11 \text{ knows. FootballPlayer}$

// a football coach knows at least 11 football player

○ (\mathcal{D}) : Individual: Joe

Facts: age "17"^^xsd:int // data assertion

○ OWL 2 Direct Semantics: $\mathcal{SROIQ}(\mathcal{D})$ description logic (DL)

- decidable **reasoning** and **consistency checking**

○ OWL 2 profiles: $(\mathcal{EL}, \mathcal{QL}, \mathcal{RL})$

- reduced expressivity \rightarrow more efficient algorithms

Formal Background

■ Axiomatic language

- Declaration (for tools): concepts, roles, individuals
- A-Box (instance model level): typify individuals, data and object role assertions
- T-Box (metamodel level): axioms, meta-axioms, expressions, vocabularies

■ OWL Dialects:

- OWL 2 RDF-Based Semantics: lightweight, simple, good for metamodeling
- OWL 2 Direct Semantics: expressive, powerful, good for reasoning
 - decidable **reasoning** and **consistency checking**
- OWL 2 profiles: (\mathcal{EL} , \mathcal{QL} , \mathcal{RL})
 - reduced expressivity \rightarrow more efficient algorithms

OWL2 EL profile:

- useful for ontologies that contain very large number of classes and relations
- only existential quantification is enabled
- polynomial reasoning algorithms can be used w.r.t. size of the ontology

Formal Background

- Axiomatic language
 - Declaration (for tools): concepts, roles, individuals
 - A-Box (instance model level): typify individuals, data and object role assertions
 - T-Box (metamodel level): subsumption between class expressions, which use role navigations and bool operators
- OWL Dialects:
 - OWL 2 RDF-Based Semantics: supports multi-level metamodeling, not decidable
 - OWL 2 Direct Semantics: $\mathcal{SROIQ}(\mathcal{D})$ description logic (DL)
 - decidable **reasoning** and **consistency checking**
 - OWL 2 profiles: $(\mathcal{EL}, \mathcal{QL}, \mathcal{RL})$
 - reduced expressivity \rightarrow more efficient algorithms

Concrete Syntaxes

- OWL is compatible with RDF, hence RDF tools can be used (at RDF semantics level)
 - RDF/XML is obligatory for OWL 2 tools
 - For data change, not human readable

```
<owl:Class rdf:about="Young">
  <owl:equivalentClass>
    <owl:Restriction>
      <owl:onProperty rdf:resource="age"/>
      <owl:someValuesFrom>
        <rdfs:Datatype>
          <owl:onDatatype rdf:resource="&xsd:int"/>
          <owl:withRestrictions rdf:parseType="Collection">
            <rdf:Description>
              <xsd:maxExclusive rdf:datatype="&xsd:integer">18</xsd:maxExclusive>
            </rdf:Description>
          </owl:withRestrictions>
        </rdfs:Datatype>
      </owl:someValuesFrom>
    </owl:Restriction>
  </owl:equivalentClass>
</owl:Class>
```

Concrete Syntaxes

- OWL is compatible with RDF, hence RDF tools can be used (at RDF semantics level)
 - RDF/XML is obligatory for OWL 2 tools
 - For data change, not human readable
- Functional Syntax: close to AST, readable

```
EquivalentClasses(:Young  
DataSomeValuesFrom(:age DatatypeRestriction(xsd:int xsd:maxExclusive "18"^^xsd:integer)))
```

Concrete Syntaxes

- OWL is compatible with RDF, hence RDF tools can be used (at RDF semantics level)
 - RDF/XML is obligatory for OWL 2 tools
 - For data change, not human readable
- Functional Syntax: close to AST, readable
- OWL/XML: more compact than RDF/XML, but not used widely

```
<EquivalentClasses>
  <Class IRI="#Young"/>
  <DataSomeValuesFrom>
    <DataProperty IRI="#age"/>
    <DatatypeRestriction>
      <Datatype abbreviatedIRI="xsd:int"/>
      <FacetRestriction facet="&xsd;maxExclusive">
        <Literal datatypeIRI="&xsd;integer">18</Literal>
      </FacetRestriction>
    </DatatypeRestriction>
  </DataSomeValuesFrom>
</EquivalentClasses>
```

Concrete Syntaxes

- OWL is compatible with RDF, hence RDF tools can be used (at RDF semantics level)
 - RDF/XML is obligatory for OWL 2 tools
 - For data change, not human readable
- Functional Syntax: close to AST, readable
- OWL/XML: more compact than RDF/XML, but not used widely
- Manchester: readable, used in ontology editors

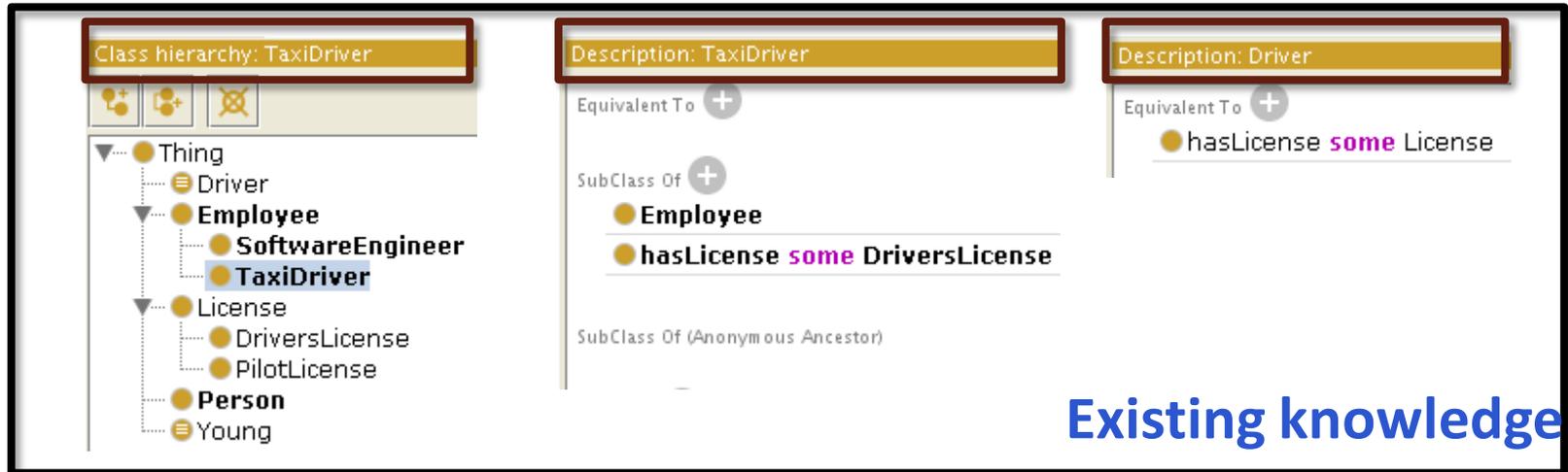
Class: Young
EquivalentTo:
age some xsd:int[< 18]

Tools

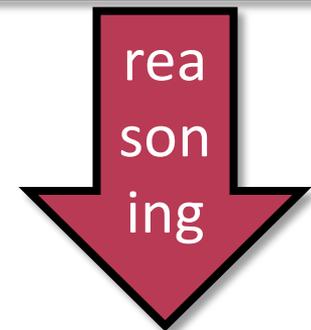
- Editors: Protégé
- Most usable part: DL subset for **reasoning** and **consistency checking**
 - Reasoning: infer new knowledge from existing ones
 - Reasoning algorithms: tableau calculi
 - Open World Assumption
 - Unique Name Assumption: not applied for OWL by default, but reasoners can be configured to use
- Reasoners: Pellet, RacerPro, Hermit
- API: OWL API (owlapi.sf.net), Java based

Ontology editing and reasoning demo

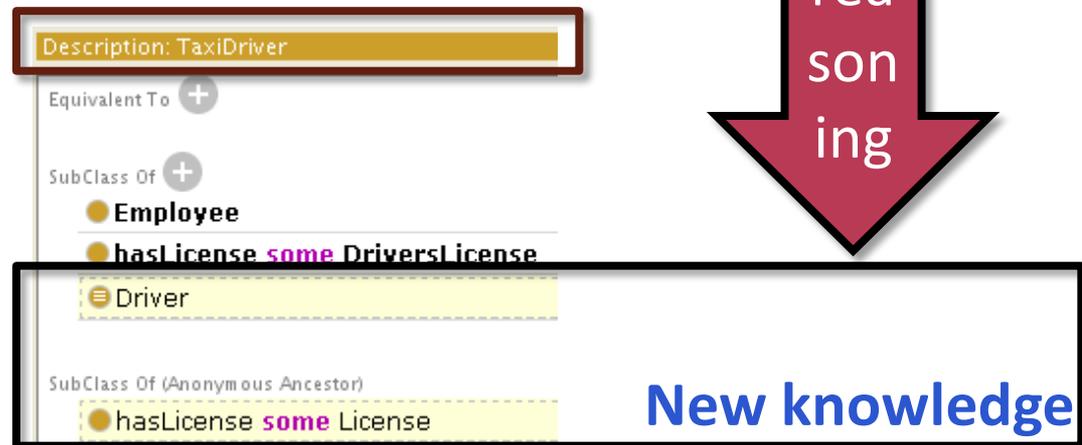
T-Box reasoning: new knowledge at meta level



Existing knowledge



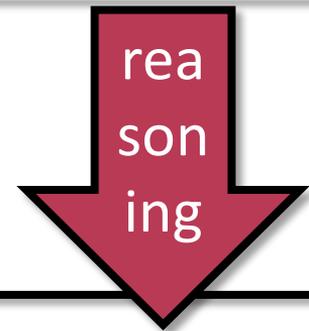
Reasoned:
Every TaxiDriver
is a Driver.



New knowledge

Ontology editing and reasoning demo

A-Box reasoning: new assertion at instance level



Reasoned:
Joe is a Driver.



Ontology editing and reasoning demo

Consistency checking: detect conflicting axioms

Description: Young

Equivalent To +

● age some int[< 18]

Description: Driver

Equivalent To +

● hasLicense some License

Disjoint With +

● Young

Property assertions: Joe

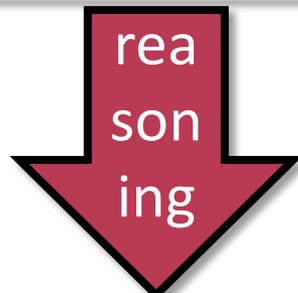
Object property assertions +

■ hasLicense JoesLicense

Data property assertions +

■ age "17"^^int

Existing knowledge



Reasoned:
Inconsistent ontology

\$ pellet explain –inconsistent profession.owl
Axiom: Thing subclassOf Nothing

Explanation(s):

- JoesLicense type DriversLicense**
Young equivalentTo age some int[< 18]

Driver equivalentTo hasLicense some License
Joe hasLicense JoesLicense
DriversLicense subclassOf License
Joe age "17"^^int
Driver disjointWith Young

SEMANTIC TECHNOLOGIES AND RESOURCES

RDF Application

■ Semantic Web

- Is a photo of my Porsche a photo of a car?
- Need standard IRIs for RDF resource/property types
- Local metadata + ontologies = semantic web

RDF Application

- RDF Site Summary (**RSS**) 
 - Items with title, description, link, creator, date, ...
 - RSS 2.0 abandons RDF, backronym

RDF Application

■ RDF Site Summary (**RSS**)



- Items with title, description, link, creator, date, ...
- RSS 2.0 abandons RDF, backronym

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://purl.org/rss/1.0/"
  xmlns:slash="http://purl.org/rss/1.0/modules/slash/"
  xmlns:dc="http://purl.org/dc/elements/1.1/" >
<title>Wikipedia Mobile Apps Switch To OpenStreetMap</title>
<link>http://rss.slashdot.org/~r/Slashdot/slashdot/~3/fQXcTk0g-aY/
  wikipedia-mobile-apps-switch-to-openstreetmap</link>
<dc:creator>timothy</dc:creator>
<dc:date>2012-04-08T14:31:00+00:00</dc:date>
<dc:subject>google</dc:subject>
<slash:department>location-aware</slash:department>
<slash:section>mobile</slash:section>
<slash:comments>125</slash:comments>
<slash:hit_parade>125,124,96,79,35,22,14</slash:hit_parade>
```

RDF Application

■ RDF Site Summary (RSS)



- Items with title, description, link, creator, date, ...
- RSS 2.0 abandons RDF, backronym

Standard
IRIs

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://purl.org/rss/1.0/"
  xmlns:slash="http://purl.org/rss/1.0/modules/slash/"
  xmlns:dc="http://purl.org/dc/elements/1.1/" >
```

```
<title>Wikipedia Mobile Apps Switch To OpenStreetMap</title>
<link>http://rss.slashdot.org/~r/Slashdot/slashdot/~3/fQXcTk0g-aY/
  wikipedia-mobile-apps-switch-to-openstreetmap</link>
```

```
<dc:creator>timothy</dc:creator>
```

```
<dc:date>2012-04-08T14:31:00+00:00</dc:date>
```

```
<dc:subject>google</dc:subject>
```

```
<slash:department>location-aware</slash:department>
```

```
<slash:section>mobile</slash:section>
```

```
<slash:comments>125</slash:comments>
```

```
<slash:hit_parade>125,124,96,79,35,22,14</slash:hit_parade>
```

Defined
vocabulary
used to
describe data

Semantic Web Vocabularies

- Dublin Core (DC) ontology
 - Librarian metadata for documents
 - Title, publisher, language, format, date, creator, etc.
 - Widespread usage (e.g. as an RSS Module)
- Friend-of-a-Friend (FOAF) ontology
 - Classes: Person, Image, Document, OnlineAccount, etc.
 - Attributes: surname, birthday, title, etc.
 - Relationships: knows, made, depicts, weblog, topic, logo, openID, page, interest, etc.
 - Used / usable in Facebook, flickr, LauchPad...

Semantic Web Vocabularies

■ Opengraph Protocol (Facebook)



A Like button on a movie page on imdb.com

* More info: [The Open Graph Protocol: Understanding the design decisions](#)

Semantic Web Vocabularies

■ Opengraph Protocol (Facebook)



A Like button on a movie page on imdb.com

Me:

- I like The Rocks

* More info: [The Open Graph Protocol: Understanding the design decisions](#)

Semantic Web Vocabularies

■ Opengraph Protocol (Facebook)



A Like button on a movie page on imdb.com

Me:

- I like The Rocks

Computer:

- What?? This HTML tag, or CSS?

* More info: [The Open Graph Protocol: Understanding the design decisions](#)

Semantic Web Vocabularies

■ Opengraph Protocol (Facebook)



A Like button on a movie page on imdb.com

Me:

- I like The Rocks

Computer:

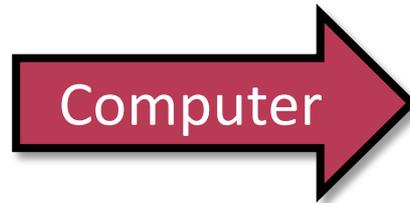
- What?? This HTML tag, or CSS?

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:og="http://opengraphprotocol.org/schema/" >
<head>
<meta property="og:title" content="The Rock (1996)"/>
<meta property="og:type" content="video.movie"/>
<meta property="og:image" content="http://media-imdb.com/images/the_rocks.jpg"/>
<meta property="og:site_name" content="IMDb"/>
<meta property="fb:app_id" content="115109575169727"/>
</head>
```

* More info: The Open Graph Protocol: Understanding the design decisions

Semantic Web Vocabularies

■ Opengraph Protocol (Facebook)



```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:og="http://opengraphprotocol.org/schema/" >
<head>
<meta property="og:title" content="The Rock (1996)"/>
<meta property="og:type" content="video.movie"/>
<meta property="og:image" content="http://media-imdb.com/images/the_rocks.jpg"/>
<meta property="og:site_name" content="IMDb"/>
<meta property="fb:app_id" content="115109575169727"/>
</head>
```

* More info: The Open Graph Protocol: Understanding the design decisions

Semantic Web Vocabularies

■ Microformat

Tuesday 17 April 2012



Epica

Epica, Xandria, Stream of Passion

Alcatraz

Milano, Milan,
Italy

Thursday 19 April 2012



Epica European Tour

Epica, Xandria, Stream of Passion

Razzmatazz

Barcelona,
Spain

Semantic Web Vocabularies

■ Microformat

Tuesday 17 April 2012



Epica

Epica, Xandria, Stream of Passion

Alcatraz

Milano, Milan,
Italy

Thursday 19 April 2012



Epica European Tour

Epica, Xandria, Stream of Passion

Razzmatazz

Barcelona,
Spain

Last.fm:

We collected the events for the band Epica. How can we make data more searchable?

Semantic Web Vocabularies

■ Microformat

Tuesday 17 April 2012



Epica

Epica, Xandria, Stream of Passion

Alcatraz

Milano, Milan,
Italy

```
<tr class="day" id="day-17-04-2012">
<td class="location">
  <span class="adr">
    <a href="/venue/8778527+Alcatraz" class="strong">Alcatraz</a>
    <br />Milano, <span class="locality">Milan</span>,
    <span class="country-name">Italy</span>
  </span>
  <div class="geo">
    <span class="latitude">45.494282</span>
    <span class="longitude">9.182612</span>
  </div>
</td>
```

Last.fm:

We collected the events for the band Epica. How can we make data more searchable?

Semantic Web Vocabularies

■ Microformat

Tuesday 17 April 2012



Epica

Epica, Xandria, Stream of Passion

Alcatraz

Milano, Milan,
Italy



[Epica's Concert Listing – Free listening, videos, concerts, stats ...](#)

www.last.fm/music/Epica/+events - Tárolt változat - Oldal lefordítása

Discover more music, **concerts**, videos, and pictures with the largest ...

ápr. 17., K [Epica](#) - Milan, Italy

máj. 11., P [Epica](#) - Munich, Germany

máj. 13., V [Epica](#) - Budapest, Hungary

```
<tr class="day" id="day-17-04-2012">
```

```
<td class="location">
```

```
<span class="adr">
```

```
<a href="/venue/8778527+Alcatraz" class="strong">Alcatraz</a>
```

```
<br />Milano, <span class="locality">Milan</span>,</td>
```

```
<span class="country-name">Italy</span>
```

```
</span>
```

```
<div class="geo">
```

```
<span class="latitude">45.494282</span>
```

```
<span class="longitude">9.182612</span>
```

```
</div>
```

```
</td>
```

Semantic Web Vocabularies

- **Microdata: schema.org (Google)**

Semantic Web Vocabularies

- **Microdata: schema.org (Google)**

My homepage

[Benedek Izsó](#)

EMF-IncQuery is a great tool. Rating based on my opinion:
5 stars - based on 1 opinion

Semantic Web Vocabularies

■ Microdata: schema.org (Google)

[Benedek Izsó](#)

EMF-IncQuery is a great tool. Rating based on my opinion:
5 stars - based on 1 opinion

```
<body>
<div itemscope itemtype="http://schema.org/People">
  <span itemprop="name">
    <a href="https://plus.google.com/111902568077770766904?rel=author">
      Benedek Izsó</a>
    </span>
  </div>
<div itemscope itemtype="http://schema.org/Product">
  <span itemprop="name">EMF-IncQuery</span> is a great tool. Rating based on my opinion:
  <div itemprop="aggregateRating" itemscope itemtype="http://schema.org/AggregateRating">
    <span itemprop="ratingValue">5</span> stars -
    based on <span itemprop="reviewCount">1</span> review
  </div>
</div>
</body>
```

Semantic Web Vocabularies

■ Microdata: schema.org (Google)

[Benedek Izsó](#)

EMF-IncQuery is a great tool. Rating based on my opinion:
5 stars - based on 1 opinion

```
<body>
```

```
<div itemscope itemtype="http://schema.org/People">
```

```
<span itemprop="name">
```

```
<a href="https://plus.google.com/111902568077770766904?rel=author">
```

```
Benedek Izsó</a>
```

```
</span>
```

```
</div>
```

```
<div itemscope itemtype="http://schema.org/Product">
```

```
<span itemprop="name">EMF-IncQuery</span> is a great tool. Rating based on my opinion:
```

```
<div itemprop="aggregateRating" itemscope itemtype="http://schema.org/AggregateRating">
```

```
<span itemprop="ratingValue">5</span> stars -
```

```
based on <span itemprop="reviewCount">1</span> review
```

```
</div>
```

```
</div>
```

```
</body>
```

Only a product
can be rated

Semantic Web Vocabularies

■ Microdata: schema.org (Google)

[Benedek Izsó](#)

EMF-IncQuery is a great tool. Rating based on my opinion:
5 stars - based on 1 opinion

```
<body>
<div itemscope itemtype="http://schema.org/People">
  <span itemprop="name">
    <a href="https://plus.google.com/111902568077770766904?rel=author">
      Benedek Izsó</a>
    </span>
  </div>
<div itemscope itemtype="http://schema.org/Product">
  <span itemprop="name">EMF-IncQuery</span> is a great tool. Rating based on my opinion:
  <div itemprop="aggregateRating" itemscope itemtype="http://schema.org/AggregateRating">
    <span itemprop="ratingValue">5</span> stars -
    based on <span itemprop="reviewCount">1</span> review
  </div>
</div>
</body>
```

Google search preview

[Benedek Izsó's sendemo page](#)
mit.bme.hu/~i/izso/sendemo.html - Cached
★★★★★ 1 review

The excerpt from the page will show up here. The reason we can't show text from your webpage is because the text depends on the query the user types.

Google may show
rich results



Benedek Izsó

Semantic Web Vocabularies

- Common vocabularies: schema.org (G), Open Graph Protocol (FB), Microformats
- Data searching and connecting services **needs data** and not the graphical look
 - A website will not upload its database
 - But may annotate their website content
- One easily usable, unified vocabulary
 - not reused existing ones which can be a big mess
 - Concepts can be the most popular ones, based on site statistics
- Data is more structured with annotations than natural language, but less structured than a DB.
 - E.g.: not every Product must have a rating in semweb
 - But in an object oriented datamodel every Product have a rating reference

Knowledge Bases

- WordNet: lexical knowledge base of english words
 - Synonym sets (synsets)
 - Semantic relations, including
 - Antonym (opposite)
 - Hypernym, hyponym (super/subtype)
- DBpedia Knowledge Base
 - RDF information
 - Automatically extracted from Wikipedia
 - Manual annotation, links to WordNet etc.
 - SPARQL endpoint

Knowledge Bases

- DBpedia SPARQL Query example with Sesame



Knowledge Bases

- DBpedia SPARQL Query example with Sesame

What is the most popular
symphonic metal record label?



Knowledge Bases

- DBpedia SPARQL Query example with Sesame

What is the most popular symphonic metal record label?



Artists have genre and record label on wikipedia, how can I get and count it programmatically?

Knowledge Bases

- DBpedia SPARQL Query example with Sesame

```
SELECT ?label (count(?label) AS ?labelCount)
WHERE {
  ?person dbp:genre dbr:Symphonic_metal .
  ?person dbp:label ?label .
  ?label rdf:type dbo:RecordLabel .
}
```



Knowledge Bases

- DBpedia SPARQL Query example with Sesame

```
SELECT ?label (count(?label) AS ?labelCount)
WHERE {
  ?person dbp:genre dbr:Symphonic_metal .
  ?person dbp:label ?label .
  ?label rdf:type dbo:RecordLabel .
}
```



Record Label	Count
Napalm_Records	68
Limb_Music	46
Roadrunner_Records	50
SPV_GmbH	44

Knowledge Bases

■ DBpedia SPARQL Query example with Sesame

```
HTTPRepository sparqlEndpoint = new HTTPRepository("http://live.dbpedia.org/sparql", "");
sparqlEndpoint.initialize(); conn = sparqlEndpoint.getConnection();
String sparqlQuery = " SELECT ?label (count(?label) AS ?labelCount) WHERE { " +
    " ?person dbp:genre dbr:Symphonic_metal . " +
    " ?person dbp:label ?label . " +
    " ?label rdf:type dbo:RecordLabel . " +
    " }";

TupleQuery query = conn.prepareTupleQuery(QueryLanguage.SPARQL, sparqlQuery);
TupleQueryResult result = query.evaluate();
while (result.hasNext()) { // print answers
    BindingSet triple = result.next();
    System.out.println(triple.getBinding("label").getValue());
    Literal labelCount = (Literal) triple.getBinding("labelCount").getValue();
    System.out.println(labelCount.intValue());
} conn.close();
```

Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)

Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)



Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)

What cafes are near
to Calvin Square?



Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)

```
Prefix lgdo: <http://linkedgeodata.org/ontology/>
Select * From <http://linkedgeodata.org>
{
  ?s a          lgdo:Cafe .
  ?s rdfs:label ?l .
  ?s geo:geometry ?g .
  Filter(bif:st_intersects (?g,
    bif:st_point (19.061667, 47.489722), 0.4)) .
};
```



Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)

```
Prefix lgdo: <http://linkedgeodata.org/ontology/>  
Select * From <http://linkedgeodata.org>
```

```
{  
  ?s a          lgdo:Cafe .  
  ?s rdfs:label ?l .  
  ?s geo:geometry ?g .  
  Filter(bif:st_intersects (?g,  
    bif:st_point (19.061667, 47.489722), 0.4)) .  
};
```

Filter near places:
Intersect function not in
SPARQL, LGD extension

Coordinate of Calvin Square

0.4 km



Knowledge Bases

- Linked GeoData: OpenStreetMap in RDF format
 - SPARQL endpoint
 - or downloadable RDF (10GB)

```
Prefix lgdo: <http://linkedgeodata.org/ontology/>
Select * From <http://linkedgeodata.org>
{
  ?s a          lgdo:Cafe .
  ?s rdfs:label ?l .
  ?s geo:geometry ?g .
  Filter(bif:st_intersects (?g,
    bif:st_point (19.061667, 47.489722), 0.4)) .
};
```



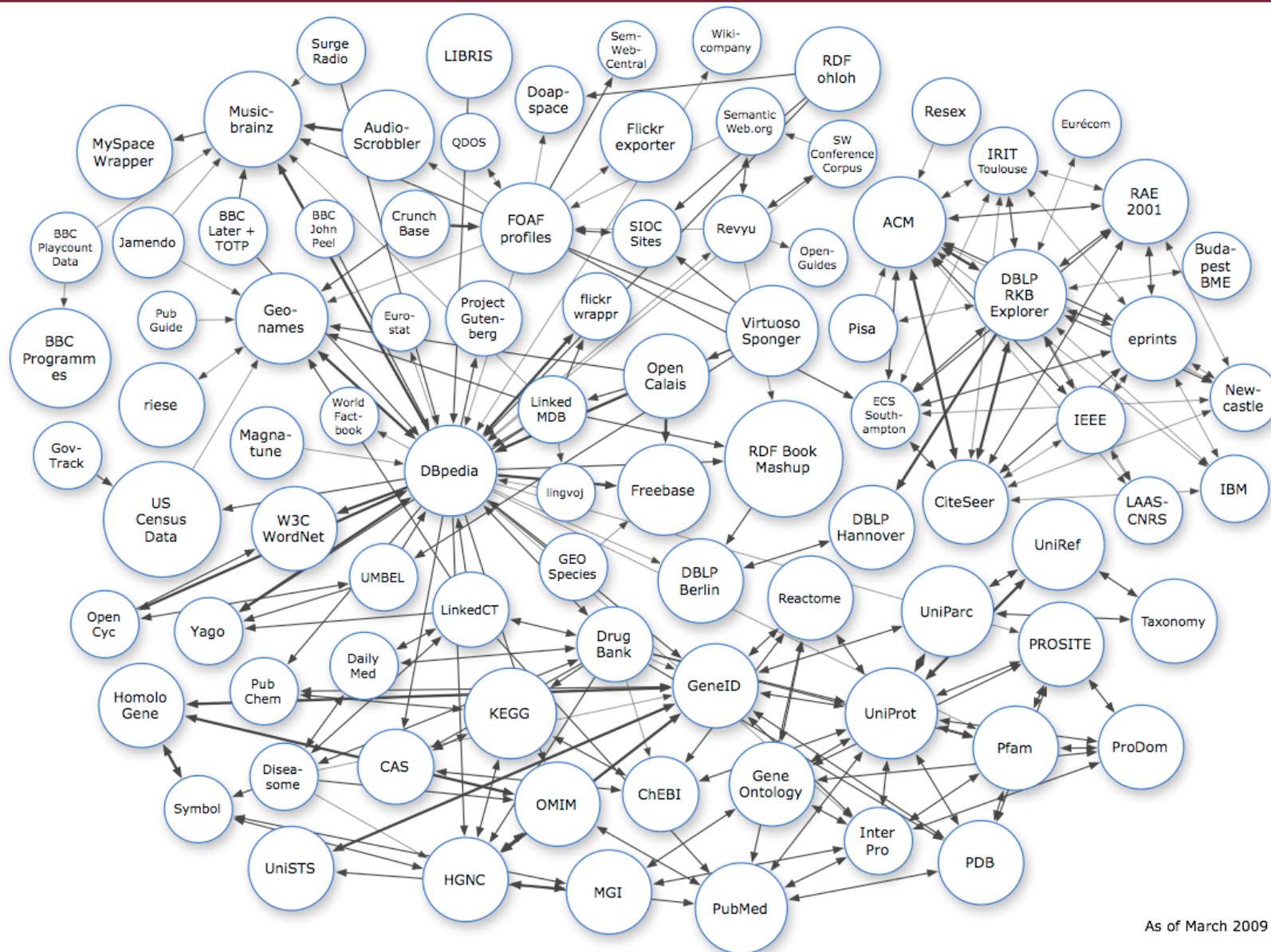
Cafes near Calvin Square (?l)

Frank Zappa

LUMEN

California Coffee Bean

Linking Open Data (LOD)



As of March 2009

MODELING APPROACHES

DSM and Ontology languages - Similarities

	Domain Specific Models	Ontology (DL)
Language	UML, EMF	RDF, OWL
1-ary predicate	EClass	Concept
2-ary predicate	EReference/EAttribute	Object/Datatype property
Model element	EObject	Instance
Enumeration	EEnum	Nominals
Cardinality	Role cardinality	Cardinality restriction
Inverse	EOpposite	owl:inverseOf
Type of	EType	rdf:type
Subsumption	Generalization	rdfs:subClassOf

DSM and Ontology languages - Differences

	Domain Specific Models	Ontology (DL)
Classifying instances	Enum	Concept
Constraint description	Other auxiliary language (e.g.: OCL)	Built into the language
Types	One instance have 1 type	One instance can belong to multiple classes
Unique Name Assumption	By default: No	Yes
Open World Assumption	Yes	No

DSM and Ontology based modeling - Usage

	Domain Specific Models	Ontology (DL)
Metamodel description	Practical description for engineers	Precise incomplete definitions for mathematicians
Instance model	Complete data	Continuously evolving data
Strengths	<ul style="list-style-type: none"> • Code generation • Model transformation • Existing tools • Fast instance querying 	<ul style="list-style-type: none"> • Reasoning • Consistency checking • Common vocabularies • Prototyping (OWA, UNA)
Modeling	Efficient instance model editing and querying	Efficient metamodel designing and reasoning
Language elements	Common OO terms	Mathematical constraints (e.g.: existential and universal quantification)
Structuredness	Formal, highly structured	Formal, but not structured

DSM and Ontology based modeling - Usage

	Domain Specific Models	Ontology (DL)
Metamodel description	Practical description for engineers	Precise incomplete definitions for mathematicians
Instance model	Complete data	Continuously evolving data
Strengths	<ul style="list-style-type: none">• Code generation• Model transformation• Existing tools• Fast instance querying	<ul style="list-style-type: none">• Reasoning• Consistency checking• Common vocabularies• Prototyping (OWA, UNA)
Modeling	Efficient instance model editing and querying	Efficient metamodel designing and reasoning
Language elements	Common OO terms	Mathematical constraints (e.g.: existential and universal quantification)
Structuredness	Formal, highly structured	Formal, but not structured

If two instances belong to the same class, have they the same references too?

SUMMARY

Summary

- Semantic technologies
 - Metadata (RDF)
 - Ontologies (OWL)
 - Formal logic based reasoning
 - Querying with SPARQL
- Applications
 - Domain ontologies in expert systems
 - Semantic Web
 - Ontology-based semantic data integration

Recommended reading

Benkő-Szeredi-Lukácsy:
*A szemantikus világháló
elmélete és gyakorlata.*
Typotex, 2005.

BMEVIMIM222
Információ- és
tudásintegrálás

(MSc intelligens rendszerek szakirány)

