

OWLED Task Force Report

Authors: Christine Golbreich, xxx
Working Draft October 2007

Part I: New features of OWL 1.1: examples and motivating Use Cases

Table of Contents

1. Introduction	3
2. Language New Features	3
3. Syntactic Sugar	3
3.1. DisjointUnion	3
Examples	4
Use Cases and Requirements from literature	4
3.2. DisjointClasses.....	5
Examples	5
Use Cases and Requirements from literature	5
3.3. NegativeObjectPropertyAssertion NegativeDataPropertyAssertion	5
Examples	6
Use Cases and Requirements from literature	6
4. New restrictions on properties	6
4.1. Local Reflexivity Restrictions	6
Examples	7
Use Cases and Requirements from literature	7
4.2. Qualified Cardinality Restrictions	7
Examples	7
Motivations from literature	8
5. New Characteristics of Properties.....	8
5.1. Reflexive	8
Examples	8
5.2. Irreflexive	9
Examples	9
5.3. Asymmetric	9
Examples	9
Use Cases and Requirements from literature	10
6. New Relationships between Properties	10
6.1. Disjoint properties.....	10
Examples	10
Use Cases and Requirements from literature	11
6.2. Property chain inclusion axiom	11
Examples	11
Use Cases and Requirements from literature	11
7. New Datatype Mechanisms	12
7.1. OWL mechanisms for user-defined datatypes.....	12
Examples	12
7.2. n-ary datatypes.....	13
8. Punning	14

Examples	14
Use Cases and Requirements from literature	14
9. Annotations	14
9.1. Annotation of entities	14
Examples	14
Use Cases and Requirements from literature	15
9.2. Annotation of axioms	15
10. Rules	15
10.1.1. SWRL rules	15
10.1.2. DL-safe Horn rule extension	15
10.1.3. Logic Programming rules	15
10.1.4. Other	15
11. Recapitulative Table	16

Part II: New extensions: Requirements and motivating Use Cases

1. Introduction

The first part of this report attempts at giving simple examples of the new features of OWL 1.1 and links to Use Cases from OWLED or other literature motivating these extensions. The presentation aims at complying with the [OWL 1.1 Web Ontology Language Overview](http://www.w3.org/Submission/2006/SUBM-owl11-owl_specification-20061219/) and at using the definitions of http://www.w3.org/Submission/2006/SUBM-owl11-owl_specification-20061219/, as much as possible.

For each new feature, (1) its meaning is informally defined, (2) illustrated by simple examples, and (3) followed by motivating use cases from various domains with their online reference for readers interested in more details. The intended goal is to present the new notions introduced in such a way that they are more easily understandable and usable by OWL users/customers who are considering using OWL 1.1 to author OWL content in future projects.

2. Language New Features

There are four main categories of new features added in OWL 1.1:

1. Syntactic sugar, shortands to make some common statements easier to say;
2. New Description Logic constructs to increase expressivity for properties;
3. New Datatype mechanism to expand datatype expressiveness;
4. Metamodeling constructs to express metalogical information about the elements of an ontology

3. Syntactic Sugar

OWL 1.1 provides the shortands ***DisjointUnion*** and ***DisjointClasses*** for Class Axioms.

3.1. DisjointUnion

A ***DisjointUnion*** axiom defines a class as the union of other classes, all of which are pair-wise disjoint.

E.g; `DisjointUnion(State Solid Liquid Gas)`
All the material on earth is in one of the three states: solid, liquid, or gas

Syntax: `DisjointUnion({ annotation } owlClassURI description description { description })`

DisjointUnion is a shorthand for `owl:disjointWith` statements used in combination with [owl:unionOf](#) to define a set of mutually disjoint and complete subclasses of a superclass.

Examples

Note: each example is numbered, e.g; #1, and related use cases in the literature denoted with brackets, e.g; [UC1], are given when available.

- **Life Sciences**

#1. *A brain hemisphere is either a right or a left hemisphere, and not both at the same time* [UC1 Ex.2].

Class Hemisphere is the DisjointUnion of the classes LeftHemisphere and RightHemisphere:
DisjointUnion(Hemisphere LeftHemisphere RightHemisphere)

#2. *A lobe is one of the following types: frontal, parietal, temporal, occipital, limbic. Lobe FrontalLobe ParietalLobe TemporalLobe OccipitalLobe LimbicLobe* [UC1 Ex.3].

The class Lobe is the DisjointUnion of the classes FrontalLobe ParietalLobe TemporalLobe OccipitalLobe LimbicLobe:
DisjointUnion(Lobe FrontalLobe ParietalLobe TemporalLobe OccipitalLobe LimbicLobe)

#3. *A Cell is either a nucleated cell or a non-nucleated cell, and it is clear that a cell cannot be at the same time nucleated and non-nucleated.* [UC2].

Class Cell is the DisjointUnion of the classes NucleatedCell and NonNucleatedCell

#4. *An Amine Group is exclusively either a Primary Amine Group, a Secondary Amine Group or a Tertiary Amine Group* [UC3].

The class AmineGroup is the DisjointUnion of the classes PrimaryAmineGroup SecondaryAmineGroup and TertiaryAmineGroup

- **Automotive industry**

#5. *A car can have only front doors, rear doors and a trunk door* [UC4]

Class CarDoor is the DisjointUnion of the classes FrontDoor, RearDoor and TrunkDoor

Use Cases and Requirements from literature

There are many use cases in literature motivating this feature. Here some representative examples among others.

- **Life Sciences**

- [UC1] Web ontology language requirements w.r.t expressiveness of taxonomy and axioms in medicine (examples # 1, 2)
 - <http://www.med.univ-rennes1.fr/lim/doc'61.pdf>
- [UC2] The Foundational Model of anatomy (FMA) explicitly distinguishes among physical entities between those that have or do not have mass and treats anatomical spaces, surfaces, lines and points as universals or classes. Material_physical_anatomical_entity is the disjointUnion of Physical_anatomical_entity and Non_Physical_anatomical_entity
 - <http://sig.biostr.washington.edu/projects/fm/omEditPublications.html>
 - <http://www.med.univ-rennes1.fr/lim/doc'128.pdf>
 - <http://www.ea3888.univ-rennes1.fr/dameron/publis/2007protege-dameron.pdf> (#3)
- [UC3] Describing chemical functional groups (#4)
 - <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper28.pdf>

- **Automotive industry**

- [UC4] An exploratory study in an automotive company (#5)

- <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper08.pdf>

3.2. DisjointClasses

A **DisjointClasses** axiom takes a set of classes and states that all classes from the set are pair-wise disjoint.

E.g; DisjointClasses(Cat Dog Bird)

It is not possible to be at the same time a Cat and a Dog, a Cat and a Bird, or a Dog and a Bird

Syntax: DisjointClasses({ *annotation* } *description description* { *description* })

DisjointClasses is used as a shorthand for several owl:disjointWith statements.

Examples

- Life Sciences

#6. *Upper, Middle, Lower lobe of lung* are exclusive [UC2].

Classes UpperLobeOfLung, MiddleLobeOfLung LowerLobeOfLung are disjoint:

DisjointClasses(UpperLobeOfLung, MiddleLobeOfLung LowerLobeOfLung)

#7. *A number of other classes of FMA are shown in* <http://www.ea3888.univ-rennes1.fr/dameron/publis/2007protege-dameron.pdf>

Left X/Right X: 3736 classes (e.g. Left lung / Right lung)

X left Y/X right Y: 13989 classes (e.g. Skin of right breast / Skin of right breast)

Male X/Female X: 25 classes (e.g. Male breast / Female breast)

X male Y/X female Y: 75 classes (e.g. Right side of male chest / Right side of female chest)

Enumeration: (e.g. First cervical nerve)

Use Cases and Requirements from literature

- Life Sciences
 - [UC2] The Foundational Model of anatomy (FMA): in the original FMA, sibling classes are most often (but not always) assumed to be disjoint. Left breast and Right breast, as well as Male breast and Female breast are disjoint (but Left breast and Male breast are not).
 - <http://www.med.univ-rennes1.fr/lim/doc`128.pdf> (#6)
 - <http://www.ea3888.univ-rennes1.fr/dameron/publis/2007protege-dameron.pdf> (#6)

3.3. NegativeObjectPropertyAssertion NegativeDataPropertyAssertion

OWL 1.1 provides the shorthands **NegativeObjectPropertyAssertion** and **NegativeDataPropertyAssertion** for Object Property and Data Property Axioms.

A **NegativeObjectPropertyAssertion** axiom states negative facts about individuals.

While an **ObjectPropertyAssertion** states that a given property holds for the given individuals, **NegativeObjectPropertyAssertion** states that a given property does not hold for the given individuals.

E.g; NegativeObjectPropertyAssertion(Sarkozy presidentOf USA)
Sarkozy is not president of USA

Syntax: NegativeObjectPropertyAssertion({ annotation }objectPropertyExpression
sourceIndividualURI targetValue)

A **NegativeDataPropertyAssertion** axiom states negative facts about individuals.

E.g; NegativeDataPropertyAssertion (old Sarkozy 20)
Sarkozy is not 20 [years] old

Syntax: NegativeDataPropertyAssertion({ annotation } dataPropertyExpression
sourceIndividualURI targetValue)

Examples

Use Cases and Requirements from literature

4. New restrictions on properties

OWL 1.1 classes can be defined using propositional connectives, restrictions on object or data properties, restrictions on cardinalities of object or data properties (cf. http://www.w3.org/Submission/2006/SUBM-owl11-owl_specification-20061219/#5). OWL1.1 provides new constructs for restrictions on object or data properties, and for restrictions on cardinalities of object or data properties

4.1. Local Reflexivity Restrictions

OWL 1.1 allows to assert restrictions on object or data properties by means of the new construct **ObjectExistsSelf**.

The class defined by an **ObjectExistsSelf** restriction on an object property denotes the set of objects that are connected to themselves via the given object property

E.g; ObjectExistsSelf(cite)
individuals who cite themselves (e.g., author of auto-citations)

Syntax: ObjectExistsSelf(*objectPropertyExpression*)

Examples

Use Cases and Requirements from literature

4.2. Qualified Cardinality Restrictions

OWL 1.1 allows to assert qualified cardinality restrictions on object or data properties by means of the new constructs ***ObjectMinCardinality*** ***ObjectMaxCardinality*** ***ObjectExactCardinality*** ***DataMinCardinality*** ***DataMaxCardinality*** ***DataExactCardinality***.

The class ***objectMinCardinality*** denotes the set of objects that are connected via the given object property to at least the given number of instances of the given class, the class ***objectMaxCardinality*** denotes the set of objects that are connected via the given object property to at most the given number of instances of the given class, and the class ***objectExactCardinality*** denotes the set of objects that are connected via the given object property to exactly the given number of instances of the given class.

A qualified cardinality restriction defines a restriction on the number of instances of the property *and* on their class or data type. The addition that 'Qualified' Cardinality Restrictions brings to OWL initial constructs for cardinality restrictions is to enable to define restrictions not only on the number of instances of the property but also on the class or data type of the instances.

Note: in OWL1.1 qualified or unqualified cardinality restrictions are both allowed (an unqualified cardinality restriction is equivalent to a qualified one where the restricting class is `owl:Thing`).

Examples

Here a few examples among many.

- **Life Sciences**

#8. Restriction stating that a Brain Hemisphere has exactly one direct part of type frontal, parietal, temporal, occipital, limbic lobe [UC1(Ex.14)]

Class of individuals having exactly one part of type frontal lobe:
`ObjectExactCardinality(1 part FrontalLobe)`

Class of individuals having exactly one part of type parietal lobe:
`ObjectExactCardinality(1 part ParietalLobe)`

#9. Restriction stating that a Secondary Amine Group has bond with exactly one hydrogen atom [UC3]

Class of individuals having bond exactly with two hydrogen atoms:
`ObjectExactCardinality(2 hasBond HydrogenAtom)`

- **Automotive industry**

#10. Restriction for stating that a car has atmost 5 doors [UC4]

Class of individuals having atmost 5 doors as part
ObjectMaxCardinality(5 part Door)

#11. Restrictions for stating that a five-door car has exactly three doors, two front doors two rear doors, plus one trunk

ObjectExactCardinality(5 part Door);
ObjectExactCardinality(2 part FrontDoor);
ObjectExactCardinality(2 part RearDoor);
ObjectExactCardinality(1 Trunk Door);

Motivations from literature

- In Life Sciences
 - [UC1] Web ontology language requirements w.r.t expressiveness of taxonomy and axioms in medicine (#6)
 - http://www.med.univ-rennes1.fr/lim/doc_61.pdf
 - http://www.med.univ-rennes1.fr/lim/doc_114.pdf
 - [UC3] Describing chemical functional groups (#7)
 - <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper28.pdf>
- In Automotive industry
 - [UC4] An exploratory study in an automotive company (#8, #9)
 - <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper08.pdf>

5. New Characteristics of Properties

5.1. Reflexive

A ReflexiveObjectProperty axiom asserts that a given property is reflexive, that is the property holds for all the individuals.

E.g; ReflexiveObjectProperty (sameParent)
Each individual as the same parents as himself

Syntax: ReflexiveObjectProperty({ **annotation** } **objectPropertyExpression**)

Examples

- Life Sciences

#12. The property sameBoundary is reflexive [UC5]

ReflexiveObjectProperty(sameBoundary);

5.2. Irreflexive

An **IrreflexiveObjectProperty** axiom asserts that a given property is irreflexive, that is the property does not hold for any individual

E.g; IrreflexiveObjectProperty (father)
No individual is his own father

Syntax: IrreflexiveObjectProperty ({ **annotation** } **objectPropertyExpression**)

Examples

- **Life Sciences**

#13. Mereological properties like anatomicalPartOf or topological properties like connectedTo or boundedBy are irreflexive [UC5]

anatomicalPartOf is an IrreflexiveObjectProperty, connectedTo is an IrreflexiveObjectProperty, boundedBy is an IrreflexiveObjectProperty:
IrreflexiveObjectProperty(anatomicalPartOf);
IrreflexiveObjectProperty(connectedTo);
IrreflexiveObjectProperty(boundedBy);

- **Earth and Space**

#14. Topological and spatial relationships like flowsInto are irreflexive [UC7]

flowsInto is irreflexive as anyone river cannot flow into the same river
IrreflexiveObjectProperty(flowsInto);

5.3. Asymmetric

An **asymmetricObjectProperty** axiom asserts that a given property is asymmetric, that is when the property hold between individuals a and b then it does not hold between b and a

E.g; AsymmetricObjectProperty (father)
If an individual is the father of someone, then the latter is not his father

Syntax: AsymmetricObjectProperty({ **annotation** } **objectPropertyExpression**)

Examples

- **Life Sciences**

#15. Many mereological properties e.g; anatomicalPartOf or topological properties e.g; connectedTo are asymmetric: when a is a part of b then b is not a part of a [UC5]

anatomicalPartOf is an AsymmetricObjectProperty, connectedTo is an AsymmetricObjectProperty:
AsymmetricObjectProperty(anatomicalPartOf);
AsymmetricObjectProperty(connectedTo);

Use Cases and Requirements from literature

- Life Sciences
 - [UC5] Use Case: Ontology with Rules for identifying brain anatomical structures annexe (#12, 13, 15). The annexe presents several properties requiring the definition of reflexive, irreflexive, asymmetric or antisymmetric features, represented here with rules as OWL1.1 did not yet exist.
 - annexe <http://www.med.univ-rennes1.fr/~cgolb/Brain/annexes.pdf>
<http://www.w3.org/2004/12/rules-ws/paper/64/>
 - [UC6] Use Case : OBO relations ontology aims at defining a set of basic relations with their semantics and propose several characteristics for these relations.
 - <http://obofoundry.org/ro/#summary>
 - [UC6B] Use Case: Mapping OBO to OWL1.1. “OBO to OWL: Go to OWL1.1!” “OBO and OWL: Leveraging Semantic Web Technologies for the Life Sciences (ISWC 2007)” gives examples of OWL1.1 new Characteristics of Properties required for mapping OBO to OWL.
OBO defines several typedef such as is_reflexive , is_symmetric, is_cyclic, is_anti_symmetric, etc. Some of them are translated using the new OWL1.1 features, other ones are transformed into annotations, for details see:
 - <http://www.cs.man.ac.uk/~horrocks/obo/>
 - http://www.med.univ-rennes1.fr/lim/doc_178.pdf
 - http://www.med.univ-rennes1.fr/lim/doc_175.pdf
- Geography
 - [UC7] Use Case: Experiences of using OWL at the Ordnance Survey: “Here [for The topological and spatial relationships], and in many other places, we need to be able to say whether a property is reflexive, irreflexive, asymmetric or antisymmetric in order to capture the true intentions of our axioms. (#14)
 - <http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-188/sub17.pdf>

6. New Relationships between Properties

6.1. Disjoint properties

A **disjointObjectProperties** axiom takes a set of object properties and states that all properties from the set are pair-wise disjoint.

E.g; DisjointObjectProperty(father mother)
It is not possible to be at the same time the father and the mother of an individual

Syntax: DisjointDataProperties({ *annotation* } *dataPropertyExpression*
dataPropertyExpression { *dataPropertyExpression* })

Examples

- Life Sciences

#16. *Properties connectedTo and contiguousTo are exclusive, that is do not hold at the same time*

Properties *connectedTo* and *contiguousTo* are disjoint [UC5].
DisjointObjectProperty (connectedTo contiguousTo).

Use Cases and Requirements from literature

- In Life Sciences
 - [UC5] Use Case: Ontology with Rules for identifying brain anatomical structures annexe (#6)
 - <http://www.w3.org/2004/12/rules-ws/paper/64/>

6.2. Property chain inclusion axiom

A **SubObjectPropertyChain** axiom asserts a chaining relationship between object properties

E.g; SubObjectPropertyOf(SubObjectPropertyChain(father brother) uncle)
the father of the brother of someone is his uncle

Syntax: SubObjectPropertyChain(*objectPropertyExpression*
objectPropertyExpression { *objectPropertyExpression* })

SubObjectPropertyChain axiom used together with SubObjectPropertyOf provide a means to represent some types of rules (while remaining decidable under special restrictions, roughly provided that there are no cyclic inclusions, see http://www.webont.org/owl/1.1/owl_specification.html#7)

Examples

- **Life Sciences**

#17. *The transfer of properties from parts to whole has often been required, e.g., a disease located in an organ part, is located in the organ as a whole [UC1] [UC8]*

Chain inclusion axiom stating that the property locatedIn propagates along the property partOf:
SubObjectPropertyOf(SubObjectPropertyChain(locatedIn partOf) locatedIn)

Use Cases and Requirements from literature

- Life Sciences
 - [UC1] a very common requirement, particularly in Life Sciences applications, is to express the propagation of one property along another property, see examples in [UC1] above: when querying for “all images with a tumor located in the frontal lobe”, it is important that the location property is transferred across the partOf property in order that tumors located in a partOf the frontal lobe are recognized as answering the query

- [UC8] An examination of OWL and the requirements of a large health care terminology: Without [property chain inclusion axioms], adoption of OWL by the SNOMED community would have required awkward workarounds with their attendant complications and complexities - effectively killing movement in that direction. With [them], we have a clear path to using OWL 1.1 for further development and integration with other biomedical ontologies.
 - http://owled2007.iut-velizy.uvsq.fr/PapersPDF/submission_26.pdf
- [UC6B] The typedef 'transitive_over' is defined in OBO as: "The id of another relationship type that this relationship type is transitive over. If P is transitive over Q, and the ontology has X P Y and Y Q Z then it follows that X P Z (term/type level). This should be translated by chain inclusion axiom
 - http://www.geneontology.org/GO.format.obo-1_2.shtml#S.2.2.2

7. New Datatype Mechanisms

Other extensions include the expansion of datatype expressiveness.

7.1. OWL mechanisms for user-defined datatypes

OWL 1.1 allows the definition of 'user-defined datatypes' through ***DatatypeRestriction***.

A ***DatatypeRestriction*** defines a data range constructed by applying a facet to a particular data range.

E.g; `DatatypeRestriction(xsd:integer minInclusive "18"^^xsd:integer))`
Restriction on integers to integers greater or equal to 18.

Syntax: `DatatypeRestriction(dataRange datatypeFacet restrictionValue)`
DatatypeRestriction(DR f v)

A restriction consists of a constant restriction value (e.g., 18) and a facet type (e.g.; *minInclusive*) that is applied to a given data range (e.g., *xsd:integer*). The following facet types are supported in OWL 1.1: *length*, *minLength*, *maxLength*, *pattern*, *minInclusive*, *minExclusive*, *maxInclusive*, *maxExclusive*, *totalDigits*, and *fractionDigits*.

New datatypes restrictions can be used in the ontology, as in the example below where the *DatatypeRestriction* is used to restrain the age range of individuals of the subclass *Adult* to *integer greater or equal to 18*.

E.g; `SubClassOf(Adult DataSomeValuesFrom(age DatatypeRestriction(xsd:integer minInclusive "18"^^xsd:integer)))`
Adults are necessary individuals whose age is some integer greater or equal to 18.

Examples

- **Life Sciences**

#18. *DatatypeRestriction* used for representing decimal greater or equal to 30 needed for specifying deep sulci [UC9]

E.g; EquivalentClasses(DeepSulcus (ObjectIntersectionOf Sulcus DataSomeValuesFrom(depth DatatypeRestriction(xsd:decimal minInclusive "30"^^xsd:decimal))))).
DeepSulcus are sulci whose depth is some decimal greater or equal to 30

#19. *DatatypeRestriction used for representing integers less or equal to 18 used for defining child (for pediatrics service) [UC10]*

EquivalentClasses(Child DataSomeValuesFrom(age DatatypeRestriction(xsd:integer minInclusive "18"^^xsd:integer))).
Child are individuals whose age is some integer <=18

#20. *DatatypeRestrictions used for representing people with an age between 16 and 18 [UC10]*

EquivalentClasses(Middle (ObjectIntersectionOf (DataSomeValuesFrom (age DatatypeRestriction(xsd:decimal minExclusive "16"^^xsd:decimal)) (DataSomeValuesFrom(age DatatypeRestriction(xsd:decimal maxInclusive "18"^^xsd:decimal))))).
*Middle are individuals whose age is some integer <= 18 and > 16**

*Note: at the moment OWL1.1 specification does not offer the possibility to use intervals as ranges but is supposed to allow multiple facets, thus intervals, in the future.

7.2. n-ary datatypes

OWL 1.1 allows simple relationships between values of functional data-valued properties.

e.g., DataSomeValuesFrom(width height greaterThan)
objects whose width is greater than their height

- Life Sciences

- [UC11] Clinical trials: TBD Vipul Kashyap?

DataAllValuesFrom(height weight owl:lessThan)
All values of height are smaller than the values of weight.

Patient that (testdate > enrollmentdate + 60)

in a clinical trial, the patient must have negative gadolinium based MRI of the contralateral breast, no more than 6 months prior to study entry.

- [UC10] Kidney Allocation Policy in France. At hospital, patients under 18 (child) depend on pediatric services while over 18 depend on adult services, but only child less than 16 years waiting for a transplant have a priority on the waiting list.
 - http://www.med.univ-rennes1.fr/lim/doc_163.pdf
 - http://www.med.univ-rennes1.fr/lim/doc_96.pdf (dialysis and transplantation ontology)
 - [UC11] Towards a Hybrid System Using an Ontology Enriched by Rules for the Semantic Annotation of Brain MRI Images (RR2007): all n-ary relations were transformed into binary relations, using reification for example an artificial class AttributedEntity was defined for it.
 - http://www.med.univ-rennes1.fr/lim/doc_165.pdf
 - <http://www.med.univ-rennes1.fr/~cgolb/Protege2005/WSProtege-CG.pdf>

- o [UC12]: n-ary built-in relations in Workshop Protégé with Rules 2005 TBD CG?

8. Punning

OWL 1.1 allows a basic form of meta-modelling, called punning.

A single name can be used for several purposes, for naming an individual, a class, a property; for example, Person can at the same time be the name of a class and the name of an individual. The different uses of a name are, however, completely independent, and from a semantic point of view they can be thought as distinct names, e.g., Person-the-Class and Person-the-Individual¹. [ref Next Steps to OWL]. No aspect of the use of the name as an individual has any effect on the meaning of the name as a class. Such a treatment of metamodeling is often called *punning*.

e.g., `OWLClass(Person); OWLClass(Human) (1)`
`SameIndividual(Person Human) (2)`
`Individual(John Person) (3)`
Person is used as the name of a class in (1) (3), and as the name of an individual in (2)
Assertion (3) is syntactically valid, but does not entail Individual(John Human) because the individual named by Person in (2) is different from the class named by Person in (3).

Examples

Use Cases and Requirements from literature

- Life Sciences
 - o [UC6B] OBO defines the typedef 'synonym' which has been translated into an OWL1.1 annotation.
However, OBO distinguishes different types of synonym properties: “The synonym scope may be one of four values: EXACT, BROAD, NARROW, RELATED” http://www.geneontology.org/GO.format.obo-1_2.shtml#S.2.2.2
A possibility might be to define synonym both as an annotation and as an objectProperty and to assert subclass and transitive axioms on the relevant properties, e.g., *transitiveObjectProperty* (exact_synonym)

9. Annotations

9.1. Annotation of entities

Examples

¹ Note that OWL1.1 breaks from RDF

- **Life Sciences**

#21. *Labels of Gene Ontology classes are represented as annotations [UC6B]*

`EntityAnnotation(OWLClass(GO'0001555) Label("oocyte'growth"))`
The class GO'0001555 has the label oocyte'growth

Use Cases and Requirements from literature

- [UC6B] OBO distinguishes different types of synonym properties: “The synonym scope may be one of four values: EXACT, BROAD, NARROW, RELATED”
<http://www.geneontology.org/GO.format.obo-1.2.shtml#S.2.2.2>
- [UC6B] Use Case : Mapping OBO to OWL1.1. OBO and OWL: Leveraging Semantic Web Technologies for the Life Sciences (ISWC 2007) gives many examples of OWL1.1 use of annotations
 - http://www.med.univ-rennes1.fr/lim/doc_178.pdf
 - http://www.med.univ-rennes1.fr/lim/doc_175.pdf

- [UC 14] Alan Ruttenberg TBD ?

“For example, it is desirable that an "annotation property" for an editing time stamp should have a range that is xsd:date, or, for SKOS, we would like to be create subproperties of rdfs:label. “

9.2. Annotation of axioms

Part II New extensions Use-cases and requirements (draft)

10. Rules

- 10.1.1. SWRL rules
- 10.1.2. DL-safe Horn rule extension
- 10.1.3. Logic Programming rules
- 10.1.4. Other

11. Recapitulative Table

Feature	Example	UC	URI
Syntactic sugar			
<u>DisjointUnion</u>	<p>HCLS</p> <p>#1-#2</p> <p>#3</p> <p>#4</p> <p>Automotive</p> <p>#5</p> <p>Telecommunication</p> <p>Manufacturing</p> <p>Earth and Space</p>	<p>UC1</p> <p>UC2</p> <p>UC3</p> <p>UC4</p>	<p>http://www.med.univ-rennes1.fr/lim/doc'61.pdf</p> <p>http://sig.biostr.washington.edu/projects/fm/omEditPublications.html</p> <p>http://www.med.univ-rennes1.fr/lim/doc'128.pdf</p> <p>http://www.ea3888.univ-rennes1.fr/dameron/publis/2007protege-dameron.pdf</p> <p>http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper28.pdf</p> <p>http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-258/paper08.pdf</p>
<u>DisjointClass</u>			
<u>NegativeObjectPropertyAssertion</u>			
<u>NegativeDataPropertyAssertion</u>			
New constructs for properties			
New Restrictions			
Qualified Cardinality Restrictions: ObjectMinCardinality ObjectMaxCardinality ObjectExactCardinality DataMinCardinality DataMaxCardinality DataExactCardinality			
ObjectExistsSelf			
New Characteristics			
ObjectPropertyReflexive			
ObjectPropertyIrreflexive			
ObjectPropertyAsymmetric			
New Relationships			
DisjointObjectProperties			

DisjointDataProperties			
New datatype mechanisms			
User-defined datatypes			
n-ary datatypes			
Metamodelling			
Punning			
Annotations			