Semantic Web

Part 3
The ontology layer 1: Ontologies, Description Logics, and OWL

Riccardo Rosati

Corso di Laurea Magistrale in Ingegneria Informatica
Sapienza Università di Roma

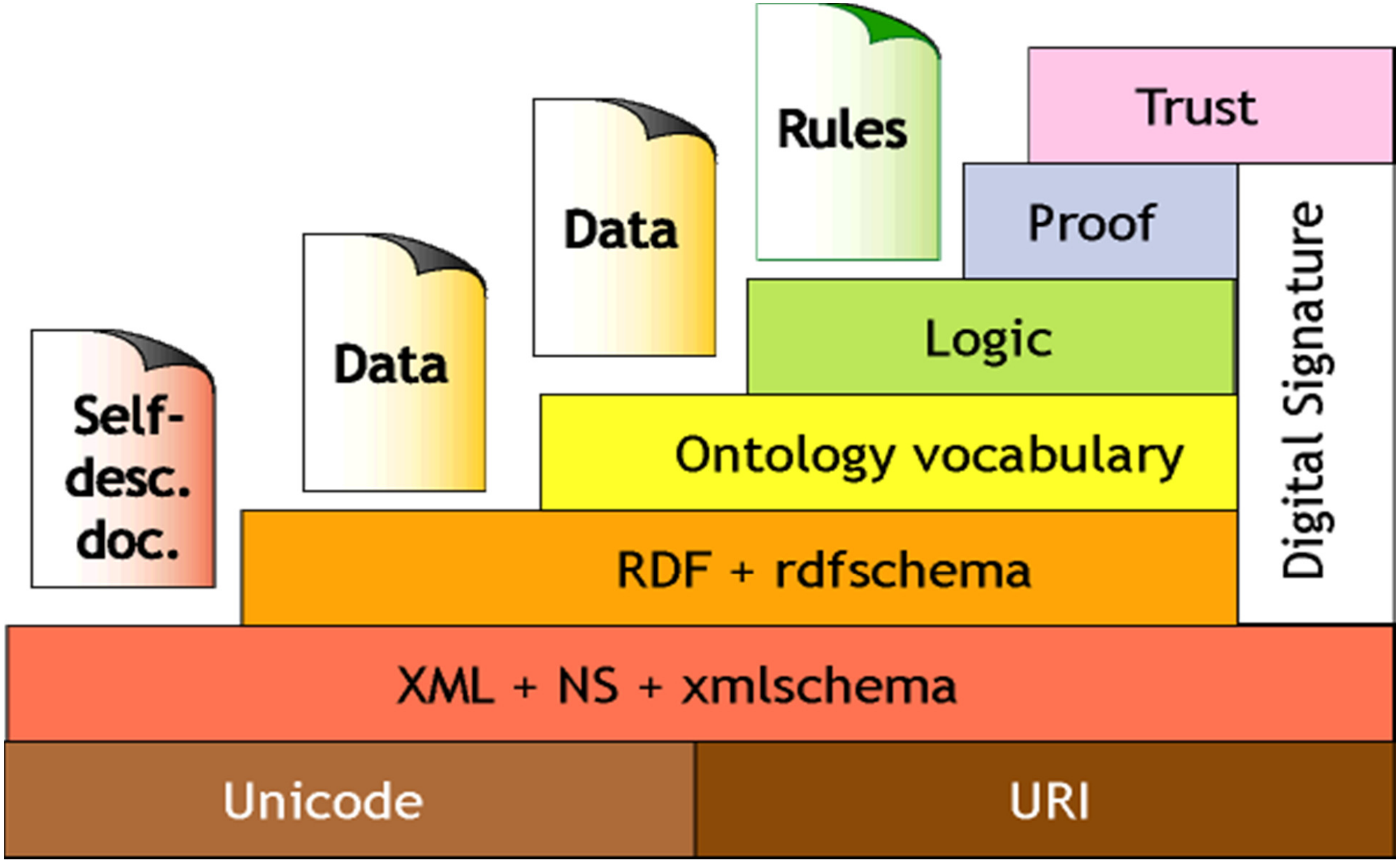
2012/2013

REMARK

Most of the material of this lecture is taken from the ISWC 2003 “Tutorial on OWL” by Sean Bechhofer, Ian Horrocks, and Peter Patel-Schneider

(http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial/)

The Semantic Web Tower



Ontology: origins and history

a philosophical discipline—a branch of philosophy that deals with the nature and the organisation of reality

- Science of Being (Aristotle, Metaphysics, IV, 1)
- Tries to answer the questions:

  What characterizes being?

  Eventually, what is being?

Ontology in computer science

- An ontology is an engineering artifact:
  - It is constituted by a specific vocabulary used to describe a certain reality, plus
  - a set of explicit assumptions regarding the intended meaning of the vocabulary.

- Thus, an ontology describes a formal specification of a certain domain:
  - Shared understanding of a domain of interest
  - Formal and machine manipulable model of a domain of interest

"An explicit specification of a conceptualisation" [Gruber93]

Ontology in computer science

- ontology = shared conceptualization of a domain of interest
- shared vocabulary

  ⇒ simple (shallow) ontology
- (complex) relationships between “terms”

  ⇒ deep ontology
- AI view:
  - ontology = logical theory (knowledge base)
- DB view:
  - ontology = conceptual data model

Structure of an ontology

Ontologies typically have two distinct components:

- Names for important concepts in the domain
  - Elephant is a concept whose members are a kind of animal
  - Herbivore is a concept whose members are exactly those animals who eat only plants or parts of plants
  - Adult_Elephant is a concept whose members are exactly those elephants whose age is greater than 20 years
- Background knowledge/constraints on the domain
  - Adult_Elephants weigh at least 2,000 kg
  - All Elephants are either African_Elephants or Indian_Elephants
  - No individual can be both a Herbivore and a Carnivore

Ontology languages

- Wide variety of languages for “Explicit Specification”
  - Graphical notations
  - Logic based
  - Probabilistic/fuzzy
  - ...
- Degree of formality varies widely
  - Increased formality makes languages more amenable to machine processing (e.g., automated reasoning)

Ontology languages

- Graphical notations:
    - Semantic networks
    - Topic Maps
    - UML
    - RDF

Ontology languages

- Logic based languages:
  - Description Logics (e.g., OIL, DAML+OIL, OWL)
  - Rules (e.g., RuleML, Logic Programming/Prolog)
  - First Order Logic (e.g., KIF)
  - Conceptual graphs
  - (Syntactically) higher order logics (e.g., LBase)
  - Non-classical logics (e.g., F-logic, Non-Monotonic Logics, Modal Logics)

Obect-oriented languages

many languages use object-oriented models based on:

- Objects/Instances/Individuals
  - Elements of the domain of discourse
  - Equivalent to constants in FOL
- Types/Classes/Concepts
  - Sets of objects sharing certain characteristics
  - Equivalent to unary predicates in FOL
- Relations/Properties/Roles
  - Sets of pairs (tuples) of objects
  - Equivalent to binary predicates in FOL

Web schema languages

- Existing Web languages extended to facilitate content description
  - XML → XML Schema (XMLS)
  - RDF → RDF Schema (RDFS)
- XMLS *not* an ontology language
  - Changes format of DTDs (document schemas) to be XML
  - Adds an extensible type hierarchy
    - Integers, Strings, etc.
    - Can define sub-types, e.g., positive integers
- RDFS *is* recognizable as an ontology language
  - Classes and properties
  - Sub/super-classes (and properties)
  - Range and domain (of properties)

Limitations of RDFS

- RDFS too weak to describe resources in sufficient detail
  - No localised range and domain constraints
    - Can’t say that the range of hasChild is person when applied to persons and elephant when applied to elephants
  - No existence/cardinality constraints
    - Can’t say that all *instances* of person have a mother that is also a person, or that persons have exactly 2 parents
  - No transitive, inverse or symmetrical properties
    - Can’t say that isPartOf is a transitive property, that hasPart is the inverse of isPartOf or that touches is symmetrical
  - …
- Difficult to provide reasoning support
  - No “native” reasoners for non-standard semantics
  - May be possible to reason via FO axiomatisation

Web ontology language requirements

Desirable features identified for Web Ontology Language:

- Extends existing Web standards
  - Such as XML, RDF, RDFS
- Easy to understand and use
  - Should be based on familiar KR idioms
- Formally specified
- Of “adequate” expressive power
- Possible to provide automated reasoning support

From RDF to OWL

- Two languages developed to satisfy above requirements
  - OIL: developed by group of (largely) European researchers (several from EU OntoKnowledge project)
  - DAML-ONT: developed by group of (largely) US researchers (in DARPA DAML programme)
- Efforts merged to produce DAML+OIL
  - Development was carried out by "Joint EU/US Committee on Agent Markup Languages"
  - Extends ("DL subset" of) RDF
- DAML+OIL submitted to W3C as basis for standardisation
  - Web-Ontology (WebOnt) Working Group formed
  - WebOnt group developed OWL language based on DAML+OIL
- OWL language became a W3C Recommendation in 2004

OWL

- OWL = Web Ontology Language
- the OWL family is constituted by 3 different languages (with different expressive power):
  - OWL Full
  - OWL-DL
  - OWL-Lite
- technology at an "early" stage
  - standardized in 2004
  - reasoning techniques and tools are very recent
  - "optimization" of reasoning unexplored
  - 2009: W3C standardization of OWL 2

OWL language

- Three species of OWL
  - OWL full is union of OWL syntax and RDF
  - OWL DL restricted to FOL fragment (DAML+OIL)
  - OWL Lite is “easier to implement” subset of OWL DL
- Semantic layering
  - OWL DL = OWL full within DL fragment
  - DL semantics officially definitive
- OWL DL based on SHIQ Description Logic
  - In fact it is equivalent to SHOIN(D_n) DL
- OWL DL Benefits from many years of DL research
  - Well defined semantics
  - Formal properties well understood (complexity, decidability)
  - Known reasoning algorithms
  - Implemented systems (highly optimised)

OWL class constructors

Constructor	DL Syntax	Example	Modal Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human $\sqcap$ Male	$C_1 \wedge \ldots \wedge C_n$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer	$C_1 \vee \ldots \vee C_n$
complementOf	$\neg C$	¬Male	$\neg C$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}	$x_1 \vee \ldots \vee x_n$
allValuesFrom	$\forall P.C$	∀hasChild.Doctor	$[P]C$
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer	$\langle P \rangle C$
maxCardinality	$\leqslant n P$	⩽1hasChild	$[P]_{n+1}$
minCardinality	$\geqslant n P$	⩾2hasChild	$\langle P \rangle_n$

- XMLS datatypes as well as classes in ∀P.C and ∃P.C
  - E.g., ∃hasAge.nonNegativeInteger
- Arbitrarily complex nesting of constructors
  - E.g., Person $\sqcap$ ∀hasChild.Doctor $\sqcup$∃hasChild.Doctor

RDFS syntax

E.g., concept $Person \sqcap \forall hasChild.Doctor \sqcup \exists hasChild.Doctor$:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

OWL axioms

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg${peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+$ $\sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1 P$	$\top \sqsubseteq \leqslant$1hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1 P^-$	$\top \sqsubseteq \leqslant$1hasSSN$^-$

Axioms (mostly) reducible to inclusion ($\sqsubseteq$)

$\mathbf{C} \equiv \mathbf{D}$ iff both $\mathbf{C} \sqsubseteq \mathbf{D}$ and $\mathbf{D} \sqsubseteq \mathbf{C}$

XML Schema datatypes in OWL

- OWL supports XML Schema primitive datatypes
  - E.g., integer, real, string, …
- Strict separation between "object" classes and datatypes
  - Disjoint interpretation domain Δ_D for datatypes
    - For a datavalue d, $d^{\mathcal{I}} \subseteq \Delta_D$
    - And $\Delta_D \cap \Delta^{\mathcal{I}} = \emptyset$
  - Disjoint "object" and datatype properties
    - For a datatype propterty P, $P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_D$
    - For object property S and datatype property P, $S^{\mathcal{I}} \cap P^{\mathcal{I}} = \emptyset$
- Equivalent to the "(D_n)" in $\mathcal{SHOIN}(D_n)$

Why separate classes and datatypes?

- Philosophical reasons:
  - Datatypes structured by built-in predicates
  - Not appropriate to form new datatypes using ontology language
- Practical reasons:
  - Ontology language remains simple and compact
  - Semantic integrity of ontology language not compromised
  - Implementability not compromised — can use hybrid reasoner

OWL DL semantics

- Mapping OWL to equivalent DL ($\mathcal{SHOIN}(D_n)$):
  - Facilitates provision of reasoning services (using DL systems)
  - Provides well defined semantics
- DL semantics defined by interpretations: $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
  - $\Delta^{\mathcal{I}}$ is the domain (a non-empty set)
  - $\cdot^{\mathcal{I}}$ is an interpretation function that maps:
    - Concept (class) name A $\rightarrow$ subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$
    - Role (property) name R $\rightarrow$ binary relation $R^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$
    - Individual name i $\rightarrow$ $i^{\mathcal{I}}$ element of $\Delta^{\mathcal{I}}$

DL semantics

- Interpretation function $\cdot^{\mathcal{I}}$ extends to concept expressions in an obvious way, i.e.:

$$
\begin{aligned}
&(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
&(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
&(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
&\{x\}^{\mathcal{I}} = \{x^{\mathcal{I}}\} \\
&(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y. \langle x, y \rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
&(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y. (x, y) \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\} \\
&(\leqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y \rangle \in R^{\mathcal{I}}\} \leqslant n\} \\
&(\geqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y \rangle \in R^{\mathcal{I}}\} \geqslant n\}
\end{aligned}
$$

DL knowledge bases (ontologies)

- An OWL ontology maps to a DL Knowledge Base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$
  - $\mathcal{T}$ (Tbox) is a set of axioms of the form:
    - $C \sqsubseteq D$ (concept inclusion)
    - $C \equiv D$ (concept equivalence)
    - $R \sqsubseteq S$ (role inclusion)
    - $R \equiv S$ (role equivalence)
    - $R^+ \sqsubseteq R$ (role transitivity)
  - $\mathcal{A}$ (Abox) is a set of axioms of the form
    - $x \in D$ (concept instantiation)
    - $\langle x,y \rangle \in R$ (role instantiation)

DL knowledge bases (ontologies)

- Two sorts of TBox axioms often distinguished:
  - "Definitions"
    - $C \sqsubseteq D$ or $C \equiv D$ where C is a concept name
  - General Concept Inclusion axioms (GCIs)
    - $C \sqsubseteq D$ where C,D arbitrary concept expressions
    - (also role inclusions: $R \sqsubseteq S$ where R,S arbitrary role expressions)

Knowledge base semantics

- An interpretation $\mathcal{I}$ satisfies (models) an axiom A ($\mathcal{I} \models$ A):
  - $\mathcal{I} \models C \sqsubseteq D$ I $\models$ ff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models R \sqsubseteq S$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
  - $\mathcal{I} \models R \equiv S$ iff $R^{\mathcal{I}} = S^{\mathcal{I}}$
  - $\mathcal{I} \models R^{+} \sqsubseteq R$ iff $(R^{\mathcal{I}})^{+} \subseteq R^{\mathcal{I}}$
  - $\mathcal{I} \models x \in D$ iff $x^{\mathcal{I}} \in D^{\mathcal{I}}$
  - $\mathcal{I} \models \langle x,y \rangle \in R$ iff $(x^{\mathcal{I}},y^{\mathcal{I}}) \in R^{\mathcal{I}}$

- $\mathcal{I}$ satisfies a TBox $\mathcal{T}$ ($\mathcal{I} \models \mathcal{T}$) iff $\mathcal{I}$ satisfies every axiom A in $\mathcal{T}$
- $\mathcal{I}$ satisfies an ABox $\mathcal{A}$ ($\mathcal{I} \models \mathcal{A}$) iff $\mathcal{I}$ satisfies every axiom A in $\mathcal{A}$
- $\mathcal{I}$ satisfies a KB $\mathcal{K}$ ($\mathcal{I} \models \mathcal{K}$) iff $\mathcal{I}$ satisfies both $\mathcal{T}$ and $\mathcal{A}$

DL vs. First-Order Logic

- in general, DLs correspond to decidable subclasses of first-order logic (FOL)
- DL KB = first-order theory
- OWL Full is NOT a FOL fragment!
  - reasoning in OWL Full is undecidable
- OWL-DL and OWL-Lite are decidable fragments of FOL

DL vs. First-Order Logic

let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an ontology about persons where:

- $\mathcal{T}$ contains the following inclusion assertions:

  MALE $\sqsubseteq$ PERSON

  FEMALE $\sqsubseteq$ PERSON

  MALE $\sqsubseteq \neg$ FEMALE

  PERSON $\sqsubseteq \exists$Father^{-}.MALE

- $\mathcal{A}$ contains the following instance assertions:

  MALE(Bob)

  PERSON (Mary)

  PERSON(Paul)

DL vs. First-Order Logic

- $\mathcal{T}$ corresponds to the following FOL sentences:

  ∀ x. MALE(x) → PERSON(x)

  ∀ x. FEMALE(x) → PERSON(x)

  ∀ x. MALE(x) → ¬FEMALE(x)

  ∀ x. PERSON(x) → ∃y. Father(y,x) and MALE(y)

- $\mathcal{A}$ corresponds to the following FOL ground atoms:

  MALE(Bob)

  PERSON (Mary)

  PERSON(Paul)

Inference tasks

- Knowledge is correct (captures intuitions)
  - C subsumes D w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- Knowledge is minimally redundant (no unintended synonyms)
  - C is equivalent to D w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $C^{\mathcal{I}} = D^{\mathcal{I}}$
- Knowledge is meaningful (classes can have instances)
  - C is satisfiable w.r.t. $\mathcal{K}$ iff there exists *some* model $\mathcal{I}$ of $\mathcal{K}$ s.t. $C^{\mathcal{I}} \neq \emptyset$
- Querying knowledge
  - x is an instance of C w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $x^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\langle x,y \rangle$ is an instance of R w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $(x^{\mathcal{I}},y^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Knowledge base consistency
  - A KB $\mathcal{K}$ is consistent iff there exists *some* model $\mathcal{I}$ of $\mathcal{K}$

Inference tasks

- OWL-DL ontology = first-order logical theory
- verifying the formal properties of the ontology corresponds to **reasoning** over a first-order theory

Example: ontology consistency

Example TBox:

$MALE \sqsubseteq PERSON$

$FEMALE \sqsubseteq PERSON$

$MALE \sqsubseteq \neg FEMALE$

$PERSON \sqsubseteq \exists hasFather.MALE$

$PERSON \sqsubseteq \exists hasMother.FEMALE$

$hasMother \sqsubseteq hasParent$

$hasFather \sqsubseteq hasParent$

$\exists hasParent.BLACK\text{-}EYES \sqsubseteq BLACK\text{-}EYES$

Example: ontology consistency

Example ABox:

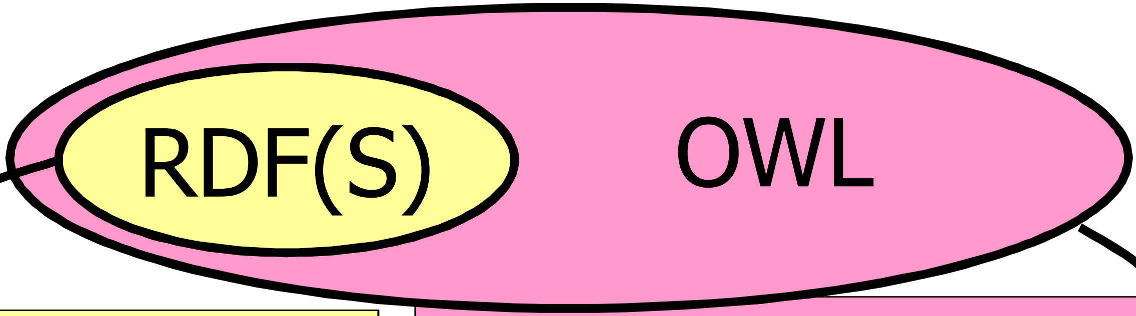
MALE(Bob)
MALE(Paul)
FEMALE(Ann)
hasFather(Ann,Paul)
hasMother(Paul,Mary)
BLACK-EYES(Paul)
¬BLACK-EYES(Ann)

⇒ TBox + ABox **inconsistent** (Ann should have black eyes)

Reasoning in OWL-DL

- reasoning in OWL-DL is decidable (and the complexity is characterized)
- however: high computational complexity (EXPTIME)
- (optimized) reasoning algorithms developed
- OWL-DL reasoning tools implemented

OWL vs. RDFS



- class-def
- subclass-of
- property-def
- subproperty-of
- domain
- range

- class-expressions
  - AND, OR, NOT
- role-constraints
  - has-value, value-type
  - cardinality
- role-properties
  - trans, symm...