Abstract

The OWL 2 Web Ontology Language, informally OWL 2, is an ontology language for the Semantic Web with formally defined meaning. OWL 2 ontologies provide classes, properties, individuals, and data values and are stored as Semantic Web documents. OWL 2 ontologies can be used along with information written in RDF, and OWL 2 ontologies themselves are primarily exchanged as RDF documents. The OWL 2 Document Overview describes the overall state of OWL 2, and should be read before other OWL 2 documents.

This document provides the direct model-theoretic semantics for OWL 2, which is compatible with the description logic SROIQ. Furthermore, this document defines the most common inference problems for OWL 2.
Status of this Document

May Be Superseded

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

Summary of Changes

This Last Call Working Draft has undergone only minor editorial changes since the previous version of 21st April, 2009.

Please Comment By 16 July 2009

The OWL Working Group seeks public feedback on this Working Draft. Please send your comments to public-owl-comments@w3.org (public archive). If possible, please offer specific changes to the text that would address your concern. You may also wish to check the Wiki Version of this document and see if the relevant text has already been updated.

No Endorsement

Publication as a Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

Patents

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

- 1 Introduction
- 2 Direct Model-Theoretic Semantics for OWL 2
  - 2.1 Vocabulary
1 Introduction

This document defines the direct model-theoretic semantics of OWL 2. The semantics given here is strongly related to the semantics of description logics [Description Logics] and it extends the semantics of the description logic SROIQ [SROIQ]. As the definition of SROIQ does not provide for datatypes and punning, the semantics of OWL 2 is defined directly on the constructs of the structural specification of OWL 2 [OWL 2 Specification] instead of by reference to SROIQ. For the constructs available in SROIQ, the semantics of SROIQ trivially corresponds to the one defined in this document.

Since each OWL 1 DL ontology is an OWL 2 ontology, this document also provides a direct semantics for OWL 1 Lite and OWL 1 DL ontologies; this semantics is equivalent to the direct model-theoretic semantics of OWL 1 Lite and OWL 1 DL [OWL 1 Semantics and Abstract Syntax]. Furthermore, this document also provides the direct model-theoretic semantics for the OWL 2 profiles [OWL 2 Profiles].

The semantics is defined for OWL 2 axioms and ontologies, which should be understood as instances of the structural specification [OWL Specification]. Parts of the structural specification are written in this document using the functional-style syntax.

OWL 2 allows ontologies, anonymous individuals, and axioms to be annotated; furthermore, annotations themselves can contain additional annotations. All these types of annotations, however, have no semantic meaning in OWL 2 and are
ignored in this document. OWL 2 declarations are used only to disambiguate class expressions from data ranges and object property from data property expressions in the functional-style syntax; therefore, they are not mentioned explicitly in this document.

2 Direct Model-Theoretic Semantics for OWL 2

This section specifies the direct model-theoretic semantics of OWL 2 ontologies.

2.1 Vocabulary

A datatype map, formalizing datatype maps from the OWL 2 Specification [OWL 2 Specification], is a 6-tuple $D = (\ N_{DT} , \ N_{LS} , \ N_{FS} , \ \cdot \ DT , \ \cdot \ LS , \ \cdot \ FS)$ with the following components:

- $N_{DT}$ is a set of datatypes (more precisely, names of datatypes) that does not contain the datatype rdfs:Literal.
- $N_{LS}$ is a function that assigns to each datatype $DT \in N_{DT}$ a set $N_{LS}(DT)$ of strings called lexical forms. The set $N_{LS}(DT)$ is called the lexical space of $DT$.
- $N_{FS}$ is a function that assigns to each datatype $DT \in N_{DT}$ a set $N_{FS}(DT)$ of pairs $(F, v)$, where $F$ is a constraining facet and $v$ is an arbitrary data value called the constraining value. The set $N_{FS}(DT)$ is called the facet space of $DT$.
- For each datatype $DT \in N_{DT}$, the interpretation function $\cdot \ DT$ assigns to $DT$ a set $\ N_{DT} (\cdot \ DT)$ called the value space of $DT$.
- For each datatype $DT \in N_{DT}$ and each lexical form $LV \in N_{LS}(DT)$, the interpretation function $\cdot \ LS$ assigns to the pair $(LV, DT)$ a data value $(LV, DT)^{LS} \in (DT)^{DT}$.
- For each datatype $DT \in N_{DT}$ and each pair $(F, v) \in N_{FS}(DT)$, the interpretation function $\cdot \ FS$ assigns to $(F, v)$ the set $(F, v)^{FS} \subseteq (DT)^{DT}$.

A vocabulary $V = (V_{C}, V_{OP}, V_{DP}, V_{I}, V_{DT}, V_{LT}, V_{FA})$ over a datatype map $D$ is a 7-tuple consisting of the following elements:

- $V_{C}$ is a set of classes as defined in the OWL 2 Specification [OWL 2 Specification], containing at least the classes owl:Thing and owl:Nothing.
- $V_{OP}$ is a set of object properties as defined in the OWL 2 Specification [OWL 2 Specification], containing at least the object properties owl:topObjectProperty and owl:bottomObjectProperty.
- $V_{DP}$ is a set of data properties as defined in the OWL 2 Specification [OWL 2 Specification], containing at least the data properties owl:topDataProperty and owl:bottomDataProperty.
- $V_{I}$ is a set of individuals (named and anonymous) as defined in the OWL 2 Specification [OWL 2 Specification].
- $V_{DT}$ is a set containing all datatypes of $D$, the datatype rdfs:Literal, and possibly other datatypes; that is, $N_{DT} \cup \{\text{rdfs:Literal}\} \subseteq V_{DT}$. 
• \( V_L T \) is a set of literals \( LV^{\wedge DT} \) for each datatype \( DT \in ND_T \) and each lexical form \( LV \in NL_S(DT) \).

• \( V_F A \) is the set of pairs \(( F, lt)\) for each constraining facet \( F \), datatype \( DT \in ND_T \), and literal \( lt \in V_L T \) such that \(( F, (LV, DT) ^ {LS} ) \in NFS(DT) \), where \( LV \) is the lexical form of \( lt \) and \( DT_1 \) is the datatype of \( lt \).

Given a vocabulary \( V \), the following conventions are used in this document to denote different syntactic parts of OWL 2 ontologies:

• \( OP \) denotes an object property;

• \( OPE \) denotes an object property expression;

• \( DP \) denotes a data property;

• \( DPE \) denotes a data property expression;

• \( C \) denotes a class;

• \( CE \) denotes a class expression;

• \( DT \) denotes a datatype;

• \( DR \) denotes a data range;

• \( a \) denotes an individual (named or anonymous);

• \( lt \) denotes a literal; and

• \( F \) denotes a constraining facet.

2.2 Interpretations

Given a datatype map \( D \) and a vocabulary \( V \) over \( D \), an interpretation \( I = (\Delta_I, \Delta_D, \cdot \ C, \cdot \ OP, \cdot \ DP, \cdot \ I, \cdot \ DT, \cdot \ LT, \cdot \ FA) \) for \( D \) and \( V \) is a 9-tuple with the following structure:

• \( \Delta_I \) is a nonempty set called the object domain.

• \( \Delta_D \) is a nonempty set disjoint with \( \Delta_I \) called the data domain such that \( (DT)^{DT} \subseteq \Delta_D \) for each datatype \( DT \in V_D T \).

• \( \cdot \ C \) is the class interpretation function that assigns to each class \( C \in VC \) a subset \((C)^{C} \subseteq \Delta_I \) such that
  • \((\text{owl:Thing})^{C} = \Delta_I \) and
  • \((\text{owl:Nothing})^{C} = \emptyset \).

• \( \cdot \ OP \) is the object property interpretation function that assigns to each object property \( OP \in V_OP \) a subset \((OP)^{OP} \subseteq \Delta_I \times \Delta_I \) such that
  • \((\text{owl:topObjectProperty})^{OP} = \Delta_I \times \Delta_I \) and
  • \((\text{owl:bottomObjectProperty})^{OP} = \emptyset \).

• \( \cdot \ DP \) is the data property interpretation function that assigns to each data property \( DP \in V_DP \) a subset \((DP)^{DP} \subseteq \Delta_I \times \Delta_D \) such that
  • \((\text{owl:topDataProperty})^{DP} = \Delta_I \times \Delta_D \) and
  • \((\text{owl:bottomDataProperty})^{DP} = \emptyset \).

• \( \cdot \ I \) is the individual interpretation function that assigns to each individual \( a \in V_I \) an element \((a)^{I} \in \Delta_I \).

• \( \cdot \ DT \) is the datatype interpretation function that assigns to each datatype \( DT \in V_DT \) a subset \((DT)^{DT} \subseteq \Delta_D \) such that
  • \((\text{rdfs:Literal})^{DT} = \Delta_D \) is the same as in \( D \) for each datatype \( DT \in ND_T \), and
  • \((\text{rdfs:Literal})^{DT} = \Delta_D \).
• $\cdot_{LT}$ is the literal interpretation function that is defined as $(lt)^{LT} = (LV, DT)^{LS}$ for each $lt \in V_{LT}$, where $LV$ is the lexical form of $lt$ and $DT$ is the datatype of $lt$.

• $\cdot_{FA}$ is the facet interpretation function that is defined as $(F, lt)^{FA} = (F, (lt)^{LT})^{FS}$ for each $(F, lt) \in V_{FA}$.

The following sections define the extensions of $\cdot_{OP}$, $\cdot_{DT}$, and $\cdot_{C}$ to object property expressions, data ranges, and class expressions.

### 2.2.1 Object Property Expressions

The object property interpretation function $\cdot_{OP}$ is extended to object property expressions as shown in Table 1.

**Table 1. Interpreting Object Property Expressions**

<table>
<thead>
<tr>
<th>Object Property Expression</th>
<th>Interpretation $\cdot_{OP}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectInverseOf( $OP$ )</td>
<td>${(x, y)</td>
</tr>
</tbody>
</table>

### 2.2.2 Data Ranges

The datatype interpretation function $\cdot_{DT}$ is extended to data ranges as shown in Table 3. All datatypes in OWL 2 are unary, so each datatype $DT$ is interpreted as a unary relation over $\Delta_D$ — that is, as a set $(DT)^{DT} \subseteq \Delta_D$. OWL 2 currently does not define data ranges of arity more than one; however, by allowing for $n$-ary data ranges, the syntax of OWL 2 provides a "hook" allowing implementations to introduce extensions such as comparisons and arithmetic. An $n$-ary data range $DR$ is interpreted as an $n$-ary relation $(DR)^{DT}$ over $\Delta_D$ — that is, as a set $(DT)^{DT} \subseteq (\Delta_D)^n$.

**Table 3. Interpreting Data Ranges**

<table>
<thead>
<tr>
<th>Data Range</th>
<th>Interpretation $\cdot_{DT}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataIntersectionOf( $DR_1 \ldots DR_n$ )</td>
<td>$(DR_1)^{DT} \cap \ldots \cap (DR_n)^{DT}$</td>
</tr>
<tr>
<td>DataUnionOf( $DR_1 \ldots DR_n$ )</td>
<td>$(DR_1)^{DT} \cup \ldots \cup (DR_n)^{DT}$</td>
</tr>
<tr>
<td>DataComplementOf( $DR$ )</td>
<td>$(\Delta_D)^n \setminus (DR)^{DT}$ where $n$ is the arity of $DR$</td>
</tr>
<tr>
<td>DataOneOf( $lt_1 \ldots lt_n$ )</td>
<td>${(lt_1)^{LT}, \ldots, (lt_n)^{LT}}$</td>
</tr>
<tr>
<td>DatatypeRestriction( $DT F_1 lt_1 \ldots F_n lt_n$ )</td>
<td>$(DT)^{DT} \cap (F_1, lt_1)^{FA} \cap \ldots \cap (F_n, lt_n)^{FA}$</td>
</tr>
</tbody>
</table>
### 2.2.3 Class Expressions

The class interpretation function $\cdot^C$ is extended to class expressions as shown in Table 4. For $S$ a set, $\#S$ denotes the number of elements in $S$.

<table>
<thead>
<tr>
<th>Class Expression</th>
<th>Interpretation $\cdot^C$</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ObjectIntersectionOf(CE_1 ... CE_n)</code></td>
<td>$(CE_1)^C \cap ... \cap (CE_n)^C$</td>
</tr>
<tr>
<td><code>ObjectUnionOf(CE_1 ... CE_n)</code></td>
<td>$(CE_1)^C \cup ... \cup (CE_n)^C$</td>
</tr>
<tr>
<td><code>ObjectComplementOf(CE)</code></td>
<td>$\Delta_I \setminus (CE)^C$</td>
</tr>
<tr>
<td><code>ObjectOneOf(a_1 ... a_n)</code></td>
<td>${(a_1)^I, ..., (a_n)^I}$</td>
</tr>
<tr>
<td><code>ObjectSomeValuesFrom(OPE CE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectAllValuesFrom(OPE CE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectHasValue(OPE a)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectHasSelf(OPE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectMinCardinality(n OPE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectMaxCardinality(n OPE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectExactCardinality(n OPE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectMinCardinality(n OPE CE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectMaxCardinality(n OPE CE)</code></td>
<td>${x</td>
</tr>
<tr>
<td><code>ObjectExactCardinality(n OPE CE)</code></td>
<td>${x</td>
</tr>
</tbody>
</table>
Data Some Values From (DPE₁ ... DPEₙ DR) = \{ x | ∃ y₁, ..., yₙ : (x, yₖ) ∈ (DPEₖ)ₜ for each 1 ≤ k ≤ n and (y₁, ..., yₙ) ∈ (DR)ₜ \}

Data All Values From (DPE₁ ... DPEₙ DR) = \{ x | ∀ y₁, ..., yₙ : (x, yₖ) ∈ (DPEₖ)ₜ for each 1 ≤ k ≤ n imply (y₁, ..., yₙ) ∈ (DR)ₜ \}

Data Has Value (DPE lt) = \{ x | (x, (lt)) ∈ (DPE)ₜ \}

Data Min Cardinality (n DPE) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ \} ≥ n \}

Data Max Cardinality (n DPE) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ \} ≤ n \}

Data Exact Cardinality (n DPE) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ \} = n \}

Data Min Cardinality (n DPE DR) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ and y ∈ (DR)ₜ \} ≥ n \}

Data Max Cardinality (n DPE DR) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ and y ∈ (DR)ₜ \} ≤ n \}

Data Exact Cardinality (n DPE DR) = \{ x | \#\{ y | (x, y) ∈ (DPE)ₜ and y ∈ (DR)ₜ \} = n \}

### 2.3 Satisfaction in an Interpretation

An interpretation I = (Δᵣ, Δₑ, C, OP, DP, I, DT, LT, FA) satisfies an axiom w.r.t. an ontology O if the axiom satisfies the relevant condition from the following sections. Satisfaction of axioms in I is defined w.r.t. O because satisfaction of key axioms uses the following function:

ISNAMEDₜO(x) = true for x ∈ Δᵣ if and only if (a)ᵣ = x for some named individual a occurring in the axiom closure of O.

### 2.3.1 Class Expression Axioms

Satisfaction of OWL 2 class expression axioms in I w.r.t. O is defined as shown in Table 5.

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubClassOf(CE₁ CE₂)</td>
<td>(CE₁)ᶜ ⊆ (CE₂)ᶜ</td>
</tr>
</tbody>
</table>
EquivalentClasses( Ces ... Cesn )
(Cesj)C = (Cesk)C for each 1 ≤ j ≤ n and each 1 ≤ k ≤ n

DisjointClasses( Ces ... Cesn )
(Cesj)C ∩ (Cesk)C = ∅ for each 1 ≤ j ≤ n and each 1 ≤ k ≤ n such that j ≠ k

DisjointUnion( C Ces ... Cesn )
(C)C = (Ces1)C ∪ ... ∪ (Cesn)C and
(Cesj)C ∩ (Cesk)C = ∅ for each 1 ≤ j ≤ n and each 1 ≤ k ≤ n such that j ≠ k

2.3.2 Object Property Expression Axioms

Satisfaction of OWL 2 object property expression axioms in I w.r.t. O is defined as shown in Table 6.

Table 6. Satisfaction of Object Property Expression Axioms in an Interpretation

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubObjectPropertyOf( OPE1 OPE2 )</td>
<td>(OPE1)OP ⊆ (OPE2)OP</td>
</tr>
<tr>
<td>SubObjectPropertyOf( ObjectPropertyChain( OPE1 ... OPEn ) OPE )</td>
<td>∀ y0, ..., yn : (y0, y1) ∈ (OPE1)OP and ... and (yn-1, yn) ∈ (OPEn)OP imply (y0, yn) ∈ (OPE)OP</td>
</tr>
<tr>
<td>EquivalentObjectProperties( OPE1 ... OPEn )</td>
<td>(OPEj)OP = (OPEk)OP for each 1 ≤ j ≤ n and each 1 ≤ k ≤ n</td>
</tr>
<tr>
<td>DisjointObjectProperties( OPE1 ... OPEn )</td>
<td>(OPEj)OP ∩ (OPEk)OP = ∅ for each 1 ≤ j ≤ n and each 1 ≤ k ≤ n such that j ≠ k</td>
</tr>
<tr>
<td>ObjectPropertyDomain( OPE CE )</td>
<td>∀ x, y : (x, y) ∈ (OPE)OP implies x ∈ (CE)C</td>
</tr>
<tr>
<td>ObjectPropertyRange( OPE CE )</td>
<td>∀ x, y : (x, y) ∈ (OPE)OP implies y ∈ (CE)C</td>
</tr>
<tr>
<td>InverseObjectProperties( OPE1 OPE2 )</td>
<td>(OPE1)OP = { (x, y)</td>
</tr>
<tr>
<td>FunctionalObjectProperty( OPE )</td>
<td>∀ x, y1, y2 : (x, y1) ∈ (OPE)OP and (x, y2) ∈ (OPE)OP imply y1 = y2</td>
</tr>
<tr>
<td>InverseFunctionalObjectProperty( OPE )</td>
<td>∀ x1, x2, y : (x1, y) ∈ (OPE)OP and (x2, y) ∈ (OPE)OP imply x1 = x2</td>
</tr>
</tbody>
</table>
### 2.3.3 Data Property Expression Axioms

Satisfaction of OWL 2 data property expression axioms in \( I \) w.r.t. \( O \) is defined as shown in Table 7.

**Table 7. Satisfaction of Data Property Expression Axioms in an Interpretation**

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubDataPropertyOf( DPE&lt;sub&gt;1&lt;/sub&gt;, DPE&lt;sub&gt;2&lt;/sub&gt; )</td>
<td>((DPE&lt;sub&gt;1&lt;/sub&gt;)^{\text{DP}} \subseteq (DPE&lt;sub&gt;2&lt;/sub&gt;)^{\text{DP}})</td>
</tr>
<tr>
<td>EquivalentDataProperties( DPE&lt;sub&gt;1&lt;/sub&gt; ... DPE&lt;sub&gt;n&lt;/sub&gt; )</td>
<td>((DPE&lt;sub&gt;j&lt;/sub&gt;)^{\text{DP}} = (DPE&lt;sub&gt;k&lt;/sub&gt;)^{\text{DP}} for each 1 \leq j \leq n and each 1 \leq k \leq n)</td>
</tr>
<tr>
<td>DisjointDataProperties( DPE&lt;sub&gt;1&lt;/sub&gt; ... DPE&lt;sub&gt;n&lt;/sub&gt; )</td>
<td>((DPE&lt;sub&gt;j&lt;/sub&gt;)^{\text{DP}} \cap (DPE&lt;sub&gt;k&lt;/sub&gt;)^{\text{DP}} = \emptyset for each 1 \leq j \leq n and each 1 \leq k \leq n such that j \neq k)</td>
</tr>
<tr>
<td>DataPropertyDomain( DPE CE )</td>
<td>(\forall x, y: (x, y) \in (DPE)^{\text{DP}} \text{ implies } x \in (CE)^{C})</td>
</tr>
<tr>
<td>DataPropertyRange( DPE DR )</td>
<td>(\forall x, y: (x, y) \in (DPE)^{\text{DP}} \text{ implies } y \in (DR)^{DT})</td>
</tr>
<tr>
<td>FunctionalDataProperty( DPE )</td>
<td>(\forall x, y_1, y_2: (x, y_1) \in (DPE)^{\text{DP}} \text{ and } (x, y_2) \in (DPE)^{\text{DP}} \text{ imply } y_1 = y_2)</td>
</tr>
</tbody>
</table>

### 2.3.4 Datatype Definitions

Satisfaction of datatype definitions in \( I \) w.r.t. \( O \) is defined as shown in Table 8.
Table 8. Satisfaction of Datatype Definitions in an Interpretation

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatatypeDefinition( DT DR )</td>
<td>((DT)^{DT} = (DR)^{DT})</td>
</tr>
</tbody>
</table>

2.3.5 Keys

Satisfaction of keys in \(I\) w.r.t. \(O\) is defined as shown in Table 9.

Table 9. Satisfaction of Keys in an Interpretation

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasKey( CE ( OPE_1 ... OPE_m ) ( DPE_1 ... DPE_n ) )</td>
<td>(\forall x, y, z_1, ..., z_m, w_1, ..., w_n:) if (x \in (CE)^C) and (ISNAMED_O(x)) and (y \in (CE)^C) and (ISNAMED_O(y)) and ((x, z_i) \in (OPE)^{OP}) and ((y, z_i) \in (OPE)^{OP}) and (ISNAMED_O(z_i)) for each (1 \leq i \leq m) and ((x, w_j) \in (DPE)^{DP}) and ((y, w_j) \in (DPE)^{DP}) for each (1 \leq j \leq n) then (x = y)</td>
</tr>
</tbody>
</table>

2.3.6 Assertions

Satisfaction of OWL 2 assertions in \(I\) w.r.t. \(O\) is defined as shown in Table 10.

Table 10. Satisfaction of Assertions in an Interpretation

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SameIndividual( a_1 ... a_n )</td>
<td>((a_j)^I = (a_k)^I) for each (1 \leq j \leq n) and each (1 \leq k \leq n)</td>
</tr>
<tr>
<td>DifferentIndividuals( a_1 ... a_n )</td>
<td>((a_j)^I \neq (a_k)^I) for each (1 \leq j \leq n) and each (1 \leq k \leq n) such that (j \neq k)</td>
</tr>
<tr>
<td>ClassAssertion( CE a )</td>
<td>((a)^I \in (CE)^C)</td>
</tr>
<tr>
<td>ObjectPropertyAssertion( OPE a_1 a_2 )</td>
<td>((a_1)^I, (a_2)^I \in (OPE)^{OP})</td>
</tr>
<tr>
<td>NegativeObjectPropertyAssertion( OPE a_1 a_2 )</td>
<td>((a_1)^I, (a_2)^I \notin (OPE)^{OP})</td>
</tr>
<tr>
<td>DataPropertyAssertion( DPE a lt )</td>
<td>((a)^I, (lt)^{LT} \in (DPE)^{DP})</td>
</tr>
</tbody>
</table>
2.3.7 Ontologies

An interpretation $I$ satisfies an OWL 2 ontology $O$ if all axioms in the axiom closure of $O$ (with anonymous individuals standardized apart as described in Section 5.6.2 of the OWL 2 Specification [OWL 2 Specification]) are satisfied in $I$ w.r.t. $O$.

2.4 Models

Given a datatype map $D$, an interpretation $I = (\Delta_I, \Delta_D, \cdot C, \cdot OP, \cdot DP, \cdot I, \cdot DT, \cdot LT, \cdot FA)$ for $D$ is a model of an OWL 2 ontology $O$ w.r.t. $D$ if an interpretation $J = (\Delta_I, \Delta_D, \cdot C, \cdot OP, \cdot DP, \cdot J, \cdot DT, \cdot LT, \cdot FA)$ for $D$ exists such that $J$ coincides with $I$ on all named individuals and $J$ satisfies $O$.

Thus, an interpretation $I$ satisfying $O$ is also a model of $O$. In contrast, a model $I$ of $O$ may not satisfy $O$ directly; however, by modifying the interpretation of anonymous individuals, $I$ can always be coerced into an interpretation $J$ that satisfies $O$.

2.5 Inference Problems

Let $D$ be a datatype map and $V$ a vocabulary over $D$. Furthermore, let $O$ and $O_1$ be OWL 2 ontologies, $CE$, $CE_1$, and $CE_2$ class expressions, and $a$ a named individual, such that all of them refer only to the vocabulary elements in $V$. Furthermore, variables are symbols that are not contained in $V$. Finally, a Boolean conjunctive query $Q$ is a closed formula of the form

$$\exists x_1, \ldots, x_n, y_1, \ldots, y_m : [A_1 \land \ldots \land A_k]$$

where each $A_i$ is an atom of the form $C(s), OP(s, t), or DP(s, u)$ with $C$ a class, $OP$ an object property, $DP$ a data property, $s$ and $t$ individuals or some variable $x_j$, and $u$ a literal or some variable $y_j$.

The following inference problems are often considered in practice.

**Ontology Consistency**: $O$ is consistent (or satisfiable) w.r.t. $D$ if a model of $O$ w.r.t. $D$ and $V$ exists.

**Ontology Entailment**: $O$ entails $O_1$ w.r.t. $D$ if every model of $O$ w.r.t. $D$ and $V$ is also a model of $O_1$ w.r.t. $D$ and $V$.

**Ontology Equivalence**: $O$ and $O_1$ are equivalent w.r.t. $D$ if $O$ entails $O_1$ w.r.t. $D$ and $O_1$ entails $O$ w.r.t. $D$. 
Ontology Equisatisfiability: $O$ and $O_1$ are *equisatisfiable* w.r.t. $D$ if $O$ is satisfiable w.r.t. $D$ if and only if $O_1$ is satisfiable w.r.t. $D$.

Class Expression Satisfiability: $CE$ is satisfiable w.r.t. $O$ and $D$ if a model $I = (\Delta_I, \Delta_D, C^*, OP^*, DP^*, I^*, DT^*, LT^*, FA)$ of $O$ w.r.t. $D$ and $V$ exists such that $(CE)^C \neq \emptyset$.

Class Expression Subsumption: $CE_1$ is *subsumed* by a class expression $CE_2$ w.r.t. $O$ and $D$ if $(CE_1)^C \subseteq (CE_2)^C$ for each model $I = (\Delta_I, \Delta_D, C^*, OP^*, DP^*, I^*, DT^*, LT^*, FA)$ of $O$ w.r.t. $D$ and $V$.

Instance Checking: $a$ is an *instance* of $CE$ w.r.t. $O$ and $D$ if $(a)^I \in (CE)^C$ for each model $I = (\Delta_I, \Delta_D, C^*, OP^*, DP^*, I^*, DT^*, LT^*, FA)$ of $O$ w.r.t. $D$ and $V$.

Boolean Conjunctive Query Answering: $Q$ is an *answer* w.r.t. $O$ and $D$ if $Q$ is true in each model of $O$ w.r.t. $D$ and $V$ according to the standard definitions of first-order logic.

In order to ensure that ontology entailment, class expression satisfiability, class expression subsumption, and instance checking are decidable, the following restriction w.r.t. $O$ needs to be satisfied:

Each class expression of type *MinObjectCardinality*, *MaxObjectCardinality*, *ExactObjectCardinality*, and *ObjectHasSelf* that occurs in $O_1$, $CE_1$, and $CE_2$ can contain only object property expressions that are *simple* in the axiom closure $Ax$ of $O$.

For ontology equivalence to be decidable, $O_1$ needs to satisfy this restriction w.r.t. $O$ and vice versa. These restrictions are analogous to the first condition from Section 11.2 of the OWL 2 Specification [OWL 2 Specification].

3 Independence of the Direct Semantics from the Datatype Map in OWL 2 DL (Informative)

OWL 2 DL has been defined so that the consequences of an OWL 2 DL ontology $O$ do not depend on the choice of a datatype map, as long as the datatype map chosen contains all the datatypes occurring in $O$. This statement is made precise by the following theorem, and it has several useful consequences:

- One can apply the direct semantics to an OWL 2 DL ontology $O$ by considering only the datatypes explicitly occurring in $O$.
- When referring to various reasoning problems, the datatype map $D$ need not be given explicitly, as it is sufficient to consider an implicit datatype map containing only the datatypes from the given ontology.
- OWL 2 DL reasoners can provide datatypes not explicitly mentioned in this specification without fear that this will change the meaning of OWL 2 DL ontologies not using these datatypes.
Theorem DS1. Let $O_1$ and $O_2$ be OWL 2 DL ontologies over a vocabulary $V$ and $D = (\text{NDT}, \text{NSL}, \text{NFS}, \cdot_{\text{DT}}, \cdot_{\text{LS}}, \cdot_{\text{FS}})$ a datatype map such that each datatype mentioned in $O_1$ and $O_2$ is rdfs:Literal, a datatype defined in the respective ontology, or it occurs in NDT. Furthermore, let $D' = (\text{NDT}', \text{NSL}', \text{NFS}', \cdot_{\text{DT}'}, \cdot_{\text{LS}'}, \cdot_{\text{FS}'})$ be a datatype map such that $\text{NDT}' \subseteq \text{NDT}$, $\text{NSL}(\text{DT}) = \text{NSL}'(\text{DT})$, and $\text{NFS}(\text{DT}) = \text{NFS}'(\text{DT})$ for each $\text{DT} \in \text{NDT}$, and $\cdot_{\text{DT}'}, \cdot_{\text{LS}'}, \text{and } \cdot_{\text{FS}'}$ are extensions of $\cdot_{\text{DT}}, \cdot_{\text{LS}}$, and $\cdot_{\text{FS}}$, respectively. Then, $O_1$ entails $O_2 \text{ w.r.t. } D$ if and only if $O_1$ entails $O_2 \text{ w.r.t. } D'$. 

Proof. Without loss of generality, one can assume $O_1$ and $O_2$ to be in negation-normal form [Description Logics]. Furthermore, since datatype definitions in $O_1$ and $O_2$ are acyclic, one can assume that each defined datatype has been recursively replaced with its definition; thus, all datatypes in $O_1$ and $O_2$ are from NDT $\cup \{ \text{rdfs:Literal} \}$. The claim of the theorem is equivalent to the following statement: an interpretation $I \text{ w.r.t. } D$ and $V$ exists such that $O_1$ is and $O_2$ is not satisfied in $I$ if and only if an interpretation $I' \text{ w.r.t. } D'$ and $V$ exists such that $O_1$ is and $O_2$ is not satisfied in $I'$. The ($\Rightarrow$) direction is trivial since each interpretation $I'$ and $V$ is also an interpretation w.r.t. $D$ and $V$. For the ($\Leftarrow$) direction, assume that an interpretation $I = (\Delta I, \Delta D, \cdot_{\text{C}}, \cdot_{\text{OP}}, \cdot_{\text{DP}}, \cdot_{\text{LT}}, \cdot_{\text{FA}})$ w.r.t. $D$ and $V$ exists such that $O_1$ is and $O_2$ is not satisfied in $I$. Let $I' = (\Delta I, \Delta D', \cdot_{\text{C}}, \cdot_{\text{OP}}, \cdot_{\text{DP}}, \cdot_{\text{LT}}, \cdot_{\text{FA}})$ be an interpretation such that

- $\Delta D'$ is obtained by extending $\Delta D$ with the value space of all datatypes in NDT $\setminus \text{NDT}$,
- $\cdot_{\text{C}}$ coincides with $\cdot_{\text{C}}$ on all classes, and
- $\cdot_{\text{DP}'}$ coincides with $\cdot_{\text{DP}}$ on all data properties apart from owl:topDataProperty.

Clearly, DataComplementOf( DR $\text{DT}$ $\subseteq$ DataComplementOf( DR $\text{DT}'$ for each data range DR that is either a datatype, a datatype restriction, or an enumerated data range. The owl:topDataProperty property can occur in $O_1$ and $O_2$ only in tautologies. The interpretation of all other data properties is the same in $I$ and $I'$, so $(CE)^I = (CE)^{I'}$ for each class expression CE occurring in $O_1$ and $O_2$. Therefore, $O_1$ is and $O_2$ is not satisfied in $I'$. QED

4 Acknowledgments

The starting point for the development of OWL 2 was the OWL 1.1 member submission, itself a result of user and developer feedback, and in particular of information gathered during the OWL Experiences and Directions (OWLED) Workshop series. The working group also considered postponed issues from the WebOnt Working Group.

This document has been produced by the OWL Working Group (see below), and its contents reflect extensive discussions within the Working Group as a whole. The editors extend special thanks to Markus Krötzsch (FZI), Michael Schneider (FZI) and Thomas Schneider (University of Manchester) for their thorough reviews.
The regular attendees at meetings of the OWL Working Group at the time of publication of this document were: Jie Bao (RPI), Diego Calvanese (Free University of Bozen-Bolzano), Bernardo Cuenca Grau (Oxford University), Martin Dzbor (Open University), Achille Fokoue (IBM Corporation), Christine Golbreich (Université de Versailles St-Quentin and LIRMM), Sandro Hawke (W3C/MIT), Ivan Herman (W3C/ERCIM), Rinke Hoekstra (University of Amsterdam), Ian Horrocks (Oxford University), Elisa Kendall (Sandpiper Software), Markus Krötzsch (FZI), Carsten Lutz (Universität Bremen), Deborah L. McGuinness (RPI), Boris Motik (Oxford University), Jeff Pan (University of Aberdeen), Bijan Parsia (University of Manchester), Peter F. Patel-Schneider (Bell Labs Research, Alcatel-Lucent), Alan Ruttenberg (Science Commons), Uli Sattler (University of Manchester), Michael Schneider (FZI), Mike Smith (Clark & Parsia), Evan Wallace (NIST), and Zhe Wu (Oracle Corporation). We would also like to thank past members of the working group: Jeremy Carroll, Jim Hendler, Vipul Kashyap.

5 References

5.1 Normative References

[OWL 2 Specification]


5.2 Nonnormative References

[Description Logics]


[OWL 1 Semantics and Abstract Syntax]


[OWL 2 Profiles]

[SROIQ]

*The Even More Irresistible SROIQ.* Ian Horrocks, Oliver Kutz, and Uli Sattler.