W3C Recommendation

OWL 2 Web Ontology Language
Direct Semantics

W3C Recommendation 27 October 2009

This version:
http://www.w3.org/TR/2009/REC-owl2-direct-semantics-20091027/
Latest version (series 2):
http://www.w3.org/TR/owl2-direct-semantics/
Latest Recommendation:
http://www.w3.org/TR/owl-direct-semantics
Previous version:
http://www.w3.org/TR/2009/PR-owl2-direct-semantics-20090922/ (color-coded diff)

Editors:
Boris Motik, Oxford University Computing Laboratory
Peter F. Patel-Schneider, Bell Labs Research, Alcatel-Lucent
Bernardo Cuenca Grau, Oxford University Computing Laboratory
Contributors: (in alphabetical order)
Ian Horrocks, Oxford University Computing Laboratory
Bijan Parsia, University of Manchester
Uli Sattler, University of Manchester

Please refer to the **errata** for this document, which may include some normative corrections.

This document is also available in these non-normative formats: PDF version.

See also translations.

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May Be Superseded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/.

XML Schema Datatypes Dependency

OWL 2 is defined to use datatypes defined in the XML Schema Definition Language (XSD). As of this writing, the latest W3C Recommendation for XSD is version 1.0, with version 1.1 progressing toward Recommendation. OWL 2 has been designed to take advantage of the new datatypes and clearer explanations available in XSD 1.1, but for now those advantages are being partially put on hold. Specifically, until XSD 1.1 becomes a W3C Recommendation, the elements of OWL 2 which are based on it should be considered *optional*, as detailed in Conformance, section 2.3. Upon the publication of XSD 1.1 as a W3C Recommendation, those elements cease to be optional and are to be considered required as otherwise specified.

We suggest that for now developers and users follow the XSD 1.1 Candidate Recommendation. Based on discussions between the Schema and OWL Working Groups, we do not expect any implementation changes will be necessary as XSD 1.1 advances to Recommendation.

Document Unchanged

There have been no changes to the body of this document since the previous version. For details on earlier changes, see the change log.

Please Send Comments

Please send any comments to public-owl-comments@w3.org (public archive). Although work on this document by the OWL Working Group is complete, comments may be addressed in the errata or in future revisions. Open discussion among developers is welcome at public-owl-dev@w3.org (public archive).

Endorsed By W3C

This document has been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and is endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

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.

Table of Contents

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 2 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 $(DT)^{DT}$ 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 $(F , v)^{FS} \subseteq (DT)^{DT}$.

The set of datatypes N_{DT} of a datatype map D is not required to contain all datatypes from the OWL 2 datatype map; this allows one to talk about subsets of the OWL 2 datatype map, which may be necessary for the various profiles of OWL 2. If, however, D contains a datatype DT from the OWL 2 datatype map, then $N_{LS}(DT)$, $N_{FS}(DT)$, $(DT)^{DT}$, $(LV , DT)^{LS}$ for each $LV \in N_{LS}(DT)$, and $(F , v)^{FS}$ for each $(F , v) \in N_{FS}(DT)$ are required to coincide with the definitions for DT in the OWL 2 datatype map.

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, $N_{DT} \cup \{ \textit{rdfs:Literal} \} \subseteq V_{DT}$.
- V_{LT} is a set of *literals* LV^^DT for each datatype $DT \in N_{DT}$ and each lexical form $LV \in N_{LS}(DT)$.
- V_{FA} is the set of pairs (F , lt) for each constraining facet F, datatype $DT \in N_{DT}$, and literal $lt \in V_{LT}$ such that $(F , (LV , DT_1)^{LS}) \in N_{FS}(DT)$, 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:

- Δ_I is a nonempty set called the *object domain*.
- Δ_D is a nonempty set disjoint with Δ_I called the *data domain* such that $(DT)^{DT} \subseteq \Delta_D$ for each datatype $DT \in V_{DT}$.
- $\cdot^{C}$ is the *class interpretation function* that assigns to each class $C \in V_C$ a subset $(C)^{C} \subseteq \Delta_I$ such that
    - $(owl{:}Thing)^{C} = \Delta_I$ and
    - $(owl{:}Nothing)^{C} = \emptyset$.
- $\cdot^{OP}$ 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
    - $(owl{:}topObjectProperty)^{OP} = \Delta_I \times \Delta_I$ and
    - $(owl{:}bottomObjectProperty)^{OP} = \emptyset$.
- $\cdot^{DP}$ 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
    - $(owl{:}topDataProperty)^{DP} = \Delta_I \times \Delta_D$ and
    - $(owl{:}bottomDataProperty)^{DP} = \emptyset$.
- $\cdot^{I}$ 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
    - $\cdot^{DT}$ is the same as in D for each datatype $DT \in N_{DT}$, and
    - $(rdfs{:}Literal)^{DT} = \Delta_D$.
- $\cdot^{LT}$ 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

Object Property Expression	Interpretation $\cdot^{OP}$
`ObjectInverseOf( OP )`	$\{ (x , y) \mid (y , x) \in (OP)^{OP} \}$

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 Δ_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 Δ_D — that is, as a set $(DT)^{DT} \subseteq (\Delta_D)^n$

Table 3. Interpreting Data Ranges

Data Range	Interpretation $\cdot^{DT}$
DataIntersectionOf(DR_1 ... DR_n)	$(DR_1)^{DT} \cap \ldots \cap (DR_n)^{DT}$
DataUnionOf(DR_1 ... DR_n)	$(DR_1)^{DT} \cup \ldots \cup (DR_n)^{DT}$
`DataComplementOf( DR )`	$(\Delta_D)^n \setminus (DR)^{DT}$ where *n* is the arity of *DR*
DataOneOf(lt_1 ... lt_n)	$\{ (lt_1)^{LT} , \ldots , (lt_n)^{LT} \}$
DatatypeRestriction(DT F_1 lt_1 ... F_n lt_n)	$(DT)^{DT} \cap (F_1 , lt_1)^{FA} \cap \ldots \cap (F_n , lt_n)^{FA}$

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 4. Interpreting Class Expressions

Class Expression	Interpretation $\cdot^{C}$

ObjectIntersectionOf(CE$_1$ ... CE$_n$)	$(CE_1)^C \cap \ldots \cap (CE_n)^C$
ObjectUnionOf(CE$_1$ ... CE$_n$)	$(CE_1)^C \cup \ldots \cup (CE_n)^C$
ObjectComplementOf(CE)	$\Delta_I \setminus (CE)^C$
ObjectOneOf(a$_1$ ... a$_n$)	$\{ (a_1)^I , \ldots , (a_n)^I \}$
ObjectSomeValuesFrom(OPE CE)	$\{ x \mid \exists\, y : (x, y) \in (OPE)^{OP} \text{ and } y \in (CE)^C \}$
ObjectAllValuesFrom(OPE CE)	$\{ x \mid \forall\, y : (x, y) \in (OPE)^{OP} \text{ implies } y \in (CE)^C \}$
ObjectHasValue(OPE a)	$\{ x \mid (x , (a)^I) \in (OPE)^{OP} \}$
ObjectHasSelf(OPE)	$\{ x \mid (x , x) \in (OPE)^{OP} \}$
ObjectMinCardinality(n OPE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \} \geq \mathrm{n} \}$
ObjectMaxCardinality(n OPE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \} \leq \mathrm{n} \}$
ObjectExactCardinality(n OPE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \} = \mathrm{n} \}$
ObjectMinCardinality(n OPE CE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \text{ and } y \in (CE)^C \} \geq \mathrm{n} \}$
ObjectMaxCardinality(n OPE CE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \text{ and } y \in (CE)^C \} \leq \mathrm{n} \}$
ObjectExactCardinality(n OPE CE)	$\{ x \mid \#\{ y \mid (x , y) \in (OPE)^{OP} \text{ and } y \in (CE)^C \} = \mathrm{n} \}$
DataSomeValuesFrom(DPE$_1$ ... DPE$_n$ DR)	$\{ x \mid \exists\, y_1, \ldots , y_n : (x , y_k) \in (DPE_k)^{DP} \text{ for each } 1 \leq k \leq n \text{ and } (y_1 , \ldots , y_n) \in (DR)^{DT} \}$
DataAllValuesFrom(DPE$_1$ ... DPE$_n$ DR)	$\{ x \mid \forall\, y_1, \ldots , y_n : (x , y_k) \in (DPE_k)^{DP} \text{ for each } 1 \leq k \leq n \text{ imply } (y_1 , \ldots , y_n) \in (DR)^{DT} \}$
DataHasValue(DPE lt)	$\{ x \mid (x , (lt)^{LT}) \in (DPE)^{DP} \}$
DataMinCardinality(n DPE)	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP} \} \geq \mathrm{n} \}$

`DataMaxCardinality( n DPE )`	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP} \} \leq n \}$
`DataExactCardinality( n DPE )`	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP} \} = n \}$
`DataMinCardinality( n DPE DR )`	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP}$ and $y \in (DR)^{DT} \} \geq n \}$
`DataMaxCardinality( n DPE DR )`	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP}$ and $y \in (DR)^{DT} \} \leq n \}$
`DataExactCardinality( n DPE DR )`	$\{ x \mid \#\{ y \mid (x , y) \in (DPE)^{DP}$ and $y \in (DR)^{DT} \} = n \}$

2.3 Satisfaction in an Interpretation

An interpretation $I = (\Delta_I , \Delta_D , \cdot^{C} , \cdot^{OP} , \cdot^{DP} , \cdot^{I} , \cdot^{DT} , \cdot^{LT} , \cdot^{FA})$ *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:

> $ISNAMED_O(x) = true$ for $x \in \Delta_I$ if and only if $(a)^I = x$ 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 5. Satisfaction of Class Expression Axioms in an Interpretation

Axiom	Condition
SubClassOf(CE_1 CE_2)	$(CE_1)^C \subseteq (CE_2)^C$
EquivalentClasses(CE_1 ... CE_n)	$(CE_j)^C = (CE_k)^C$ for each $1 \leq j \leq n$ and each $1 \leq k \leq n$
DisjointClasses(CE_1 ... CE_n)	$(CE_j)^C \cap (CE_k)^C = \emptyset$ for each $1 \leq j \leq n$ and each $1 \leq k \leq n$ such that $j \neq k$
DisjointUnion(C CE_1 ... CE_n)	$(C)^C = (CE_1)^C \cup \ldots \cup (CE_n)^C$ and $(CE_j)^C \cap (CE_k)^C = \emptyset$ for each $1 \leq j \leq n$ and each $1 \leq k \leq n$ such that $j \neq k$

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

Axiom	Condition
SubObjectPropertyOf(OPE$_1$ OPE$_2$)	$(OPE_1)^{OP} \subseteq (OPE_2)^{OP}$
SubObjectPropertyOf(ObjectPropertyChain(OPE$_1$ ... OPE$_n$) OPE)	$\forall\, y_0 , \ldots , y_n : (y_0 , y_1) \in (OPE_1)^{OP}$ and ... and $(y_{n-1} , y_n) \in (OPE_n)^{OP}$ imply $(y_0 , y_n) \in (OPE)^{OP}$
EquivalentObjectProperties(OPE$_1$ ... OPE$_n$)	$(OPE_j)^{OP} = (OPE_k)^{OP}$ for each $1 \le j \le n$ and each $1 \le k \le n$
DisjointObjectProperties(OPE$_1$ ... OPE$_n$)	$(OPE_j)^{OP} \cap (OPE_k)^{OP} = \emptyset$ for each $1 \le j \le n$ and each $1 \le k \le n$ such that $j \ne k$
ObjectPropertyDomain(OPE CE)	$\forall\, x , y : (x , y) \in (OPE)^{OP}$ implies $x \in (CE)^C$
ObjectPropertyRange(OPE CE)	$\forall\, x , y : (x , y) \in (OPE)^{OP}$ implies $y \in (CE)^C$
InverseObjectProperties(OPE$_1$ OPE$_2$)	$(OPE_1)^{OP} = \{ (x , y) \mid (y , x) \in (OPE_2)^{OP} \}$
FunctionalObjectProperty(OPE)	$\forall\, x , y_1 , y_2 : (x , y_1) \in (OPE)^{OP}$ and $(x , y_2) \in (OPE)^{OP}$ imply $y_1 = y_2$
InverseFunctionalObjectProperty(OPE)	$\forall\, x_1 , x_2 , y : (x_1 , y) \in (OPE)^{OP}$ and $(x_2 , y) \in (OPE)^{OP}$ imply $x_1 = x_2$
ReflexiveObjectProperty(OPE)	$\forall\, x : x \in \Delta_I$ implies $(x , x) \in (OPE)^{OP}$
IrreflexiveObjectProperty(OPE)	$\forall\, x : x \in \Delta_I$ implies $(x , x) \notin (OPE)^{OP}$
SymmetricObjectProperty(OPE)	$\forall\, x , y : (x , y) \in (OPE)^{OP}$ implies $(y , x) \in (OPE)^{OP}$

`AsymmetricObjectProperty( OPE )`	$\forall x , y : (x , y) \in (OPE)^{OP}$ implies $(y , x) \notin (OPE)^{OP}$
`TransitiveObjectProperty( OPE )`	$\forall x , y , z : (x , y) \in (OPE)^{OP}$ and $(y , z) \in (OPE)^{OP}$ imply $(x , z) \in (OPE)^{OP}$

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

Axiom	Condition
`SubDataPropertyOf( DPE`$_1$ `DPE`$_2$ `)`	$(DPE_1)^{DP} \subseteq (DPE_2)^{DP}$
`EquivalentDataProperties( DPE`$_1$ `... DPE`$_n$ `)`	$(DPE_j)^{DP} = (DPE_k)^{DP}$ for each $1 \le j \le n$ and each $1 \le k \le n$
`DisjointDataProperties( DPE`$_1$ `... DPE`$_n$ `)`	$(DPE_j)^{DP} \cap (DPE_k)^{DP} = \emptyset$ for each $1 \le j \le n$ and each $1 \le k \le n$ such that $j \ne k$
`DataPropertyDomain( DPE CE )`	$\forall x , y : (x , y) \in (DPE)^{DP}$ implies $x \in (CE)^{C}$
`DataPropertyRange( DPE DR )`	$\forall x , y : (x , y) \in (DPE)^{DP}$ implies $y \in (DR)^{DT}$
`FunctionalDataProperty( DPE )`	$\forall x , y_1 , y_2 : (x , y_1) \in (DPE)^{DP}$ and $(x , y_2) \in (DPE)^{DP}$ imply $y_1 = y_2$

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

Axiom	Condition
`DatatypeDefinition( DT DR )`	$(DT)^{DT} = (DR)^{DT}$

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

Axiom	Condition
HasKey(CE (OPE$_1$ ... OPE$_m$) (DPE$_1$ ... DPE$_n$))	$\forall\ x , y , z_1 , ... , z_m , w_1 , ... , w_n$: if $x \in (CE)^C$ and $ISNAMED_O(x)$ and $y \in (CE)^C$ and $ISNAMED_O(y)$ and $(x , z_i) \in (OPE_i)^{OP}$ and $(y , z_i) \in (OPE_i)^{OP}$ and $ISNAMED_O(z_i)$ for each $1 \le i \le m$ and $(x , w_j) \in (DPE_j)^{DP}$ and $(y , w_j) \in (DPE_j)^{DP}$ for each $1 \le j \le n$ then x = y

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

Axiom	Condition
SameIndividual(a$_1$ ... a$_n$)	$(a_j)^I = (a_k)^I$ for each $1 \le j \le n$ and each $1 \le k \le n$
DifferentIndividuals(a$_1$ ... a$_n$)	$(a_j)^I \neq (a_k)^I$ for each $1 \le j \le n$ and each $1 \le k \le n$ such that $j \neq k$
ClassAssertion(CE a)	$(a)^I \in (CE)^C$
ObjectPropertyAssertion(OPE a$_1$ a$_2$)	$((a_1)^I , (a_2)^I) \in (OPE)^{OP}$
NegativeObjectPropertyAssertion(OPE a$_1$ a$_2$)	$((a_1)^I , (a_2)^I) \notin (OPE)^{OP}$
DataPropertyAssertion(DPE a lt)	$((a)^I , (lt)^{LT}) \in (DPE)^{DP}$
NegativeDataPropertyAssertion(DPE a lt)	$((a)^I , (lt)^{LT}) \notin (DPE)^{DP}$

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 $\cdot^{J}$ coincides with $\cdot^{I}$ 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 $I = (\Delta_I , \Delta_D , \cdot^{C} , \cdot^{OP} , \cdot^{DP} , \cdot^{I} , \cdot^{DT} , \cdot^{LT} , \cdot^{FA})$ 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 $I = (\Delta_I , \Delta_D , \cdot^{C} , \cdot^{OP} , \cdot^{DP} , \cdot^{I} , \cdot^{DT} , \cdot^{LT} , \cdot^{FA})$ 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 $I = (\Delta_I , \Delta_D , \cdot^{C} , \cdot^{OP} , \cdot^{DP} , \cdot^{I} , \cdot^{DT} , \cdot^{LT} , \cdot^{FA})$ 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, 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 $D = (N_{DT} , N_{LS} , N_{FS} , \cdot^{DT} , \cdot^{LS} , \cdot^{FS})$ 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 N_{DT}. Furthermore, let $D' = (N_{DT}' , N_{LS}' , N_{FS}' , \cdot^{DT\,'} , \cdot^{LS\,'} , \cdot^{FS\,'})$ be a datatype map such that $N_{DT} \subseteq N_{DT}'$, $N_{LS}(DT) = N_{LS}'(DT)$, and $N_{FS}(DT) = N_{FS}'(DT)$ for each $DT \in N_{DT}$, and $\cdot^{DT\,'}$, $\cdot^{LS\,'}$, and $\cdot^{FS\,'}$ are extensions of $\cdot^{DT}$, $\cdot^{LS}$, and $\cdot^{FS}$, respectively. Then, O_1 entails O_2 w.r.t. D if and only if O_1 entails O_2 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 $N_{DT} \cup \{$ *rdfs:Literal* $\}$. The claim of the theorem is equivalent to the following statement: an interpretation I 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' w.r.t. D' and V exists such that O_1 is and O_2 is not satisfied in I'. The (⇐) direction is trivial since each interpretation I w.r.t. D' and V is also an interpretation w.r.t. D and V. For the (⇒) direction, assume that an interpretation $I = (\Delta_I , \Delta_D , \cdot^{C} , \cdot^{OP} , \cdot^{DP} , \cdot^{I} , \cdot^{DT} , \cdot^{LT} , \cdot^{FA})$ w.r.t. D and V exists such that O_1 is and O_2 is not satisfied in I. Let $I' = (\Delta_I , \Delta_D' , \cdot^{C\,'} , \cdot^{OP} , \cdot^{DP\,'} , \cdot^{I} , \cdot^{DT\,'} , \cdot^{LT\,'} , \cdot^{FA\,'})$ be an interpretation such that

- Δ_D' is obtained by extending Δ_D with the value space of all datatypes in $N_{DT}' \setminus N_{DT}$,
- $\cdot^{C\,'}$ coincides with $\cdot^{C}$ on all classes, and
- $\cdot^{DP\,'}$ coincides with $\cdot^{DP}$ on all data properties apart from *owl:topDataProperty*.

Clearly, `DataComplementOf( DR )`$^{DT} \subseteq$ `DataComplementOf( DR )`$^{DT\,'}$ 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 $(CE)^{C} = (CE)^{C\,'}$ 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 Appendix: Change Log (Informative)

4.1 Changes Since Proposed Recommendation

No changes have been made to this document since the Proposed Recommendation of 22 September, 2009.

4.2 Changes Since Candidate Recommendation

This section summarizes the changes to this document since the Candidate Recommendation of 11 June, 2009.

- An editorial comment was added to clarify the role played by the OWL 2 datatype map.

4.3 Changes Since Last Call

This section summarizes the changes to this document since the Last Call Working Draft of 21 April, 2009.

- Some minor editorial changes were made.

5 Acknowledgments

The starting point for the development of OWL 2 was the OWL1.1 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 Computing Laboratory), 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 Computing Laboratory), Elisa Kendall (Sandpiper Software), Markus Krötzsch (FZI), Carsten Lutz (Universität Bremen), Deborah L. McGuinness (RPI), Boris Motik (Oxford University Computing Laboratory), Jeff Pan (University of Aberdeen), Bijan Parsia (University of Manchester), Peter F. Patel-Schneider (Bell Labs Research, Alcatel-Lucent), Sebastian Rudolph (FZI), Alan Ruttenberg (Science Commons), Uli Sattler (University of Manchester), Michael Schneider (FZI), Mike Smith (Clark & Parsia), Evan Wallace (NIST), Zhe Wu (Oracle Corporation), and Antoine Zimmermann (DERI Galway). We would also like to thank past members of the working group: Jeremy Carroll, Jim Hendler, Vipul Kashyap.

6 References

6.1 Normative References

[OWL 2 Specification]
OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax Boris Motik, Peter F. Patel-Schneider, Bijan Parsia, eds. W3C Recommendation, 27 October 2009, http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/. Latest version available at http://www.w3.org/TR/owl2-syntax/.

6.2 Nonnormative References

[Description Logics]
The Description Logic Handbook: Theory, Implementation, and Applications, second edition. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, eds. Cambridge University Press, 2007. Also see the *Description Logics Home Page*.

[OWL 1 Semantics and Abstract Syntax]
OWL Web Ontology Language: Semantics and Abstract Syntax. Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks, eds. W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-owl-semantics-20040210/. Latest version available at http://www.w3.org/TR/owl-semantics/.

[OWL 2 Profiles]
OWL 2 Web Ontology Language: Profiles Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, Carsten Lutz, eds. W3C Recommendation, 27 October 2009, http://www.w3.org/TR/2009/REC-owl2-profiles-20091027/. Latest version available at http://www.w3.org/TR/owl2-profiles/.

[SROIQ]
The Even More Irresistible SROIQ. Ian Horrocks, Oliver Kutz, and Uli Sattler. In Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006). AAAI Press, 2006.