FOUNDATIONS OF SEMANTIC WEB TECHNOLOGIES

OWL 2 – Syntax and Semantics

Markus Krötzsch

Dresden, 16 May 2014
Content

Overview & XML 11 APR DS5  Tableau I 23 MAY DS6
Introduction into RDF 11 APR DS6  Tableau II 30 MAY DS5
RDFS – Syntax & Intuition 16 APR DS6  Tutorial 5 30 MAY DS6
Tutorial 1 23 APR DS6  Hypertableau I 4 JUN DS6
RDFS – Semantics 25 APR DS5  Hypertableau II 6 JUN DS5
RDFS Rule-based Reasoning 25 APR DS6  Tutorial 6 6 JUN DS6
Tutorial 2 30 APR DS6  SPARQL 1.1 18 JUN DS6
SPARQL – Syntax & Intuition 02 MAY DS5  SPARQL Entailment 20 JUN DS5
SPARQL – Semantics 02 MAY DS6  Tutorial 7 20 JUN DS6
SPARQL Algebra 09 MAY DS5  OWL & Rules 25 JUN DS6
Tutorial 3 09 MAY DS6  Ontology Editing 27 JUL DS5
OWL – Syntax & Intuition 14 MAY DS6  Ontology Engineering 27 JUL DS6
OWL & Description Logics 16 MAY DS5  Tutorial 8 2 JUL DS6
OWL 2 16 MAY DS6  Linked Data & Applications 4 JUL DS5
Tutorial 4 23 MAY DS5  Q&A Session 9 JUL DS6
Q&A Session 11 JUL DS5

TU Dresden, 16 May 2014  Foundations of Semantic Web Technologies
OWL 2
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic SROIQ
- Inferencing with SROIQ
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $SROIQ$
- Inferencing with $SROIQ$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
Insufficiencies of OWL

OWL still too weak for certain tasks
Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
  \[\rightarrow\] conjunctive queries, SPARQL for OWL
Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
  ~ conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
  ~ FOL-based rule extensions, SWRL & RIF
Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
  $\leadsto$ conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
  $\leadsto$ FOL-based rule extensions, SWRL & RIF

Should the OWL standard itself be extended?
Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
  ~ Conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
  ~ FOL-based rule extensions, SWRL & RIF

Should the OWL standard itself be extended?
~ OWL 2
Development of OWL 2

OWL 2 as “updated version” of OWL

Extensions due to practical experiences with OWL 1.0:
- additional expressivity due to new ontological axioms
- extralogical extensions (syntax, metadata, . . . )
- complete revision of the OWL variants (Lite/DL/Full)

Goals:
- compatibility with the existing OWL standard
- preservation of decidability of OWL DL
- correction of problems in the OWL 1.0 standard
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $SROIQ$
- Inferencing with $SROIQ$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
From $SHOIN$ to $SROIQ$

OWL DL based on DL $SHOIN(D)$:

- **axioms:**
  - TBox: subclass relationships $C \sqsubseteq D$
  - RBox: subrole relationships $R \sqsubseteq S (\mathcal{H})$, inverse roles $R^-(\mathcal{I})$, transitivity
  - ABox: class assertions $C(a)$, role assertions $R(a,b)$, equality $a \approx b$, inequality $a \not\approx b$

- **class constructors:**
  - conjunction $C \sqcap D$, disjunction $C \sqcup D$, negation $\neg C$ of classes
  - role restrictions: universal $\forall R.C$ and existential $\exists R.C$
  - number restrictions ($\mathcal{N}$): $\leq n R$ and $\geq n R$ ($n$ non-negative integer)
  - nominals ($\mathcal{O}$): $\{a\}$

- **datatypes ($D$)**

OWL 2 extends this to $SROIQ(D)$
ABox

\(SHO\text{IN} \) supports different ABox assertions:

- class membership \( C(a) \) (\( C \) complex class),
- special case: negated class membership \( \neg C(a) \) (\( C \) complex class),
- equality \( a \approx b \),
- inequality \( a \not\approx b \)
- role membership \( R(a, b) \)
ABox

SHOIN supports different ABox assertions:

• class membership $C(a)$ ($C$ complex class),
• special case: negated class membership $\neg C(a)$ ($C$ complex class),
• equality $a \approx b$,
• inequality $a \not\approx b$
• role membership $R(a, b)$
• negated role membership?
ABox

\textit{SHOIN} supports different ABox assertions:

- class membership $C(a)$ ($C$ complex class),
- special case: negated class membership $\neg C(a)$ ($C$ complex class),
- equality $a \approx b$,
- inequality $a \not\approx b$
- role membership $R(a, b)$
- negated role membership?

\textit{SROIQ} allows \textbf{negated roles} in der ABox: $\neg R(a, b)$
Number Restrictions

$SHOIN$ supports only unqualified number restrictions ($\forall$):

Person $\sqcap \geq 3 \text{hasChild}$

“class of all persons with 3 or more children”
Number Restrictions

$SHOIN$ supports only unqualified number restrictions ($\geq$):  

Person $\geq 3$ hasChild

“class of all persons with 3 or more children”

$\sim SROIQ$ also allows qualified number restrictions ($\sqcap$):

Person $\geq 3$ hasChild.$(\text{Woman} \sqcap \text{Professor})$

“class of all persons with 3 or more daughters who are professors”
The Self “Concept”

Modeling task: “Every human knows himself/herself.”
The Self “Concept”

Modeling task: “Every human knows himself/herself.”

- **SHOIN:**

  \[
  \text{knows}(\text{tom}, \text{tom}) \quad \text{knows}(\text{tina}, \text{tina}) \quad \text{knows}(\text{udo}, \text{udo}) \quad \ldots
  \]
The Self “Concept”

Modeling task: “Every human knows himself/herself.”

- \textit{SHOIN}:

  \[\text{knows(tom, tom)} \quad \text{knows(tina, tina)} \quad \text{knows(udo, udo)} \quad \ldots\]

  \[\Rightarrow \text{not generally applicable}\]

- \textit{SROIQ}: specific notation \textit{Self}

  \[\text{Human} \sqsubseteq \exists \text{knows}.\text{Self}\]
Role Axioms in \textit{SHOIN}

\textit{SHOIN} provides few role axioms:

- \texttt{Trans}(r), \texttt{owl:TransitiveProperty}: \textit{r} is transitive
  
  Example: \texttt{Trans}(\texttt{locatedIn})
Role Axioms in SHOIN

SHOIN provides few role axioms:

- **Trans(r), owl:TransitiveProperty**: r is transitive
  Example: Trans(locatedIn)

- **Sym(r), owl:SymmetricProperty**: r is symmetric
  Example: Sym(relativeOf)
  also: $r \subseteq r^-$

Role Axioms in \textit{SHOIN}

\textit{SHOIN} provides few role axioms:

- \textbf{Trans}(r), \textit{owl:TransitiveProperty}: \textit{r} is \textit{transitive}
  
  Example: \textbf{Trans}(\textit{locatedIn})

- \textbf{Sym}(r), \textit{owl:SymmetricProperty}: \textit{r} is \textit{symmetric}
  
  Example: \textbf{Sym}(\textit{relativeOf})
  
  also: \textit{r} \sqsubseteq \textit{r}^-

- \textbf{Func}(r), \textit{owl:FunctionalProperty}: \textit{r} is \textit{functional}
  
  Example: \textbf{Func}(\textit{hasFather})
  
  also: \textit{T} \sqsubseteq \leq 1 \textit{r}
Role Axioms in *SHOIN*

*SHOIN* provides few role axioms:

- **Trans**(*r*), *owl:TransitiveProperty*: *r* is transitive
  
  Example: **Trans**(locatedIn)

- **Sym**(*r*), *owl:SymmetricProperty*: *r* is symmetric
  
  Example: **Sym**(relativeOf)
  
  also: *r* ⊑ *r*⁻

- **Func**(*r*), *owl:FunctionalProperty*: *r* is functional
  
  Example: **Func**(hasFather)
  
  also: ⊤ ⊑ ⩽ 1 *r*

- **InvFunc**(*r*), *owl:InverseFunctionalProperty*: *r* is inverse functional
  
  Example: **InvFunc**(isFatherOf)
  
  also ⊤ ⊑ ⩽ 1 *r*⁻ or **Func**(r⁻)
Role Axioms in \textit{SROIQ}

\textit{SROIQ} provides additional statements about roles:

- \texttt{Asym(}r\texttt{), \texttt{owl:AsymmetricProperty}: } \textit{r} \textit{is asymmetric, } (x, y) \in r^I \textit{ implies } (y, x) \notin r^I \\
  \text{Example: } \texttt{Asym(hasChild)}
Role Axioms in \textit{SROIQ}

\textit{SROIQ} provides additional statements about roles:

- \textbf{Asym}($r$), \texttt{owl:AsymmetricProperty}: $r$ is \textit{asymmetric}, $(x, y) \in r^I$ implies $(y, x) \notin r^I$
  
  Example: Asym(hasChild)

- \textbf{Dis}($r, s$), \texttt{owl:propertyDisjointWith,}, \texttt{owl:AllDisjointProperties}: $r$ and $s$ are \textit{disjoint}, $(x, y) \notin r^I \cap s^I$ for all $x, y$
  
  Example: Dis(hasFather, hasSon)
Role Axioms in \textit{SROIQ}

\textit{SROIQ} provides additional statements about roles:

- **Asym** ($r$), \texttt{owl:AsymmetricProperty}: $r$ is \textcolor{red}{asymmetric}, $(x, y) \in r^I$ implies $(y, x) \not\in r^I$
  
  Example: Asym(hasChild)

- **Dis** ($r$, $s$), \texttt{owl:propertyDisjointWith},
  \texttt{owl:AllDisjointProperties}: $r$ and $s$ are \textcolor{red}{disjoint}, $(x, y) \not\in r^I \cap s^I$ for all $x, y$
  
  Example: Dis(hasFather, hasSon)

- **Ref** ($r$), \texttt{owl:ReflexiveProperty}: $r$ is \textcolor{red}{reflexive}, $(x, x) \in r^I$ for all domain individuals $x$
  
  Example: Ref(knows)

  (But does, say, a table really “know” itself? Maybe the least used OWL 2 feature . . . )
Role Axioms in *SROIQ*

*SROIQ* provides additional statements about roles:

- **Asym**(r), *owl:AsymmetricProperty*: r is asymmetric, \((x, y) \in r^I\)
  implies \((y, x) \notin r^I\)
  Example: Asym(hasChild)

- **Dis**(r, s), *owl:propertyDisjointWith*,
  *owl:AllDisjointProperties*: r and s are disjoint, \((x, y) \notin r^I \cap s^I\) for all \(x, y\)
  Example: Dis(hasFather, hasSon)

- **Ref**(r), *owl:ReflexiveProperty*: r is reflexive, \((x, x) \in r^I\) for all domain individuals \(x\)
  Example: Ref(knows)
  (But does, say, a table really “know” itself? Maybe the least used OWL 2 feature . . .)

- **Irr**(r), *owl:IrreflexiveProperty*: r is irreflexive, \((x, x) \notin r^I\) for all domain individuals \(x\)
  Example: Irr(hasChild)
The Universal Role

SROIQ provides the universal role:

- universal role $U$ (owl:TopObjectProperty):
  $(x, y) \in U$ for all $x, y$

Example

$\top \sqsubseteq 7\,000\,000\,000\,U.$ Human
(not recommended!)

$\sim$ $U$ is mainly useful as a counterpart for $\top$, e.g., as root element in a
graphically displayed role hierarchy

- the converse owl:BottomObjectProperty has been introduced in
  OWL, but has no corresponding syntactic element in DLs
  (Exercise: use DL axioms to define an empty role)

- for datatype properties analog owl:TopDataProperty and
  owl:BottomDataProperty
Complex Role Inclusion

“The friends of my friends are my friends.”

\[ \leadsto \text{can be expressed in } SHOIN: \]

hasFriend is transitive

“The enemies of my friends are my enemies.”

\[ \leadsto \text{annot be expressed in } SHOIN \]
Complex Role Inclusion

“The friends of my friends are my friends.”

⇝ can be expressed in \( \text{SHOIN} \): hasFriend is transitive

“The enemies of my friends are my enemies.”

⇝ annot be expressed in \( \text{SHOIN} \)

**complex role inclusion**

- RBox-expressions of the form \( r_1 \circ r_2 \circ \ldots \circ r_n \sqsubseteq s \)
- Semantics: if \( (x_0, x_1) \in r_1^I, (x_1, x_2) \in r_2^I, \ldots, (x_{n-1}, x_n) \in r_n^I \),
  then \( (x_0, x_n) \in s^I \)
Complex Role Inclusions – Example

Example

\[ \text{hasFriend} \circ \text{hasEnemy} \sqsubseteq \text{hasEnemy}: \]
if \((x, y) \in \text{hasFriend}^\mathcal{I}\) and \((y, z) \in \text{hasEnemy}^\mathcal{I}\),
then also \((x, z) \in \text{hasEnemy}^\mathcal{I}\)

Further examples

\[ \text{partOf} \circ \text{belongsTo} \sqsubseteq \text{belongsTo} \quad \text{hasBrother} \circ \text{hasChild} \sqsubseteq \text{isUncleOf} \]
Expressivity of Complex Role Inclusions

How complicated are complex role inclusions?

RBoxes allow for encoding formal languages:

grammar for language of words $ab$, $aabb$, $aaabbb$, …:

\[
L ::= ab \quad \text{becomes the following RBox}
\]

\[
L ::= aLb
\]

In fact, this way, all context-free languages can be encoded. This even enables us to encode the emptiness problem for intersection of two context-free languages into KB satisfiability.

$\leadsto$ OWL with (unrestricted) role inclusions is undecidable.
Can complex role inclusion be restricted in order to retain decidability?

- RBoxes correspond to grammars for context-free languages
- intersection of these problematic

→ restriction to regular languages!
Regularity Conditions for RIAs

In order to guarantee decidability of inferencing, the set of role inclusions has to be regular.

- There has to be a strict linear order $\prec$ over the roles such that every RIA has one of the following forms (with $s_i \prec r$ for all $1 \leq i \leq n$):

  - $r \circ r \sqsubseteq r$
  - $r^- \sqsubseteq r$
  - $s_1 \circ s_2 \circ \ldots \circ s_n \sqsubseteq r$
  - $r \circ s_1 \circ s_2 \circ \ldots \circ s_n \sqsubseteq r$
  - $s_1 \circ s_2 \circ \ldots \circ s_n \circ r \sqsubseteq r$
Regularity Conditions for RIAs

- **Example 1:** \( r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t \)
Regularity Conditions for RIAs

- Example 1: \( r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t \)
  \( \mapsto \) regular with order \( s \prec r \prec t \)

- Example 2: \( r \circ t \circ s \sqsubseteq t \)
Regularity Conditions for RIAs

- Example 1: \( r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t \)
  \( \leadsto \) regular with order \( s \prec r \prec t \)

- Example 2: \( r \circ t \circ s \sqsubseteq t \)
  \( \leadsto \) not regular, form not allowed

- Example 3: \( r \circ s \sqsubseteq s \quad s \circ r \sqsubseteq r \)
Regularity Conditions for RIAs

- **Example 1:** \( r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t \)
  \(\leadsto\) regular with order \( s \prec r \prec t \)

- **Example 2:** \( r \circ t \circ s \sqsubseteq t \)
  \(\leadsto\) not regular, form not allowed

- **Example 3:** \( r \circ s \sqsubseteq s \quad s \circ r \sqsubseteq r \)
  \(\leadsto\) not regular, since no appropriate order exists
Revisiting the Definition of Simple Roles

- simple roles in $SHOIN$ = roles without transitive subroles
- in $SROIQ$ we need to take RIAs into account
Revisiting the Definition of Simple Roles

simple roles are all roles . . .
  • that do not occur on the right of a role inclusion,
  • that are inverses of other simple roles,
  • that occur only on the right of RIAs where the left consists of a length-one chain with a simple role.

(Caution: inductive definition)
→ non-simple are roles that can be derived from a chain of roles with length at least 2
Revisiting the Definition of Simple Roles

Simple roles are all roles... 

- that do not occur on the right of a role inclusion,
- that are inverses of other simple roles,
- that occur only on the right of RIAs where the left consists of a length-one chain with a simple role.

(Caution: inductive definition)

\[ \sim \) non-simple are roles that can be derived from a chain of roles with length at least 2

Expressions \( \leq n \ r \ . \ C \), \( \geq n \ r \ . \ C \), \( \text{Irr} (r) \), \( \text{Dis} (r, s) \), \( \exists r \ . \text{Self} \), \( \neg (a, b) \)

are only allowed for simple roles \( r \) and \( s \! 

(Reason: ensure decidability)
# Overview $SROIQ$ – TBoxes

## Class Expressions

<table>
<thead>
<tr>
<th>Class Names</th>
<th>$A, B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conjunction</td>
<td>$C \sqcap D$</td>
</tr>
<tr>
<td>Disjunction</td>
<td>$C \sqcup D$</td>
</tr>
<tr>
<td>Negation</td>
<td>$\neg C$</td>
</tr>
<tr>
<td>Existential Role Restriction</td>
<td>$\exists r.C$</td>
</tr>
<tr>
<td>Universal Role Restriction</td>
<td>$\forall r.C$</td>
</tr>
<tr>
<td>Self</td>
<td>$\exists s.\text{Self}$</td>
</tr>
<tr>
<td>Atleast Restriction</td>
<td>$\geq n s.C$</td>
</tr>
<tr>
<td>Atmost Restriction</td>
<td>$\leq n s.C$</td>
</tr>
<tr>
<td>Nominals</td>
<td>${a}$</td>
</tr>
</tbody>
</table>

## TBox (Class Axioms)

<table>
<thead>
<tr>
<th>Inclusion</th>
<th>$C \sqsubseteq D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equivalence</td>
<td>$C \equiv D$</td>
</tr>
</tbody>
</table>
# Overview *SROIQ* – RBoxes & ABoxes

## Roles

<table>
<thead>
<tr>
<th>Role Type</th>
<th>Roles</th>
<th>Simple Roles</th>
<th>Universal Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>roles</td>
<td>$r, s, t$</td>
<td>$s, t$</td>
<td>$u$</td>
</tr>
</tbody>
</table>

## ABox (Assertions)

<table>
<thead>
<tr>
<th>Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class membership</td>
<td>$C(a)$</td>
</tr>
<tr>
<td>Role membership</td>
<td>$r(a, b)$</td>
</tr>
<tr>
<td>Neg. role membership</td>
<td>$\neg s(a, b)$</td>
</tr>
<tr>
<td>Equality</td>
<td>$a \approx b$</td>
</tr>
<tr>
<td>Inequality</td>
<td>$a \not\approx b$</td>
</tr>
</tbody>
</table>

## RBox (Role Axioms)

<table>
<thead>
<tr>
<th>Type</th>
<th>Axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inclusion</td>
<td>$r_1 \sqsubseteq r_2$</td>
</tr>
<tr>
<td>Complex role inclusion</td>
<td>$r_1 \circ \ldots \circ r_n \sqsubseteq r$</td>
</tr>
<tr>
<td>Transitivity</td>
<td>$\text{Trans}(r)$</td>
</tr>
<tr>
<td>Symmetry</td>
<td>$\text{Sym}(r)$</td>
</tr>
<tr>
<td>Reflexivity</td>
<td>$\text{Ref}(r)$</td>
</tr>
<tr>
<td>Irreflexivity</td>
<td>$\text{Irr}(s)$</td>
</tr>
<tr>
<td>Disjointness</td>
<td>$\text{Dis}(s, t)$</td>
</tr>
</tbody>
</table>
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $SROIQ$
- Inferencing with $SROIQ$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
How complicated is $SROIQ$?

Recap: $SHOIN$ (OWL DL) is very complex (NExpTime)
How complicated is $SROIQ$?

Recap: $SHOIN$ (OWL DL) is very complex ($N\text{ExpTime}$)
Observation: some modeling features are not really necessary ("syntactic sugar")

- $\text{Trans}(r)$ can be expressed as $r \circ r \sqsubseteq r$
- $\text{Sym}(r)$ can be expressed as $r^- \sqsubseteq r$
- $\text{Asym}(r)$ can be expressed as $\text{Dis}(r, r^-)$
- $\text{Irr}(s)$ can be expressed as $\top \sqsubseteq \neg \exists S.\text{Self}$
- $\text{ABox}$ can be represented by $\text{TBox}$ axioms with nominals, e.g. $r(a, b)$ becomes $\{a\} \sqsubseteq \exists r.\{b\}$

Qualified number restrictions do not cause problems (known and implemented before)

$\rightsquigarrow$ main problem: role axioms ($\text{RBox}$)
Role Inclusions, Languages, Automata

How to deal with RBoxes?

- RBox inclusions resemble formal grammars
- every role $r$ defines a regular language: the language of role chains from which it follows
- regular languages $\equiv$ regular expressions $\equiv$ finite automata

$\Rightarrow$ approach: tableau methods are extended by "RBox automata"
Decidability of $SROIQ$

Tableau method for $SROIQ$ shows decidability

- Algorithm has a good adaptation behaviour: modeling features that are not used do hardly impede computation ("pay as you go")
- Tableau method not useful for complexity considerations
- $SROIQ$ N2ExpTime-complete
  - Lower bound: encoding of a 2Exp tiling problem
  - Upper bound: exponential translation into the 2-variable fragment of FOL with counting quantifiers, $C_2$, for which satisfiability checking is known to be NExpTime-complete)
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $SROIQ$
- Inferencing with $SROIQ$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
OWL 2 DL: Further Aspects

SROIQ is “only” logical foundation of OWL 2 DL

Further non-logical aspects:
- Syntax (extension necessary)
- Datatype declarations and datatype functions, new datatypes
- Metamodelling: “punning”
- Comments and ontological metadata
- Inverse-functional concrete roles (datatype properties): Keys
- Mechanisms for ontology import
- ... various smaller changes
Metamodeling

Specification of ontological knowledge about elements of the ontology (including classes, roles, axioms).

Examples:

- “The class Person was created on the 1st Jan 2008 by bglimm.”
- “For the class City, we recommend the property numberOfCitizens.”
- “The statement ‘Dresden was founded in 1206’ was extracted automatically with a confidence of 85%.”

(Compare Reification in RDF Schema)
Punning in OWL

Metamodelling in expressive logics is dangerous and expensive!

OWL 2 currently supports the simplest form of metamodelling:

**Punning**

- the names for classes, roles, individuals do not have to be disjoint
- no logical relationship between class, individual and role of the same name
- only relevant for pragmatic interpretation

Example:

Person(Birte) classCreatedBy(Person, bglimm)
Comments and Metadata

Punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

• comments in XML files: <!-- comment -->
Comments and Metadata

Punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

- comments in XML files: `<!-- comment -->`
  - no relation to the OWL axioms in this file
- non-logical annotations in OWL 2: `owl:AnnotationProperty`
Comments and Metadata

Punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

- comments in XML files: `<!-- comment -->`
  → no relation to the OWL axioms in this file
- non-logical annotations in OWL 2: `owl:AnnotationProperty`
  → attached to (semantic) ontological element
Syntactic Aspects

New/extended syntaxes:

- RDF/XML: extension by OWL 2 elements
- functional-style syntax: replaces “abstract syntax” in OWL 1
- OWL/XML: syntax for simpler processing in XML tools
- Turtle: RDF triple syntax
- Manchester syntax: syntax that is easier to read for humans
Quo vadis, OWL Lite?

OWL Lite as a Failure:

• almost as complex as OWL DL
• complicated syntax that does not provide direct access to actual modeling power
• use in ontologies only “by accident”, not deliberately

Original goal:
capture the part of OWL that is easy and efficiently implementable

⇝ OWL 2 Profiles
Quo vadis, OWL Lite?

OWL Lite as a Failure:

- almost as complex as OWL DL
- complicated syntax that does not provide direct access to actual modeling power
- use in ontologies only “by accident”, not deliberately

Original goal:
capture the part of OWL that is easy and efficiently implementable
Quo vadis, OWL Lite?

OWL Lite as a Failure:
- almost as complex as OWL DL
- complicated syntax that does not provide direct access to actual modeling power
- use in ontologies only “by accident”, not deliberately

Original goal:
capture the part of OWL that is easy and efficiently implementable

⇒ OWL 2 Profiles
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $\mathcal{SROIQ}$
- Inferencing with $\mathcal{SROIQ}$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- OWL EL
  - computation of the class hierarchy (all subclass relationships) in PTime
OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- OWL EL
  - computation of the class hierarchy (all subclass relationships) in PTime

- OWL QL
  - conjunctive queries in $AC_0$ (data complexity) $\leadsto$ reducible to SQL
OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- **OWL EL**
  - computation of the class hierarchy (all subclass relationships) in PTime

- **OWL QL**
  - conjunctive queries in $AC_0$ (data complexity) $\leadsto$ reducible to SQL

- **OWL RL**
  - can be used as an extension of RDFS or as a fragment of OWL DL (OWL Direct Semantics)
  - complexity PTime
OWL 2 EL

- An (almost maximal) fragment of OWL 2 such that
  - satisfiability can be checked in PTime (PTime-complete)
  - data complexity for ABox queries also PTime-complete
- Class hierarchy (all subsumption relationships between atomic classes) can be computed in one pass
- Reasoning based on saturation methods first developed for the description logic $\mathcal{EL}$
  (with significant contributions from researchers at TU Dresden . . .)
OWL 2 EL

- **Allowed:**
  - subclass axioms with conjunction, existential restriction, $\top$, $\bot$, singleton nominals
  - complex RIAs, range restrictions (under certain conditions)

- **Not allowed:**
  - negation, disjunction, universal restrictions, inverse roles
OWL 2 QL

- An (almost maximal) fragment of OWL 2 such that
  - data complexity of conjunctive query answering is in $\text{AC}^0$
- Queries can be rewritten such that no terminological knowledge has to be taken into account
  $\Rightarrow$ standard RDBMS can be used for storage and querying
OWL 2 QL

- **Allowed:**
  - simple role hierarchies, domain & range axioms
  - subclass axioms with
    - left: class name or existential restriction with $\top$
    - right: conjunction of class names, existential restriction and negation of left expressions
- **Not allowed:** everything else
- Supports RDFS with “standard use” graphs (like all OWL profiles)
OWL 2 RL

• An (almost maximal) fragment of OWL 2 such that
  – automated inferencing is PTime-complete (consistency, satisfiability of classes, subsumption, class membership checks)
  – automated inferencing is correct (sound & complete) if the given RDF graph satisfies certain requirements
  – otherwise the automated reasoning may be sound but incomplete.

• Can operate directly on RDF triples in order to enrich instance data (materialization, forward chaining for facts)

• Automated inferencing can be implemented via a set of rules (using a rule engine that supports equality)
Agenda

• Recap OWL & Overview OWL 2
• The Description Logic $SROIQ$
• Inferencing with $SROIQ$
• OWL 2 DL
• OWL 2 Profiles
• OWL 2 Full
• Summary
What to do with OWL Full?

Goal of OWL 2 DL: make many OWL Full 1.0 ontologies interpretable as OWL DL (cf., e.g., punning)
What to do with OWL Full?

Goal of OWL 2 DL: make many OWL Full 1.0 ontologies interpretable as OWL DL (cf., e.g., punning)

- extension of OWL Full by OWL 2 features is required by a few practitioners
- allows to work on all kinds of RDF graphs
- despite undecidability: many FOL verification tools do not guarantee termination and are still useful
- alternative implementation techniques can be used, which might be faster (but do not guarantee termination)
Crucial Differences in the Semantics

- annotations do not have a semantics in the direct semantics (which is used for OWL DL), but they do in the RDF-based semantics (which is used for OWL Full)
- import commands are only parser commands in the direct semantics, but do have a presence as triple in the RDF-based Semantics
- in the RDF-based semantics, classes are individuals, that are endowed with an extension ~ semantic conditions are only applicable to those classes that have an individual representative
Example

- \( C(a) \)
- query for all instances of the class \( C \sqcup D \)
Crucial Differences in the Semantics

**Example**

- C(a)
- query for all instances of the class C \( \sqcup \) D
- RDF-based semantics: \( \emptyset \), direct semantics: a
Crucial Differences in the Semantics

Example

- C(a)
- query for all instances of the class C ⊔ D
- RDF-based semantics: ∅, direct semantics: a

⇝ under the RDF-based semantics, we only have the guarantee that the union of the extensions of C and D do exist as subsets of the domain, however it is not ensured that an element exists which has this set as extension.

⇝ contrarily, in the direct semantics class names “directly” represent sets and not domain elements

⇝ the answer coincides for both semantics after adding E ≡ C ⊔ D
Agenda

- Recap OWL & Overview OWL 2
- The Description Logic $SROIQ$
- Inferencing with $SROIQ$
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary
Summary

OWL 2 as first extension of the OWL standard

- Standardized 27th Oct 2009
- Logical extension based on description logic $SROIQ$
- New modeling features, most notably complex RIs, qualified number restrictions
- Non-logical extensions: punning, comments, datatypes, etc.
- Profiles with polynomial reasoning procedures