Protocol for Web Description Resources (POWDER): Formal Semantics

W3C Recommendation 1 September 2009

This version
http://www.w3.org/TR/2009/REC-powder-formal-20090901/
Latest version
http://www.w3.org/TR/powder-formal/
Previous version
http://www.w3.org/TR/2009/PR-powder-formal-20090604/

Editors:
Stasinos Konstantopoulos, Institute of Informatics & Telecommunications (IIT), NCSR "Demokritos"
Phil Archer, Institute of Informatics & Telecommunications (IIT), NCSR "Demokritos" (formerly with FOSI)

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

See also **translations**.

Copyright © 2009 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark and document use rules apply.

Abstract

This document underpins the Protocol for Web Description Resources (POWDER). It describes how the relatively simple operational format of a POWDER document can be transformed through two stages: first into a more tightly constrained XML format (POWDER-BASE), and then into an RDF/OWL encoding (POWDER-S) that may be processed by Semantic Web tools. Such processing is only possible, however, if tools implement the semantic extension defined within this document. The formal semantics of POWDER are best understood after the reader is acquainted with the Description Resources [DR] and Grouping of Resources [GROUP] documents.

Status of this document

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/.

This document is a W3C Recommendation that was developed by the POWDER Working Group.

Please see the Working Group's implementation report and Disposition of Last Call Comments. The disposition of comments received during previous calls are also available. Changes since the previous version of this document are minor in nature and are fully documented in the Change log.

Publication of this Recommendation is synchronized with several other documents:

- POWDER: Description Resources (Recommendation)
- POWDER: Grouping of Resources (Recommendation)
- POWDER: Primer (Working Group Note)
- POWDER: Test Suite (Working Group Note)

The W3C Membership and other interested parties are invited to review the document and send comments to public-powderwg@w3.org (with public archive).

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.

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 & Scope

The Protocol for Web Description Resources, POWDER, offers a simple method of associating RDF data with groups of resources. Its primary 'unit of information' is the Description Resource (DR). This comprises three elements:

- attribution (who is providing the description)
- scope (defined as a set of IRIs over which the description applies to the resources de-referenced from those IRIs)
- the description itself (the 'descriptor set').

To some extent, this approach is in tension with the core semantics of RDF and OWL. To resolve that tension, it is necessary to extend RDF semantics as described below. In order to minimize the required extension, while at the same time preserving the relatively simple encoding of POWDER in XML which is generally readable by humans, we define a multi-layered approach. The operational semantics, i.e. the encoding of POWDER in XML, is first transformed into a more restricted XML encoding that is less easily understood by humans and depends on matching IRIs against regular expressions to determine whether or not they are within the scope of the DR. This latter encoding is, in its own turn, transformed into the extended-RDF encoding.

This document formalizes the semantics of POWDER to ensure consistency between the different layers. It is of particular importance to RDF/OWL-based, as opposed to purely operational, implementations of POWDER.

The data model makes the attribution element mandatory for all POWDER documents. These may contain any number of Description Resources (DRs) that effectively inherit the attribution of the document as a whole. Descriptor sets may also be included independently of a specific DR, and these too inherit the attribution. This model persists throughout the layers of the POWDER model, which are as follows:

POWDER

The operational encoding, a dialect of XML, that transports the RDF data. It is expected that POWDER will typically be published and processed in this form.

POWDER's resource grouping methods are mostly geared towards URLs and Information Resources as defined in the Architecture of the World Wide Web [WEBARCH].

POWDER-BASE

This is a largely theoretical XML encoding of POWDER that reduces all means of grouping resources according to their IRI into a single grouping method, that of matching IRIs against arbitrary regular expressions.

POWDER-BASE is provided as a means of formally specifying the semantics of the various IRI grouping methods defined in POWDER. POWDER-BASE is generated automatically from POWDER by means of the GRDDL transform [GRDDL] that is associated with the POWDER namespace and defined in this document.

Elements not concerned with IRI set definition are identical in POWDER and POWDER-BASE.

POWDER-S (Semantic POWDER)

The Semantic encoding uses a fragment of RDF/OWL that has been extended in a way that facilitates the matching of the string representation of a resource's identifier against a regular expression. The regular expression syntax used is defined by XML schema as modified by XQuery 1.0 and XPath 2.0 Functions and Operators [XQXP].

OWL classes are used to represent sets of resources, grouped according to their IRI and according to their properties (descriptors). Resources are described by asserting that a class that defines a set of IRIs is a sub class of a descriptor-defined class-set. Attribution is provided by way of OWL annotation properties.

A small RDF vocabulary is needed to support POWDER-S. Although it is valid RDF/OWL, generic tools will only be able to process the semantics of POWDER-S if they implement the necessary extension defined in this document.

POWDER-S is generated from POWDER-BASE by means of the GRDDL transform [GRDDL] that is associated with the POWDER namespace. POWDER-S MAY be created directly, but this is generally inadvisable since, whilst a POWDER Processor MUST understand and process POWDER-BASE and SHOULD understand POWDER, it MAY NOT understand and process POWDER-S. The aim of POWDER-S is to make the data available to the broader Semantic Web via GRDDL, not to create an alternative encoding.

The conformance criteria for a POWDER Processor are given in the Description Resources document [DR].

The GRDDL transform from POWDER to POWDER-BASE to POWDER-S is defined in this document. The transform from POWDER to POWDER-BASE only affects elements related to IRI set definition, taking tokens as input and yielding a set of regular expressions as output. Such a transformation will be achievable by several means in different application environments. One available method is to use the XSLT stylesheet associated with the POWDER XML Schema [WDR2B], which uses XSLT 2.

A separate XSLT stylesheet, which only uses XSLT 1 is associated with the POWDER-BASE schema and achieves the more complex task of transforming POWDER-BASE to POWDER-S [B2S]. Both stylesheets are consistent with the normative text in this document, but their syntactic specifics are not normative; in effect, a POWDER processor MAY use different transforms to produce syntactically different but semantically equivalent OWL/RDF for processing POWDER documents.

Description Resources are defined separately [DR] and a further document defines the creation of IRI sets [GROUP]. Readers should be familiar with those documents before proceeding with this one. The full set of POWDER documents also includes its Use Cases, Primer and Test Suite, together with two namespace documents: the POWDER XML schema [WDR] and POWDER-S vocabulary [WDRS].

1.1 Namespaces, Terminology and Conventions Used in This Document.

The POWDER vocabulary namespace is `http://www.w3.org/2007/05/powder#` for which we use the prefix `wdr` The POWDER-S vocabulary namespace is `http://www.w3.org/2007/05/powder-s#` for which we use the prefix `wdrs` All prefixes used in this document, together with their associated namespaces, are shown in the table below.

Prefix	Namespace
`wdr`	`http://www.w3.org/2007/05/powder#`
`wdrs`	`http://www.w3.org/2007/05/powder-s#`
`rdf`	`http://www.w3.org/1999/02/22-rdf-syntax-ns#`
`rdfs`	`http://www.w3.org/2000/01/rdf-schema#"`
`owl`	`http://www.w3.org/2002/07/owl#`
`xsd`	`http://www.w3.org/2001/XMLSchema#`
`xsl`	`http://www.w3.org/1999/XSL/Transform`
`ex`	An arbitrary prefix used to denote an 'example vocabulary' from the `example.org` domain.

Table 1: Prefixes and Namespaces used in this document

Unqualified elements in this document are from the `wdr` namespace.

In this document, the words MUST, MUST NOT, SHOULD, SHOULD NOT, MAY and MAY NOT are to be interpreted as described in RFC2119 [RFC2119].

For convenience and transparency, we have used the RDF/XML serialization for POWDER-S as we have throughout the document set. Other serializations, such as N3 [N3], are equally valid for POWDER-S. The GRDDL transformations associated with the POWDER namespace, which use XSLT to effect the transform, produces RDF/XML as output.

Examples in this document show fragments of data and each is linked to two external files that mirror the data in the text, one serialized as RDF/XML and one as Turtle [TTL]. However, in order to be valid documents, the external files include generic data not shown in the text that has been taken largely from examples 2-1 and 2-3 in the Description Resources document [DR].

The regular expressions in this documents should be interpreted as per Section 7.6.1 of the XQuery/XPath Recommendation [XQXP].

1.2 Conformance

Conformant implementations of this Recommendation will transform POWDER to POWDER-BASE and/or POWDER-BASE to POWDER-S documents as described in sections 2 - 4.5 below.

The conformance criteria for a POWDER processor are specified in the Description Resources document [DR]. This makes it clear that a conformant POWDER Processor MAY NOT process POWDER-S documents. In applications that do process POWDER-S, however, conformance with this document means that the software will implement the Semantic Extension defined in Section 4.3 such that the results of the queries set out in Section 5 are indistinguishable from those returned from a conformant POWDER Processor working purely with POWDER or POWDER-BASE.

2 Attribution Element Semantics

The `attribution` element, present in all POWDER documents, provides data about the authorship, validity period, and other issues that a user or user agent can use when deciding whether or not to confer their trust on a POWDER document.

2.1 POWDER Attribution Semantics

Most `attribution` elements are not involved in IRI grouping, and as such are untouched during the transformation from POWDER to POWDER-BASE. The only exception is `abouthosts`, which sets an outer limit on the resources described by the DRs within the document. POWDER `abouthosts` elements are translated into POWDER-BASE `aboutregex` elements, as discussed in Section 4.5 below.

2.2 POWDER-S Attribution Semantics

Since the `attribution` element provides meta-data about the document itself, it is transformed from POWDER (through POWDER-BASE) into POWDER-S as annotation properties of an `owl:Ontology` instance referring to `rdf:about=""` (i.e., the current document). This data does not receive any further semantics, but is meaningful to POWDER tools when deciding whether a POWDER document *as a whole* should be taken into account or discarded. The only exception is the POWDER `aboutregex` element, as discussed in Section 4.5 below.

With the explicit exception of `abouthosts`, child elements of the `attribution` POWDER element are reproduced as OWL annotation properties in the POWDER-S instance. This is always possible as these POWDER elements are required to have either resource references or datatype values. This general rule applies to the POWDER elements `issuedby`, `issued`, `validfrom`, `validuntil`, `certifiedby` and `supportedby` where the only transformation necessary is to make their (transformed) namespace explicit. In each case the same string is used as their element name and `wdrs` property name. The `wdrs:issuedby` property is treated in the same manner, but is noteworthy as it is required for all POWDER documents.

POWDER provides a shortcut to avoid the automatic need to declare the `rdf` namespace in all POWDER documents. Where child elements of the `attribution` element of a POWDER document point to an IRI, this is provided as the value of a `src` attribute. The GRDDL transformation renders such attributes as values for `rdf:resource`.

Example 2-1 shows the generic semantics of the `attribution` element.

Example 2-1: The Generic Semantics of the `attribution` Element

POWDER [XML]

```
<attribution>
  <ex:property1>value</ex:property1>
  <ex:property2 rdf:resource="http://example.org/foo.rdf#frag" />
  <ex:property3 src="http://example.com/bar.rdf#frag" />
</attribution>
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Ontology rdf:about="">
  <ex:property1>value</ex:property1>
  <ex:property2 rdf:resource="http://example.org/foo.rdf#frag" />
  <ex:property3 rdf:resource="http://example.org/foo.rdf#frag" />
</<owl:Ontology>
```

As noted above, some elements within the POWDER namespace that are treated differently or have noteworthy semantics:

issuedby
: The `issuedby` element in POWDER documents MUST contain an external reference to an RDF resource. This resource SHOULD identify the creator of the POWDER document; a POWDER processor MAY use this information to evaluate the trust-worthiness of the document. There is no restriction on the range of the `wdrs:issuedby` property, but POWDER authors are advised to use well-known and widely-used vocabularies, such as Dublin Core [DC] or FOAF [FOAF], and refer to instances of `dc:Agent` or `foaf:Agent`, respectively.

issued, validfrom, validuntil
: The `issued`, `validfrom`, and `validuntil` elements in POWDER documents MUST contain a value conformant with the XML dateTime datatype. The transformation to POWDER-S consists of simply creating annotation properties in the `wdrs` namespace with these values. The value of `issued` SHOULD be the date and time of formal issuance (e.g., publication) of the document. The values of `validfrom` and `validuntil` SHOULD be specifying the validity period of the document.

certifiedby, supportedby
: The `certifiedby` and `supportedby` elements in POWDER documents MUST have a `src` attribute that has an IRI as its value.

abouthosts
: The `abouthosts` element is discussed in Section 4.5 below.

Example 2-2 below shows all the POWDER-specific attribution elements. Note that there is no `abouthosts` element in the example, as it will be discussed in Section 4.3 below.

Example 2-2: POWDER-Specific Child Elements of the `attribution` Element

POWDER [XML]

```
<attribution>
  <issuedby src="http://example.org/company.rdf#me" />
  <issued>2007-12-23T00:00:00</issued>
  <validfrom>2008-01-01T00:00:00</validfrom>
  <validuntil>2008-12-31T23:59:59</validuntil>
  <certifiedby src="http://authority.example/powder.xml" />
  <supportedby src="http://service.example.com?id=abc" />
</attribution>
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Ontology rdf:about="">
  <wdrs:issuedby rdf:resource="http://example.org/company.rdf#me" />
  <wdrs:issued>2008-12-23T00:00:00</wdrs:issued>
  <wdrs:validfrom>2008-01-01T00:00:00</wdrs:validfrom>
  <wdrs:validuntil>2008-12-31T23:59:59</wdrs:validuntil>
  <wdrs:certifiedby rdf:resource="http://authority.example/powder.xml" />
  <wdrs:supportedby rdf:resource="http://service.example.com?id=abc" />
</owl:Ontology>
```

There is one final POWDER element that is transformed into an annotation of the POWDER-S document: the `more` element. This may occur as a child of the root (`powder`) element and provides a link from one POWDER document to another. It is simply transformed into a `rdfs:seeAlso` property of the POWDER-S document (within the ontology header).

3 Description Resource Semantics

Description Resources use vocabularies defined in RDF and/or plain string literals (tags) to describe resources de-referenced from instances of the IRI set. Since descriptor set elements are not involved in the specification of the IRI set itself, they are transferred verbatim from POWDER to POWDER-BASE. Example 3-1 below shows a generic example of a DR in which the IRI set has been elided for clarity (the semantics of the IRI set is discussed in Section 4 below).

Example 3-1: A Generic Example of A Descriptor Set and Tag Set within a DR [XML]

```
<dr>
  <iriset>…</iriset>

  <descriptorset>
    <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
    <ex:shape>square</ex:shape>
  </descriptorset>

  <tagset>
```

```
 8     <tag>red</tag>
 9     <tag>light</tag>
10   </tagset>
11 </dr>
```

The `ex:finish` element specifies that the `ex:finish` relation holds between all resources specified by `iriset` and the http://example.org/vocab#shiny resource.

The content of `ex:shape` is interpreted as a string literal. The `ex:shape` element specifies that all resources in `iriset` have the value "square" for the `ex:shape` dataproperty.

`tag` is a string property defined by POWDER. Its content is a single string literal, possibly including spaces.

The overall description of the resources in `iriset` is the union of the descriptions in the `descriptorset` and the `tagset`. In our example these are:

an ex:finish relation to http://example.org/vocab#shiny
AND
an ex:shape of "square"
AND
the tags "red" and "light"

We formally interpret the above as follows: there is an OWL class containing all resources that share all of these properties, and there is an OWL class of all resources denoted by `iriset`, and the latter is a subset of the former. In POWDER-S we say:

Example 3-2 The POWDER-S Encoding of Example 3-1 [RDF/XML,TURTLE]

```
 1  <owl:Class rdf:nodeID="iriset_1">
 2    all resources specified by <iriset>...</iriset>
 3  </owl:Class>

 4  <owl:Class rdf:nodeID="descriptorset_1">
 5    <rdfs:subClassOf>
 6      <owl:Class>
 7        <owl:intersectionOf rdf:parseType="Collection">
 8          <owl:Restriction>
 9            <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
10            <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
11          </owl:Restriction>
12         <owl:Restriction>
13           <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
14           <owl:hasValue>square</owl:hasValue>
15         </owl:Restriction>
16       </owl:intersectionOf>
17     </owl:Class>
18   </rdfs:subClassOf>
19 </owl:Class>

20 <owl:Class rdf:nodeID="tagset_1">
21   <rdfs:subClassOf>
22     <owl:Class>
23       <owl:intersectionOf rdf:parseType="Collection">
24         <owl:Restriction>
25           <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#tag"/>
26           <owl:hasValue>red</owl:hasValue>
27         </owl:Restriction>
28         <owl:Restriction>
29           <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#tag"/>
30           <owl:hasValue>light</owl:hasValue>
31         </owl:Restriction>
32       </owl:intersectionOf>
33     </owl:Class>
34   </rdfs:subClassOf;>
35 </owl:Class>

36 <owl:Class rdf:nodeID="iriset_1">
37   <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
38   <rdfs:subClassOf rdf:nodeID="tagset_1"/>
39 </owl:Class>
```

It is possible to have more than one `iriset` element, in which case a resource receives all of the descriptions by belonging to any one of the corresponding IRI sets. For example:

Example 3-3: A DR with Multiple IRI Sets in its Scope [XML]

```
1  <dr>
2   <iriset>.1.</iriset>
3   <iriset>.2.</iriset>

4    <descriptorset xml:id="silver">
5     <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
6   </descriptorset>
7  </dr>
```

receives the following semantics:

Example 3-4: The POWDER-S encoding of Example 3-3 [RDF/XML,TURTLE]

```
 1  <owl:Class rdf:nodeID="iriset_1">
 2     all resources specified by <iriset>.1.</iriset>
 3  </owl:Class>

 5  <owl:Class rdf:nodeID="iriset_2">
 6     all resources specified by <iriset>.2.</iriset>
 7  </owl:Class>

 8  <owl:Class rdf:ID="silver">
 9    <rdfs:subClassOf>
10      <owl:Class>
11        <owl:intersectionOf rdf:parseType="Collection">
12          <owl:Restriction>
13            <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
14            <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
15          </owl:Restriction>
16        </owl:intersectionOf>
17      </owl:Class>
18    </rdfs:subClassOf>
19 </owl:Class>

20 <owl:Class rdf:nodeID="iriset_1">
21   <rdfs:subClassOf rdf:resource="#silver"/>
22 </owl:Class>

23 <owl:Class rdf:nodeID="iriset_2">
24   <rdfs:subClassOf rdf:resource="#silver"/>
25 </owl:Class>
```

Examples 3-3 and 3-4 also show that if a `descriptorset` element has an ID of its own, this is used in the POWDER-S document. This is reflected in the way that the sub class relationship is asserted. Where no `xml:id` attribute is set in the original POWDER document, the transform assigns `rdf:nodeID` identifiers for the blank nodes. As a result, `rdf:nodeID` attributes are used within the POWDER-S document in the sub class assertion (see lines 36 - 39 in Example 3-2). However, where the original POWDER document includes an `xml:id` attribute, as in line 4 of Example 3-3, the sub class assertions in lines 21 and 24 of Example 3-4 is correctly asserted using the `rdf:resource` attribute.

A POWDER processor is free to choose any traversal policy for treating multiple `iriset` elements in a DR: first match wins, last match wins, shortest `iriset` first, and so on, as long as all `iriset` elements are tried before deciding that DR does not apply to a candidate resource (candidate resource is defined in the Grouping of Resources document [GROUP]). However, DR authors may use the order of the `iriset` elements to suggest an efficient scope evaluation strategy, by putting the `iriset` with the widest coverage first, so that a processor that chooses to follow the `iriset` elements in document order is more likely to terminate the evaluation after fewer checks.

3.1 Multiple Description Resources in a Single POWDER Document

A POWDER document may have any number of DRs, all of which are simultaneously asserted and ordering is not important. So, for example:

Example 3-5: A POWDER Document Containing Multiple DRs [XML]

```
<powder>

  <dr>
    <iriset>.1.</iriset>
    <descriptorset>
      <ex:shape>square</ex:shape>
    </descriptorset>
  </dr>

  <dr>
    <iriset>.2.</iriset>
    <descriptorset>
      <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
    </descriptorset>
  </dr>

</powder>
```

receives the following semantics:

Example 3-6: The POWDER-S Encoding of Example 3-5 [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="iriset_1">
  all resources specified by <iriset>.1.</iriset>
</owl:Class>

<owl:Class rdf:nodeID="descriptorset_1">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
          <owl:hasValue>square</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:nodeID="iriset_1">
  <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
</owl:Class>

<owl:Class rdf:nodeID="iriset_2">
  all resources specified by <iriset>.2.</iriset>
</owl:Class>

<owl:Class rdf:nodeID="descriptorset_2">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
          <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:nodeID="iriset_2">
  <rdfs:subClassOf rdf:nodeID="descriptorset_2"/>
</owl:Class>
```

The `owl:intersectionOf` of a singleton collection in both descriptor sets, although redundant, is a result of the GRDDL transformation. As noted above, syntactically different but semantically equivalent representations are equally valid.

Note that `iriset_1` and `iriset_2` are not necessarily disjoint — some resources may be both shiny AND square.

A POWDER document may have an `ol` element which is an ordered list of DRs. Such a list receives first-match semantics, that is, when seeking the description of a candidate IRI, processors extract the descriptor set from the first DR in the ordered list in which it is in scope. `ol` elements allow the easy expression of exceptions to more general rules. So, for example:

Example 3-7: An Ordered List of DRs [XML]

```
<ol>

  <dr>
    <iriset>.1.</iriset>
    <descriptorset>
      <ex:shape>square</ex:shape>
    </descriptorset>
  </dr>

  <dr>
    <iriset>.2.</iriset>
    <descriptorset>
      <ex:shape>round</ex:shape>
    </descriptorset>
  </dr>

  <dr>
    <iriset>.3.</iriset>
    <descriptorset>
      <ex:shape>triangular</ex:shape>
    </descriptorset>
  </dr>

</ol>
```

receives the following semantics, where belonging to description_1 automatically precludes belonging to description_2 and description_3; and belonging to description_2 automatically precludes belonging to description_3:

Example 3-8: The POWDER-S Encoding of Example 3-7 [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="iriset_1">
  all resources specified by <iriset>.1.</iriset>
</owl:Class>

<owl:Class rdf:nodeID="iriset_1_not">
  all resources not specified by <iriset>.1.</iriset>
</owl:Class>

<owl:Class rdf:nodeID="descriptorset_1">
 <rdfs:subClassOf>
   <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
```

```
13            <owl:hasValue>square</owl:hasValue>
14          </owl:Restriction>
15        </owl:intersectionOf>
16      </owl:Class>
17    </rdfs:subClassOf>
18 </owl:Class>

19 <owl:Class rdf:nodeID="iriset_1">
20   <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
21 </owl:Class>

22 <owl:Class rdf:nodeID="iriset_2">
23   <owl:equivalentClass>
24     <owl:Class>
25       <owl:intersectionOf rdf:parseType="Collection">
26         all resources specified by <iriset>.2.</iriset>
27         <owl:Class rdf:nodeID="iriset_1_not" />
28       </owl:intersectionOf>
29     </owl:Class>
30   </owl:equivalentClass>
31 </owl:Class>

32 <owl:Class rdf:nodeID="iriset_2_not">
33   all resources not specified by <iriset>.2.</iriset>
34 </owl:Class>

35 <owl:Class rdf:nodeID="descriptorset_2">
36   <rdfs:subClassOf>
37     <owl:Class>
38       <owl:intersectionOf rdf:parseType="Collection">
39         <owl:Restriction>
40           <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
41           <owl:hasValue>round</owl:hasValue>
42         </owl:Restriction>
43       </owl:intersectionOf>
44     </owl:Class>
45   </rdfs:subClassOf>
46 </owl:Class>

47 <owl:Class rdf:nodeID="iriset_2">
48   <rdfs:subClassOf rdf:nodeID="descriptorset_2"/>
49 </owl:Class>

50 <owl:Class rdf:nodeID="iriset_3">
51   <owl:equivalentClass>
52     <owl:Class>
53       <owl:intersectionOf rdf:parseType="Collection">
54         all resources specified by <iriset>.3.</iriset>
55         <owl:Class rdf:nodeID="iriset_1_not" />
56         <owl:Class rdf:nodeID="iriset_2_not" />
57       </owl:intersectionOf>
58     </owl:Class>
59   </owl:equivalentClass>
60 </owl:Class>

61 <owl:Class rdf:nodeID="descriptorset_3">
62   <rdfs:subClassOf>
63     <owl:Class>
64       <owl:intersectionOf rdf:parseType="Collection">
65         <owl:Restriction>
66           <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
67           <owl:hasValue>triangular</owl:hasValue>
68         </owl:Restriction>
69       </owl:intersectionOf>
70     </owl:Class>
71   </rdfs:subClassOf>
72 </owl:Class>

73 <owl:Class rdf:nodeID="iriset_3">
74   <rdfs:subClassOf rdf:nodeID="descriptorset_3"/>
75 </owl:Class>
```

The GRDDL transformation does a lot of work here so let's break it down:

For each DR in the ordered list:

For each IRI set in the DR, two classes are defined: one that represents all resources that are members of the IRI set (as in the earlier examples) and another that defines all the resources that are not members of the IRI set (how this is done is the subject of Section 4). In the example these are given the nodeIDs iriset_n and iriset_n_not.

It should be noted any iriset_n_not class includes the resources that are not included in the raw IRI set definition, as opposed to the iriset_n class which excludes the resources in the previous irisets (see lines 27, 55 & 56).

The following assertion is made: the intersection of the iriset of the current DR and the 'not' irisets from all preceding DRs is subsumed under the descriptorset (or tagset) of the current DR. (Note that there must be only one IRI set generated for each DR, possibly using owl:unionOf, regardless of how the POWDER/XML original is expressed.)

It follows that the first DR in the list yields a simple sub class relationship between its IRI set(s) and descriptor set(s); and that the final DR in the list's 'iriset_not' is not used and may safely be omitted.

3.2 Descriptor Set Semantics

3.2.1 Descriptor Sets expressed as RDF Properties and Values

A descriptor set contains RDF properties that have fillers that are not blank nodes and that do not identify either a class or a property, as shown in Example 3-9 below. Subsumption of descriptorsets is expressed as discussed in Section 3.2.2 below.

Example 3-9: A Descriptor Set Containing an RDF Property with a Literal Value and One with an RDF Resource as its Value.

POWDER [XML]

```
1  <descriptorset>
2    <ex:shape>square</ex:shape>
3    <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
4  </descriptorset>
```

POWDER-S [RDF/XML,TURTLE]

```
1 <owl:Class rdf:nodeID="descriptorset_1">
2   <rdfs:subClassOf>
3     <owl:Class>
2       <owl:intersectionOf rdf:parseType="Collection">
3         <owl:Restriction>
4           <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
5           <owl:hasValue>square</owl:hasValue>
6         </owl:Restriction>
7         <owl:Restriction>
8           <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
9           <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
12        </owl:Restriction>
13      </owl:intersectionOf>
14    </owl:Class>
15  </rdfs:subClassOf>
16 </owl:Class>
```

3.2.2 Asserting the `rdf:type` Relationship

Asserting the `rdf:type` property, i.e. that all elements within an IRI set are instances of a particular OWL or RDFS Class, is achieved most easily using the `typeof` element which takes the IRI of the class as the value of its `src` attribute as shown in Example 3-10 below. The POWDER-S translation of the `descriptorset` element intersects `typeof` classes with the property restrictions (if any) in the `descriptorset`.

Example 3-10: A Descriptor Set Asserting that IRIs within its Scope are Instances of a Class.

POWDER [XML]

```
<descriptorset>
  <typeof src="http://example.org/vocab#Conformance_Class" />
  <ex:shape>square</ex:shape>
</descriptorset>
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="descriptorset_1">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="http://example.org/vocab#Conformance_Class" />
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
          <owl:hasValue>square</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>
```

This is particularly useful in the context of the POWDER use cases [USECASES] when claiming that resources on a Web site conform to a published set of criteria. In such situations, multiple criteria can be grouped together by defining the class of resources that satisfy all the criteria as the intersection of a number of property restrictions; series of increasingly stricter conformance levels can be defined as a subsumption hierarchy of such classes.

If used directly, the `rdf:type` property will be treated in the same way as the `typeof` element in the POWDER to POWDER-S transform.

3.2.3 Referring to External Descriptor Sets

A descriptor set may defer to a second descriptor set in another POWDER document using the `src` attribute. The transformation uses the value of the `src` attribute directly in the sub class relationship as shown below.

Example 3-11: A Descriptor Set Referring to one in an External Document

POWDER [XML]

```
<descriptorset src="http://remote.example.org/powder2.xml#d1" />
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="iriset_1">
  <rdfs:subClassOf rdf:resource="http://remote.example.org/powder2.xml#d1"/>
</owl:Class>
```

3.2.4 Further Descriptors

There are two POWDER elements that can be included as child elements of `descriptorset` that are mapped to property restrictions in POWDER-S. In both cases the same string is used as the element name in POWDER and vocabulary term in POWDER-S:

`shalsum`
: A SHA-1 sum of the described resource

`certified`
: An element of type `xsd:boolean` used when a DR certifies another resource.

The usage of both `shalsum` and `certified` is shown in section 5.2 of the Description Resources document [DR].

We define further elements that can be included as child elements of `descriptorset` that, when transformed into POWDER-S, become annotation properties of the descriptive OWL class (not property restrictions).

`displaytext`
: is transformed to `wdrs:text`. The text supplied as the value of this element may be displayed in user agents.

`displayicon`
: has a `src` attribute, the value of which is a URI (or IRI) u which, in POWDER-S, becomes `wdrs:logo rdf:resource="`u`"`. The referred-to image may be displayed in user agents.

`seealso`, `label`, `comment`
: Each of these elements is transformed into an annotation of the OWL class using the term from the `rdfs` vocabulary with which it shares a local name. For the avoidance of doubt:

```
<seealso src="http://www.example.com/page.html" />
<label>An example to us all</label>
<comment>Comments make code easier to read</comment>
```

: are transformed into:

```
<rdfs:seeAlso rdf:resource="http://www.example.com/page.html" />
<rdfs:label>An example to us all</rdfs:label>
<rdfs:comment>Comments make code easier to read</rdfs:comment>
```

Usage of these elements is exemplified in the following section. As with `rdf:type`, they are provided as shortcuts within POWDER — the direct use of `rdfs:seeAlso`, `rdfs:comment` and `rdfs:label` will be rendered in exactly the same way (as annotations and not property restrictions) by the transform.

No other property from the `rdfs` namespace receives any special treatment. If it is meaningful, in a POWDER document, to use terms from the `rdfs` vocabulary other than those mentioned above, the POWDER author should be aware that they will be transformed into OWL property restrictions in exactly the same manner as properties from any other namespace.

3.3 Tag Set Semantics

The semantics of the (free text) tags are similar to those for properties with literal values. Each tag given in a `tagset` element in a POWDER document is a value for the RDF datatype property `wdrs:tag` as shown below. Note also the use of the `seealso` (which puts the tags in context), `label` and `comment` elements described in the previous section.

Example 3-12: A Tag Set.

POWDER [XML]

```
<tagset>
  <label>Tags for the London landmark</label>
  <tag>London</tag>
  <tag>Swiss Re</tag>
  <tag>gherkin</tag>
  <seealso src="http://encyclopaedia.example.com/gherkin.html" />
  <seealso src="http://photo.example.com/gherkin.jpg" />
  <comment>Tags are linked to specific resources that contextualize them</comment>
</tagset>
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="tagset_1">
  <rdfs:label>Tags for the London landmark</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder#tag" />
          <owl:hasValue>London</owl:hasValue>
```

```
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder#tag" />
          <owl:hasValue>Swiss Re</owl:hasValue>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder#tag" />
          <owl:hasValue>gherkin</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
  <rdfs:seeAlso rdf:resource="http://encyclopaedia.example.com/gherkin.html" />
  <rdfs:seeAlso rdf:resource="http://photo.example.com/gherkin.jpg" />
  <rdfs:comment>Tags are linked to specific resources that contextualize them</rdfs:comment>
</owl:Class>
```

4 IRI Set Semantics

The previous sections have shown that the semantics of several elements of a POWDER document can be obtained by applying the GRDDL transform associated with the namespace to generate native RDF/OWL as POWDER-S. This is not so for the IRI set element which, although transformed into valid RDF/OWL syntax, does not express the full semantics.

The IRI constraints defined in the POWDER Grouping of Resources document [GROUP] are given regular-expression semantics by the first part of the GRDDL transform from POWDER to POWDER-BASE. Regular-expression IRI groups are, in their turn, given semantics using datarange restrictions by the POWDER-BASE to POWDER-S transformation. It is noteworthy that the value space of POWDER's IRI constraints is, for the most part, a white space separated list of alternative values. This makes POWDER in its XML form relatively simple, but the implications for the semantics are substantial.

4.1 White Space and List Pre-Processing

Many elements of a POWDER IRI set definition have white space separated lists of strings as their value. White space is any of U+0009, U+000A, U+000D and U+0020. A space-separated list is a string in which the items are separated by one or more space characters (in any order). The string may also be prefixed or suffixed with zero or more of those characters. The GRDDL transform associated with the POWDER namespace converts these into components of a regular expression for use in POWDER-BASE and POWDER-S by following the steps set out below:

- Replace any sequence of space characters with a single space (U+0020) character, dropping any leading or trailing U+0020 characters
- Replace each remaining space (U+0020) character with the vertical bar character | (U+0124)
- Escape all instances of the following characters within the string using a \ character

```
. \ ? * + { } ( ) [ ] ! " # % & ' , - / : ; = > @ [ ] _ ` ~
```

- Enclose the resulting string in parentheses

The resulting string is used in a template regular expression to give the element and list's desired semantics. For example

```
<includehosts>example.com example.org </includehosts>
```

becomes

```
<owl:Restriction>
  <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
  <owl:hasValue  rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/</owl:hasValue>
</owl:Restriction>
```

4.2 POWDER and POWDER-BASE IRI Set Semantics

POWDER's use cases involve information resources available on the Web, identified by IRIs containing host names, directory paths, IP addresses, port numbers, and so on. To make it as easy as possible to create IRI sets we define a series of IRI constraints in the Grouping of Resources document [GROUP]. These all receive semantics through being mapped to `includeregex` and `excluderegex` elements in POWDER-BASE.

Re-visiting the example given in the previous section, the POWDER element

```
<iriset>
  <includehosts>example.com example.org</includehosts>
</iriset>
```

is expressed in POWDER-BASE as:

```
<iriset>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(:([0-9]+))?\/</includeregex>
</iriset>
```

IRIs are always interpreted as strings, even if they include numerical parts such as ports and IP numbers as shown in the following example:

Example 4-1: The POWDER and POWDER-BASE Encoding of an Example IRI Set

POWDER: [XML]

```
<iriset>
  <includehosts>example.com example.org</includehosts>
  <includeports>80 8080 8081 8082</includeports>
</iriset>
```

POWDER-BASE: [XML]

```
<iriset>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/</includeregex>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+\:(80|8080|8081|8082)\/</includeregex>
</iriset>
```

This approach is applied to several of the POWDER IRI set elements. The following table shows these and their associated template regular expressions. In each case, *var* means the value of the POWDER element after processing as defined in Section 4.1.

POWDER IRI Constraint (`include/exclude...`)	POWDER-BASE Regular Expression (used in `includeregex/excluderegex`)
schemes	^***var***\:\/\/
hosts	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?***var***(\:([0-9]+))?\/
ports	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+\:***var***\/
exactpaths	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?***var***($|\?|\#)
pathcontains	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/[^\?\#]***var***[^\?\#]*[\?\#]?
pathstartswith	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?***var***
pathendswith	\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/[^\?\#]***var***($|\?|\#)
resources	^***var***$

Table 3. Template regular expressions for IRI constraints that take a white space separated list of values.

Note that the Grouping of Resources document [GROUP] sets out a data processing and canonicalization process that must be followed. This has particular implications for the `ihost` component of IRIs as the regular expressions must match their IDNA ASCII representation [RFC 3490], whereas for the rest of the IRI components the regular expressions must match their Unicode representations. Furthermore,

where the port number is constrained, default port numbers for the relevant scheme must be taken into account.

Two further pairs of IRI set constraints defined in the Grouping of Resources document undergo additional processing when transformed from POWDER to POWDER-BASE: `includequerycontains` and `includeiripattern` (and their 'exclude' counterparts). Each of these maps to multiple elements in the POWDER-BASE document as shown in the following subsections.

4.2.1 Include/Exclude Query Contains Semantics

`includequerycontains` and `excludequerycontains` take a single value, not a white space separated list of values. Furthermore, an attribute `delimiter` takes a single character that delimits the name/value pairs in the query string. If no such attribute is set, the ampersand (`&`) character is used as the default. To transform these elements from POWDER to POWDER-BASE regular expressions the following steps are carried out:

- Split the supplied value at the delimiter, ***d***, dropping that character in the process
- For each resulting sub string, ***q***, create an `includeregex` or `excluderegex` as appropriate using the following regular expression template:

  `\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/[^\?\#]*\?([^\#]*`***d***`)?`***q***`(`***d***`|$)`

This transformation is exemplified below.

Example 4-2: The POWDER and POWDER-BASE Encoding of an Example IRI Set including an `includequerycontains` Constraint

POWDER: [XML]

```
<iriset>
  <includehosts>example.org</includehosts>
  <includequerycontains>id=123456&group=abcdefg</includequerycontains>
</iriset>
```

POWDER-BASE: [XML]

```
<iriset>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.org)(\:([0-9]+))?\/</includeregex>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/[^\?\#]*\?([^\#]*\&)?id=123456(\&|$)</includeregex>
  <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/[^\?\#]*\?([^\#]*\&)?group=abcdefg(\&|$)</includeregex>
</iriset>
```

4.2.2 Include/Exclude IRI Pattern Semantics

`includeiripattern` and `excludeiripattern` also take a single value, not a white space separated list of values, and generate `includeregex` and `excluderegex` elements in POWDER-BASE as follows:

- Match the value given against this regular expression:

  `(([^:/\?\.]+):)?(//)?([^:/\?#@]+)(:([0-9]+))?`

  from which $2 is a constraint on the scheme, $4 is a constraint on the host and $6 is a constraint on the port (the host is always constrained, $2 and $6 may be empty).
- Let the value of $2 be *s*, $4 be *h* and $6 be *p*
- If *s* is empty then let *s* be `[A-Za-z]+`
- If *p* is empty then let *p* be `(\:[0-9]+)?`, else let *p* be `\:`***p***
- If *h* is exactly * then let *h* be ([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+
- Else if *h* matches the regular expression `^\*\.(.*)` then let *h* be `([^\:\/\?\#\@]+\.)+$1`
- Else let *h* be `([^\:\/\?\#\@]+\.)*`***h***
- Create the following element in the POWDER-BASE document: `<includeregex>^`***s***`\:\/\/`***hp***`</includeregex>`

The following example shows these steps.

Example 4-3: The POWDER and POWDER-BASE Encoding of an Example IRI Set including an `includeiripattern` Constraint

POWDER: [XML]

```
<iriset>
  <includeiripattern>http://*.example.org:8080</includeiripattern>
</iriset>
```

POWDER-BASE: [XML]

```
<iriset>
  <includeregex>^http\:\/\/([^\:\/\?\#\@]+\.)+example.org:8080</includeregex>
</iriset>
```

Incidentally, the IRI set defined here is 'all resources on all subdomains of example.org (but not on example.org) accessed via HTTP through port 8080.'

4.3 POWDER-S IRI Set Semantics

Providing OWL/RDF semantics for `iriset` elements is not directly possible, since RDF does not provide any means for accessing or manipulating the string representation of an IRI.

POWDER-S uses two OWL DatatypeProperties (`wdrs:matchesregex` and `wdrs:notmatchesregex`) to relate resources to regular expressions which that resource matches. While POWDER-S uses OWL classes to group resources, any engine determining if a resource belonged in one of these OWL classes would need to be able to test a resource against a regular expression.

```
wdrs:matchesregex rdf:type owl:DatatypeProperty .
wdrs:matchesregex rdfs:range xsd:string .
wdrs:notmatchesregex rdf:type owl:DatatypeProperty .
wdrs:notmatchesregex rdfs:range xsd:string .
```

We further stipulate that <*x*, *reg*> is in IEXT(I(wdrs:matchesregex)) if and only if:

- *reg* conforms with regular expression syntax, AND
- there exist literal *sss*^^xsd:string and URI reference *uuu* such that:
  - *uuu* and *sss* satisfy the 1:1 correspondence between URI refrences and their string representation, as specified in Section 6.4 of the RDF Concepts document [URIREF].
  - *sss* matches the regular expression *reg*, AND
  - I(*uuu*)=*x*

and that <*x*, *reg*> is in IEXT(I(wdrs:notmatchesregex)) if and only if:

- *reg* conforms with regular expression syntax, AND
- there exist literal *sss*^^xsd:string and URI reference *uuu* such that:
  - *uuu* and *sss* satisfy the 1:1 correspondence between URI refrences and their string representation, as specified in Section 6.4 of the RDF Concepts document [URIREF].
  - *sss* does not match the regular expression *reg*, AND

- I(*uuu*)=*x*

It is now possible to express `includeregex` and `excluderegex` as `owl:hasValue` restrictions [OWL] on `wdrs:matchesregex` and `wdrs:notmatchesregex` respectively and build up an OWL Class to represent the IRI set in the POWDER-S encoding. Furthermore, asserting a `rdfs:subClassOf` relationship between an *iriset* and a *descriptorset* expresses the claim that the resources in the former are described by the latter.

The following example takes a complete example POWDER document through POWDER-BASE to POWDER-S. Note that the only change from POWDER to POWDER-BASE is in the elements within the IRI set.

Example 4-4: The Full Transformation of an Example from POWDER Through POWDER-BASE to POWDER-S

POWDER [XML]

```
<?xml version="1.0"?>
<powder xmlns="http://www.w3.org/2007/05/powder#"
        xmlns:ex="http://example.org/vocab#">

  <attribution>
    <issuedby src="http://authority.example.org/company.rdf#me" />
    <issued>2007-12-14T00:00:00</issued>
  </attribution>

  <dr>
    <iriset>
     <includehosts>example.com example.org</includehosts>
     <excludeports>8080 8081 8082</excludeports>
   </iriset>
   <descriptorset>
    <ex:color>red</ex:color>
    <ex:shape>square</ex:shape>
    <displaytext>Everything on example.org and example.com is red and square</displaytext>
    <displayicon src="http://example.org/icon.png" />
  </descriptorset>
 </dr>

</powder>
```

POWDER-BASE [XML]

```
<?xml version="1.0"?>
<powder xmlns="http://www.w3.org/2007/05/powder#"
        xmlns:ex="http://example.org/vocab#">

  <attribution>
    <issuedby src="http://authority.example.org/company.rdf#me" />
    <issued>2007-12-14T00:00:00</issued>
  </attribution>

  <dr>
    <iriset>
     <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/</includeregex>
     <excluderegex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+\:(8080|8081|8082)\/</excluderegex>
   </iriset>
   <descriptorset>
    <ex:color>red</ex:color>
    <ex:shape>square</ex:shape>
    <displaytext>Everything on example.org and example.com is red and square</displaytext>
    <displayicon src="http://example.org/icon.png" />
  </descriptorset>
 </dr>

</powder>
```

POWDER-S [RDF/XML,TURTLE]

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:wdrs="http://www.w3.org/2007/05/powder-s#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:ex="http://example.org/vocab#">

   <owl:Ontology rdf:about="">
    <wdrs:issuedby rdf:resource="http://authority.example.org/company.rdf#me" />
    <wdrs:issued>2007-12-14</wdrs:issued>
  </owl:Ontology>

  <owl:Class rdf:nodeID="iriset_1">
    <owl:equivalentClass>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
            <owl:hasValue  rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/</ow
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#notmatchesregex" />
            <owl:hasValue  rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+\:(8080|8081|8082)\/</owl:hasVal
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </owl:equivalentClass>
  </owl:Class>

  <owl:Class rdf:nodeID="descriptorset_1">
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://example.org/vocab#color" />
            <owl:hasValue>red</owl:hasValue>
         </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://example.org/vocab#shape" />
            <owl:hasValue>square</owl:hasValue>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
    <wdrs:text>Everything on example.org and example.com is red and square</wdrs:text>
    <wdrs:logo rdf:resource="http://example.org/icon.png" />
  </owl:Class>

  <owl:Class rdf:nodeID="iriset_1">
    <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
  </owl:Class>

</rdf:RDF>
```

It should be noted that `excluderegex` is expressed as a `wdrs:notmatchesregex` restriction as opposed to the complement of a `wdrs:matchesregex` restriction for the following reason: according to OWL open-world semantics, the absence of a `wdrs:matchesregex` triple does not entail membership in the complement of a `wdrs:matchesregex` restriction. Furthermore, it is practically impossible to enumerate the possible regular expressions and use `owl:DataRange` to locally close the interpretation and be able to infer membership in the complement of `hasValue` restrictions.

Similarly, the union of `wdrs:notmatchesregex` restrictions is used to define the complements of iriset classes, for the purpose of expressing ordered lists of DRs (see Section 3.1 above). In this manner, the `iriset_1_not` and `iriset_2_not` classes of Example 3-8 are defined as follows:

Example 3-8b: The POWDER-S Encoding of the iriset_1_not and iriset_2_not classes of Example 3-8 [RDF/XML,TURTLE]

```
<owl:Class rdf:nodeID="iriset_1_not">
```

```
  <owl:equivalentClass>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#notmatchesregex" />
          <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com)(:([0-9]+))?\/</owl:hasValue>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#notmatchesregex" />
          <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/foo</owl:hasValue>
        </owl:Restriction>
      </owl:unionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:nodeID="iriset_2_not">
  <owl:equivalentClass>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#notmatchesregex" />
          <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com)(:([0-9]+))?\/</owl:hasValue>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#notmatchesregex" />
          <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?\/bar</owl:hasValue>
        </owl:Restriction>
      </owl:unionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

Software can distinguish those RDF graphs to which the extended semantics apply by testing for the appearance of either the `wdrs:matchesregex` or the `wdrs:notmatchesregex` resource as the object of a triple. For instance, in Example 4-4 the following class description suffices to recognize a document that uses the semantic extension:

```
<owl:Restriction>
  <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
  <owl:hasValue  rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/</owl:hasValue>
</owl:Restriction>
```

4.4 Direct Descriptions

POWDER and, consequently, POWDER-BASE documents might include `descriptorset` elements that are not inside a `dr` element but directly subsumed by the document's root. Such descriptions are not meant to be implicitly applied to any IRI groups, but are only made available by explicit reference by resources, as explained in Section 2.5 of the Description Resources document [DR].

In POWDER-S, such descriptions are translated into classes, but no subsumption of an IRI group is asserted, as shown in Example 4-5. Note, however, that as shown in Section 3, each derived OWL class is given an `rdf:ID` equivalent to the `xml:id` in the original POWDER document, not an `rdf:nodeID`, so that it can be referred to from outside.

Example 4-5: The Semantics of Direct Description Elements

POWDER [XML]

```
<descriptorset xml:id="square">
  <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
  <ex:shape>square</ex:shape>
</descriptorset>

<descriptorset xml:id="round">
  <ex:finish rdf:resource="http://example.org/vocab#matt"/>
  <ex:shape>round</ex:shape>
</descriptorset>

<descriptorset xml:id="hexagonal">
  <ex:finish rdf:resource="http://example.org/vocab#eggshell"/>
  <ex:shape>hexagonal</ex:shape>
</descriptorset>
```

POWDER-S [RDF/XML,TURTLE]

```
<owl:Class rdf:ID="square">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
          <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
          <owl:hasValue>square</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="round">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
          <owl:hasValue rdf:resource="http://example.org/vocab#matt"/>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
          <owl:hasValue>round</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="hexagonal">
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
          <owl:hasValue rdf:resource="http://example.org/vocab#eggshell"/>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
          <owl:hasValue>hexagonal</owl:hasValue>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>
```

4.5 Semantics of `abouthosts` and `aboutregex`

The value of `abouthosts` is a whitespace-separated list of hosts. This list receives identical semantics to the value of the `includehosts` grouping element. In consequence, the transformation from POWDER to POWDER-BASE transforms the `abouthosts` elements into an `aboutregex` element using the same processing steps as for transforming `includehosts` into `includeregex`, as described in Section 4.2 above.

The `aboutregex` element of POWDER-BASE documents sets an outer limit on the resources described by the DRs within the document. That is to say, it restricts the resources that may receive a description not only implicitly (via subsumption by an IRI set) but also explicitly as described in Section 2.5 of the Description Resources document [DR].

In order to capture these semantics, the POWDER-BASE to POWDER-S transformation must add an implicit restriction to all descriptor sets in the document, effectively subsuming all resource classes created by `descriptorset` elements under the resource class that is created by the `aboutregex` element. In this manner, the implicit or explicit assignment of a description to a resource (cf. Section 2.5 of the Description Resources document [DR]) will create an inconsistency if the resource lies outside the `aboutregex` class.

It should be noted that `iriset` classes are *not* implicitly intersected with the `aboutregex` class, and it is the responsibility of the POWDER document author to ensure that all IRI sets in the document are within the scope defined by `abouthosts/aboutregex`.

This process is demonstrated by Example 4.7. Notice that in the POWDER-S document, each descriptor class is intersected with the 'aboutset' (lines 31, 51 and 63). A logical inconsistency will arise (i.e. an error) if an IRI set is not a subset of the aboutset class.

Example 4-6: The Semantics of `abouthosts`

POWDER Document [XML]

```
 1  <attribution>
 2    <issuedby src="http://authority.example.org/company.rdf#me" />
 3    <abouthosts>example.org example.com</abouthosts>
 4  </attribution>

 5  <dr>
 6    <iriset>
 7      <includehosts>square.example.org</includehosts>
 8    </iriset>
 9    <descriptorset>
10     <ex:shape>square</ex:shape>
11   </descriptorset>
12 </dr>

13 <dr>
14   <iriset>
15     <includehosts>round.example.com</includehosts>
16   </iriset>
17   <descriptorset>
18     <ex:shape>round</ex:shape>
19   </descriptorset>
20 </dr>

21 <descriptorset xml:id="silver">
22   <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
23   <ex:shape>square</ex:shape>
24 </descriptorset>
```

POWDER-BASE Document [XML]

```
 1  <attribution>
 2    <maker ref="http://authority.example.org/company.rdf#me" />
 3    <aboutregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.org|example\.com)(\:([0-9]+))?\/</aboutregex>
 4  </attribution>

 5  <dr>
 6    <iriset>
 7      <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(square\.example\.org)(\:([0-9]+))?\/</includeregex>
 8    </iriset>
 9    <descriptorset>
10     <ex:shape>square</ex:shape>
11   </descriptorset>
12 </dr>

13 <dr>
14   <iriset>
15     <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(round\.example\.com)(\:([0-9]+))?\/</includeregex>
16   </iriset>

17   <descriptorset>
18     <ex:shape>round</ex:shape>
19   </descriptorset>
20 </dr>

21 <descriptorset xml:id="silver">
22   <ex:finish rdf:resource="http://example.org/vocab#shiny"/>
23   <ex:shape>square</ex:shape>
24 </descriptorset>
```

POWDER-S Document [RDF/XML,TURTLE]

```
 1  <?xml version="1.0"?>
 2  <rdf:RDF
 3     xmlns:wdrs="http://www.w3.org/2007/05/powder-s#"
 4     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 5     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 6     xmlns:owl="http://www.w3.org/2002/07/owl#"
 7     xmlns:ex="http://example.org/vocab#">

 8     <owl:Ontology rdf:about="">
 9      <wdrs:issuedby rdf:resource="http://authority.example.org/company.rdf#me" />
10      <wdrs:issued>2007-12-14</wdrs:issued>
11    </owl:Ontology>

12    <owl:Class rdf:nodeID="aboutset"> <!-- From the abouthosts element -->
13      <owl:equivalentClass>
14        <owl:Class>
15          <owl:intersectionOf rdf:parseType="Collection">
16            <owl:Restriction>
17              <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
18              <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.org|example\.net)(\:([0-9]+))?\/</owl
19            </owl:Restriction>
20          </owl:intersectionOf>
21        </owl:Class>
22      </owl:equivalentClass>
23    </owl:Class>

24    <owl:Class rdf:nodeID="iriset_1">
25      <owl:equivalentClass>
26        <owl:Class>
27          <owl:intersectionOf rdf:parseType="Collection">
28            <owl:Restriction>
29              <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
30              <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(square\.example\.org)(\:([0-9]+))?\/</owl:hasV
31           </owl:Restriction>
32         </owl:intersectionOf>
33        </owl:Class>
34      </owl:equivalentClass>
35    </owl:Class>

36    <owl:Class rdf:nodeID="descriptorset_1">
37      <rdfs:subClassOf>
38        <owl:Class>
39          <owl:intersectionOf rdf:parseType="Collection">
40            <owl:Class rdf:nodeID="aboutset"/>
41            <owl:Restriction>
42              <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
43              <owl:hasValue>square</owl:hasValue>
44            </owl:Restriction>
45          </owl:intersectionOf>
46        </owl:Class>
47      </rdfs:subClassOf>
48    </owl:Class>

49    <owl:Class rdf:nodeID="iriset_1">
50      <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
51    </owl:Class>

52    <owl:Class rdf:nodeID="iriset_2">
53      <owl:equivalentClass>
54        <owl:Class>
```

```
55          <owl:intersectionOf rdf:parseType="Collection">
56            <owl:Restriction>
57              <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#matchesregex" />
58              <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(round\.example\.com)(\:([0-9]+))?\/</owl:hasVa
59            </owl:Restriction>
60          </owl:intersectionOf>
61        </owl:Class>
62      </owl:equivalentClass>
63    </owl:Class>

64    <owl:Class rdf:nodeID="descriptorset_2">
65      <rdfs:subClassOf>
66        <owl:Class>
67          <owl:intersectionOf rdf:parseType="Collection">
68            <owl:Class rdf:nodeID="aboutset"/>
69            <owl:Restriction>
70              <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
71              <owl:hasValue>round</owl:hasValue>
72            </owl:Restriction>
73          </owl:intersectionOf>
74        </owl:Class>
75      </rdfs:subClassOf>
76    </owl:Class>

77    <owl:Class rdf:nodeID="iriset_2">
78      <rdfs:subClassOf rdf:nodeID="descriptorset_2"/>
79    </owl:Class>

80    <owl:Class rdf:ID="silver">
81      <rdfs:subClassOf>
82        <owl:Class>
83          <owl:intersectionOf rdf:parseType="Collection">
84            <owl:Class rdf:nodeID="aboutset"/>
85            <owl:Restriction>
86              <owl:onProperty rdf:resource="http://example.org/vocab#finish"/>
87              <owl:hasValue rdf:resource="http://example.org/vocab#shiny"/>
88            </owl:Restriction>
89            <owl:Restriction>
90              <owl:onProperty rdf:resource="http://example.org/vocab#shape"/>
91              <owl:hasValue>square</owl:hasValue>
92            </owl:Restriction>
93          </owl:intersectionOf>
94        </owl:Class>
95      </rdfs:subClassOf>
96    </owl:Class>

97 </rdf:RDF>
```

As discussed in Section 3.2.3, a DR MAY refer to `descriptorset` elements in other POWDER documents. In such a situation, although it is not possible using XSLT (the technology used to effect the POWDER transforms) to generate `aboutregex` elements for POWDER-BASE as shown in the previous example, a conformant POWDER Processor MUST take account of `abouthosts` elements in both documents. The `abouthosts` element is designed to place an outer limit on the scope of any description in a POWDER document so that publishers descriptions retain effective control over the assertions made using their descriptions, even when such assertions are not attributed to them. For example, publishers of descriptions can use `abouthosts` to state that their descriptions are only meaningful for a particular set of domains, and may not be used to describe anything else.

4.6 POWDER-BASE IRI Set Semantics in OWL 2 (Informative)

At the time of this writing, the OWL 2 [OWL2] working draft provides for user-defined datatypes, using the restriction facet mechanism in XSD 1.1 [XSD]. As this includes regular expression patterns, it is possible to translate POWDER into OWL 2 requiring a simpler extension than the one in Section 4.3.

More specifically, we extend RDF semantics [RDF-SEMANTICS] with a functional datatype property `hasIRI`, defined as:

```
wdrs:hasIRI rdf:type owl:DatatypeProperty .
wdrs:hasIRI rdf:type owl:FunctionalProperty .
wdrs:hasIRI rdfs:range xsd:anyURI .
```

and the further stipulation that:

<*x*, *sss*> is in IEXT(I(wdrs:hasIRI)) if and only if there exists URI reference *uuu* such that:

- *uuu* and *sss* satisfy the 1:1 correspondence between URI refrences and their string representation, as specified in Section 6.4 of the RDF Concepts document [URIREF], AND
- I(*uuu*)=*x*.

Such an extension makes it possible to provide semantics to `iriset` by constructing an RDF datatype for each `iriset` and restricting the values of `hasIRI` to this datatype's range. In this manner, the POWDER-S translation of Example 4-4 becomes as shown in Example 4-7, where `iriset_1` is a class of abstract resources, the concrete IRI string of which is within a user-defined datatype (lines 20-29 and 37-46).

Example 4-7: The POWDER-S Encoding of Example 4-4 With User-defined Datatypes [RDF/XML,TURTLE]

```
 1  <?xml version="1.0"?>
 2  <rdf:RDF
 3    xmlns:wdrs="http://www.w3.org/2007/05/powder-s#"
 4    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 5    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 6    xmlns:xsd="http://www.w3.org/2001/XMLSchema-datatypes#"
 7    xmlns:owl="http://www.w3.org/2002/07/owl#"
 8   xmlns:ex="http://example.org/vocab#">

 9   <owl:Ontology rdf:about="">
10     <wdrs:issuedby rdf:resource="http://authority.example.org/company.rdf#me" />
11     <wdrs:issued>2007-12-14</wdrs:issued>
12   </owl:Ontology>

13   <owl:Class rdf:nodeID="iriset_1">
14     <owl:equivalentClass>
15       <owl:Class>
16         <owl:intersectionOf rdf:parseType="Collection">
17           <owl:Restriction>
18             <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#hasIRI" />
19             <owl:someValuesFrom>
20               <rdfs:Datatype>
21                 <owl:onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema-datatypes#string"/>
22                 <owl:withRestrictions rdf:parseType="Collection">
23                   <rdf:Description>
24                     <xsd:pattern rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">
25                       \:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com|example\.org)(\:([0-9]+))?\/
26                     </xsd:pattern>
27                   </rdf:Description>
28                 </owl:withRestrictions>
29               </rdfs:Datatype>
30             </owl:someValuesFrom>
31           </owl:Restriction>
32           <owl:Class>
33             <owl:complementOf>
34               <owl:Restriction>
35                 <owl:onProperty rdf:resource="http://www.w3.org/2007/05/powder-s#hasIRI" />
36               <owl:someValuesFrom>
37                 <rdfs:Datatype>
38                   <owl:onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema-datatypes#string"/>
39                   <owl:withRestrictions rdf:parseType="Collection">
40                     <rdf:Description>
41                       <xsd:pattern rdf:datatype="http://www.w3.org/2001/XMLSchema-datatypes#string">
42                         \:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)*[^\:\/\?\#\@]+\:(8080|8081|8082)\/
43                         </xsd:pattern>
44                       </rdf:Description>
45                     </owl:withRestrictions>
46                   </rdfs:Datatype>
```

```
                </owl:someValuesFrom>
              </owl:Restriction>
            </owl:complementOf>
          </owl:Class>
        </owl:intersectionOf>
      </owl:Class>
    </owl:equivalentClass>
  </owl:Class>

  <owl:Class rdf:nodeID="descriptorset_1">
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://example.org/vocab#color" />
            <owl:hasValue>red</owl:hasValue>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="http://example.org/vocab#shape" />
            <owl:hasValue>square</owl:hasValue>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
    <wdrs:text>Everything on example.org and example.com is red and square</wdrs:text>
    <wdrs:logo rdf:resource="http://example.org/icon.png" />
  </owl:Class>

  <owl:Class rdf:nodeID="iriset_1">
    <rdfs:subClassOf rdf:nodeID="descriptorset_1"/>
  </owl:Class>

</rdf:RDF>
```

Note how `hasIRI`, being axiomatically functional, makes it possible to use `owl:complementOf` to express the semantics of `excluderegex`. This should be contrasted to the more complex solution of the two complementary properties adopted in Section 4.3.

5 POWDER Processor Semantics

The operational semantics of the POWDER processor are specified in Section 3 of the Description Resources document [DR], where the core functionality of the POWDER processor is defined as the implementation of a function `describe(u)`, which accepts as input an IRI and returns an RDF document describing the resource denoted by the IRI.

Notwithstanding the processing and syntactic transformations required to serialize and deliver an RDF document, the triples present in the document can be formally specified using SPARQL queries with exactly two variables in the `SELECT` clause (predicate and object, the subject being the resource denoted by *u*.)

More specifically, a POWDER processor must return:

1. All the RDF triples with *u* as the subject:
   `SELECT ?p ?o WHERE { <u> ?p ?o }`
2. All annotation triples known about the descriptorset itself:
   `SELECT ?p ?o WHERE { <u> rdf:type ?ds . ?ds ?p ?o . ?p rdf:type owl:AnnotationProperty . }`
3. Any `rdf:seeAlso`, `rdfs:label`, and `rdfs:comment` statements about the descriptorset itself:
   `SELECT ?o WHERE { <u> rdf:type ?ds . ?ds rdfs:seeAlso ?o . }`
   `SELECT ?o WHERE { <u> rdf:type ?ds . ?ds rdfs:label ?o . }`
   `SELECT ?o WHERE { <u> rdf:type ?ds . ?ds rdfs:comment ?o . }`
   noting that only the object is returned, since the property is hard-wired in the query.

In addition, the POWDER processor may return a triple with a *u* as subject, `wdrs:describedby` as property, and the IRI of the POWDER document from which the description was obtained as object. If more than one POWDER document is the source of the description then each should be the object *o* of a separate

`<u> wdrs:describedby <o>`

triple. If the processor can find no information about the candidate resource within the data available at the time of the request, then it returns a single triple with *u* as subject, `wdrs:notknownto` as property, and the IRI of the POWDER processor as object. This extra-logical behavior is handled by the serialization and delivery shell, based on the SPARQL query's result set being empty or not.

6 Acknowledgements

The Working Group would like to thank Jeremy Carroll for his substantial contribution to the development of the semantics of POWDER. Michael Schneider provided valuable assistance and expertise concerning the definition of the semantic extension.

7 References

7.1 Normative References

[GRDDL]
Gleaning Resource Descriptions from Dialects of Languages (GRDDL), W3C Recommendation 11 September 2007. D. Connolly. This document is at http://www.w3.org/TR/grddl/
[DR]
Protocol for Web Description Resources (POWDER): Description Resources, P. Archer, A. Perego, K. Smith. This document is at http://www.w3.org/TR/powder-dr/
[GROUP]
Protocol for Web Description Resources (POWDER): Grouping of Resources, A. Perego, P. Archer This document is at http://www.w3.org/TR/powder-grouping/
XML dateTime
XML Schema Part 2: Datatypes Second Edition W3C Recommendation 28 October 2004. P. Biron, A. Malhotra (Eds). This document is at http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/#dateTime
[OWL]
OWL Web Ontology Language, Semantics and Abstract Syntax, W3C Recommendation 10 February 2004. P. F. Patel-Schneider, I. Horrocks. This document is at http://www.w3.org/TR/owl-semantics/
[XQXP]
XQuery 1.0 and XPath 2.0 Functions and Operators, A. Malhotra, J. Melton, N. Walsh. W3C Recommendation 23 January 2007. This document is at http://www.w3.org/TR/xpath-functions/
[RFC2119]
Key words for use in RFCs to Indicate Requirement Levels, RFC 2119, S. Bradner. This document is at http://tools.ietf.org/html/rfc2119
[RDF-SEMANTICS]
RDF Semantics, W3C Recommendation 10 February 2004, P. Hayes. The key text is in Section 1.3. This document is at http://www.w3.org/TR/2004/REC-rdf-mt-20040210/
[URIREF]
RDF Concepts, W3C Recommendation 10 February 2004, G. Klyne and J. Carrol. The key text is in Section 6.4. This document is at http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/
[RFC 3490]
Internationalizing Domain Names in Applications (IDNA). P. Faltstrom, P. Hoffman, A. Costello. This document is at http://www.faqs.org/rfcs/rfc3490.html.

7.2 Informative References

[WEBARCH]
Architecture of the World Wide Web, Volume One, Section 2.2. W3C Recommendation 15 December 2004. I. Jacobs, N. Walsh. This document is at http://www.w3.org/TR/webarch/#id-resources
[XSLT]
XSL Transformations (XSLT), W3C Recommendation 16 November 1999. J. Clark. This document is at http://www.w3.org/TR/xslt
[XSLT2]
XSL Transformations (XSLT) Version 2.0, W3C Recommendation 23 January 2007. M. Kay. This document is at http://www.w3.org/TR/xslt20/
[USECASES]
POWDER: Use Cases and Requirements W3C Working Group Note 31 October 2007, P. Archer . This document is at http://www.w3.org/TR/powder-use-cases/
[PRIMER]
Protocol for Web Description Resources (POWDER): Primer 2008, K. Scheppe, D. Pentecost. This document is at http://www.w3.org/TR/powder-primer/)
[TESTS]
Protocol for Web Description Resources (POWDER): Test Suite 2008, A. Kukurikos. This document is at http://www.w3.org/TR/powder-test/
[WDR]
Protocol for Web Description Resources (POWDER): Web Description Resources XML Schema (WDR), A. Perego, K. Smith. This document is at http://www.w3.org/2007/05/powder
[WDRS]
Protocol for Web Description Resources (POWDER): POWDER-S Vocabulary (WDRS), A. Perego, P. Archer. This document is at http://www.w3.org/2007/05/powder-s

[WDR2B]
Protocol for Web Description Resources (POWDER): POWDER to POWDER-BASE XSLT 2008, K. Smith. (URI TBC)
[B2S]
Protocol for Web Description Resources (POWDER): POWDER-BASE to POWDER-S XSLT 2008, K. Smith (URI TBC)
[XSD]
W3C XML Schema Definition Language 1.1, Part 2: Datatypes W3C Working Draft 30 January 2009. D Peterson, S. Gao, A Malhotra, C M Sperberg-McQueen, and H S Thompson. This document is at http://www.w3.org/TR/2009/WD-xmlschema11-2-20090130/
[N3]
Notation3 (N3): A readable RDF syntax, T. Berners-Lee, D. Connolly. This document is at http://www.w3.org/TeamSubmission/n3/
[TTL]
Turtle - Terse RDF Triple Language, D. Beckett, T. Berners-Lee. This document is at http://www.w3.org/TeamSubmission/turtle/
[DC]
Dublin Core Metadata Initiative Terms. See http://dublincore.org/documents/dcmi-terms/
[FOAF]
FOAF Vocabulary Specification 0.9 Namespace Document 24 May 2007, D. Brickley, L. Miller. This document is at http://xmlns.com/foaf/spec/
[OWL2]
OWL 2 Web Ontology Language: Document Overview. W3C OWL Working Group. This document is at http://www.w3.org/TR/owl2-overview/

8 Change Log

8.1 Changes since First Public Working Draft

- Status section updated
- Mention of separate POWDER-BASE namespace removed (a legacy from pre-publication draft)
- `<maker>` and `foaf:maker` replaced by `<issuedby>` and `wdrs:issuedby` in all examples. This is defined in the WDRS vocabulary as a sub property of both `foaf:maker` and `dcterms:creator` so that Agent classes from both vocabularies may be used. Support for both now included. See e-mail thread.
- Sentence added to clarify that the `dcterms/foaf:Agent` class can be included directly
- Where previous examples have had <ex:color rdf:resource="...#red" /> this has been changed to <ex:finish rdf:resource="...#shiny" /> to avoid confusion following comments by Ivan Herman
- Correction of error in text following example 3-1
- Use of `rdf:nodeID` corrected throughout following comments by Masahide Kanzaki and Ivan Herman.
- Unnecessary detail elided from Example 3-8
- Line numbers added to all examples in the text following comment from Ivan Herman (see the P.S.)
- Section 3.2 substantially re-written and tidied up, introducing `typeof, seealso, label and comment` elements. Blank nodes no longer forbidden but their usage is strongly discouraged. Text flows from e-mail discussion.
- Wording of section on tags tidied up, `ref` attribute removed in favor of using `seealso`.
- Slight change in the presentation of the semantic extension in Section 4.3. This now has the fragment identifier of #SE. The semantic extension in the informative example in Section 4.6 now has the fragment identifier of #SE2
- Error in preamble to example 4-4 fixed. This referred to an extra namespace for POWDER-BASE which is no longer used.
- Section 4.4 and Example 4-6 amended slightly to state that descriptor sets not inside a DR are transformed into OWl classes with `rdf:ID` identifiers cf. `rdf:nodeID`s.
- In section 4.5, the logical inconsistency of an IRI set being outwith an abouthosts limit is made clear. Example 4-7 reworked a little to make consistent with other examples.
- Slight revision to text concerning semantics of descriptor sets referred to in external documents as it relates to `abouthosts`. Reference is now to section 3.2.3 of this document cf. the DR document.
- Use of `rdf:Description` to describe the POWDER-S document changed to `owl:Ontology` primarily to work around RDF's lack of named graphs, The ontology instance can be described using ontology headers.
- The use of `owl:complementOf` in Example 3-8 and related examples corrected following comment from Ivan Herman
- Collection removed from line 8 of POWDER-S example in Example 4-5 following comment from Ivan Herman
- Template regular expressions (Table 3 etc.) updated to support candidate resource IRIs that include user info following e-mail comment.
- Support for arbitrary RDF in the `attribution` and `descriptorset` elements flagged as a Feature at Risk - see Status section.

8.2 Changes since Last Call Working Draft

- Removed mention of null subject in Section 2.2 and aligned the in-line Example 2-1 with the external file, following e-mail comment by Peter Patel-Schneider.
- Replaced all instances of `wdrs:matchesregexp` with `wdrs:matchesregex` following e-mail comment by Peter Patel-Schneider.
- Re-phrased 2nd paragraph of Section 2.2, and also corrected the range of `wdrs:certifiedby` `wdrs:supportedby` following e-mail comment by Peter Patel-Schneider.
- Amended both formulations of the RDF extension (Sect. 4.3 and 4.6) following e-mail comment by Peter Patel-Schneider.
- Removed the arbitrary RDF feature-at-risk (Sect. 2.2 and 3.2.1).
- Fixed various typos in Sections 3, 3.1 (Ex 3-5), 3.2.1, 4.6 following e-mail comment by Michael Schneider.
- Added a short note providing the rational behind the MUST NOT in Section 3.2.1, following e-mail comment by Michael Schneider.
- Purged various left-overs of the situation before using owl:AnnotationProperty instances for attribution. Namely, replaced foaf:depiction with wdrs:logo and dcterms vocabulary with dcelements vocabulary, since foaf:depiction and dcterms specify doamins and/or ranges and cannot be an owl:AnnotationProperty instances. Also changed from using dcterms:issued to wdrs:issued, foaf:depiction to wdrs:logo and dcterms:description to wdrs:text to ensure properties of the correct types with correct domain and range restrictions are used.
- Better clarified the last paragraph of Sect. 3.2.4.
- Various changes made to reflect the positon of this document as normative wrt to the transform and the XSLTs as useful, but non-normative, aids. [1, 2, 3, 4]
- Various minor clarifications made and typos corrected following comments by Michael Schneider. [1, 2, 3, 4, 5, 6, 7, 8]
- Minor additions made to fulfil QA requirements [Introduction renamed as Introduction & Scope with additional line, Conformance Statement added]
- Handling of descriptor sets defined in external documents clarified and simplified to remove expectation of processors having knowledge of what is being pointed to, following comment from Ivan Herman. The text on this issue following Example 4-7 has been deleted.
- Added Turtle serializations for all examples, following e-mail comment by Peter Patel-Schneider.
- Added section on POWDER processor semantics, showing the SPARQL queries that retrieve the information expected from a PP.
- excluderegex demonstrated in Ex 4-4; Ex 4-5 removed; Ex 4-6 renamed to Ex 4-5, Ex 4-7 to Ex 4-6, and Ex 4-8 to Ex 4-7.
- Section 4.3 modified to define `notmatchesregex`; rationale provided inline. This also influenced ordered lists of DRs (Section 3.1).
- Added `owl:FunctionalProperty` axiom to the definition of `wdrs:hasIRI` in Section 4.6.
- Fixed minor mistakes in the examples.
- Text added to describe the transformation of the `more` element.

8.3 Changes since Second Last Call Working Draft

- Conformance statement amended following a comment from Dan Connolly
- Reference to W3C DTF replaced with reference to XML dataType following comment from Dan Connolly
- References split into normative and informative following comment from Dan Connolly. Also obsolete references deleted.
- Error corrected in Section 5 relating to the object of the wdrs:describedby property. Additional clarification added.
- Added in Section 1.1 a reference to the particular flavour of regular expressions used in this document, and in Section 4.3 a reference to RDF Semantics, following comment from Jonathan Rees.
- Added in Section 4.3 a simple test for recognizing documents that use the POWDER extension, as required by the preliminary section of the RDF Semantics document.
- Fixed minor bugs in `iripattern` transforms in Section 4.2.
- Example 3-8 has a syntactic change (no change in semantics) which makes the transform simpler to implement. Added a clarification note after Example 3-8, to avoid a possible misunderanding.
- The introductory sentence to the Semantic Extension was rephrased to refer to an owl:DatatypeProperty cf. an RDF Semantic Extension. This was only accepted after a prolonged discussion begun by Eric Prud'hommeaux (initially raised as a minor nit!).
- Slight change in the referencing of the Grouping document concerning data processing and IRI canonicalization, following input from Thomas Roessler.

8.4 Changes since Third Last Call Working Draft

- Updates in Section 4.6 reflecting changes in the OWL 2 specification, following input from Ivan Herman.
- Minor fix in Section 4.3 and Section 4.6 removing the problematic use rdf:XMLLiteral and directly referring to the URIref - string correspondence. Problem spotted and fix provided by Michael Schneider.
- Further slight change in the referencing of the Grouping document to emphasize use of Unicode in IRI strings.

8.5 Changes since Proposed Recommendation

- XML Schema namespace corrected
- Sentence added to clarify the Regular Expression Syntax used