The POWDER Protocol as Infrastructure to Serving and Compressing Semantic Data

S. Konstantopoulos
P. Archer
P. Karampiperis*
V. Karkaletsis

Institute of Informatics and Telecommunications,
NCSR 'Demokritos',
Ag. Paraskevi 15310, Athens, Greece
Fax: +30 21 06532175
E-mail: pythk@iit.demokritos.gr
*Corresponding author

Abstract: The POWDER protocol is a Semantic Web technology that takes advantage of natural groupings of URIs in order to annotate all the resources in a regular expression-delineated sub-space of the URI space. POWDER is a mechanism for accreditation, trustmarking, and resource discovery, emphasising the publishing of attributed metadata by third parties and trusted authorities. Demonstrating its versatility, it has also been deployed in unforeseen use cases, such as repository compression. In this paper we present the POWDER protocol, explain its position in the Semantic Web architecture, expose and discuss current implementations and use cases, and future directions.

Keywords: POWDER; Semantic Web; metadata publishing; triple store compression.

Reference to this paper should be made as follows: Konstantopoulos, S., Archer, P., Karampiperis, P., and Karkaletsis, V. (2012) 'The POWDER Protocol as Infrastructure to Serving and Compressing Semantic Data', *Int. J. Metadata, Semantics, and Ontologies*, Vol. X, Nos. 1/2, pp. 197–212.

Biographical notes: Stasinos Konstantopoulos received his PhD on Inductive Logic Programming and Language Technology in 2003 from University of Groningen, the Netherlands. Since 2005 he is research associate at NCSR 'Demokritos', working on knowledge representation and language technology. He participated in the POWDER W3C Working Group, where he was mainly involved in establishing the formal semantics of the POWDER specification. Together with Phil Archer, he edited the *POWDER Formal Semantics* W3C Recommendation.

Phil Archer lead the W3C Incubator Group that laid the foundation for the later POWDER Working Group which he also chaired. As well as co-editing most of the specification documents, he also created one of the reference implementations of a POWDER Processor. Having worked previously in on-line safety, later with NCSR 'Demokritos' and its commercial spin-off *i-sieve Technologies*, he has been a member of the W3C Team since February 2009 focusing at first on mobile Web issues and latterly on eGovernment.

Pythagoras P. Karampiperis, holds a Diploma and MSc in Electronics and Computer Engineering and an MSc in Operational Research, all from the Technical University of Crete, Greece. He also received a PhD on Adaptive Educational Hypermedia Systems from the Department of Digital Systems, University of Piraeus. His main scientific interests are in the areas of Technology-enhanced Learning, and Semantic Web Technologies. He is the co-author of more than 50 publications in scientific books, journals and conferences with at least 360 known citations (h-index: 10).

Vangelis Karkaletsis is Research Director at NCSR and head of the Software & Knowledge Engineering Lab (SKEL). He was coordinator of the DG-SANCO project MedIEQ on the analysis of health-related web content, and technical manager of the SIAP project QUATRO Plus on web content labeling, both of which exploited POWDER for metadata description. He led NCSR work in the POWDER working group. He is co-founder of the spin-off company *i-sieve Technologies*, which focuses on content analysis applications. He served for several years in the Board of the Hellenic Association of Artificial Intelligence.

1 Introduction

POWDER is a recently developed Semantic Web technology that takes advantage of natural groupings of URIs in order to annotate all the resources in a regular expression-delineated sub-space of the URI space.

Take, for example, the triples in Figure 1, published by the Ordnance Survey, Britain's mapping agency. Following best practices in the design of this URI, the Ordnance Survey has included a lot of information within the URIs themselves. By convention, any linked data practitioner can immediately tell from the inclusion of a path segment of `/id/` that this identifies a non information resource and that dereferencing that URI is likely to result in an HTTP *303 redirect* to a URI where `/id/` has been replaced with `doc`, returning a document describing the originally identified resource.

Furthermore, all URIs on the `data.ordnancesurvey.co.uk` domain that include consecutive path segments of `/id/postcodeunit/` identify a geographical area that is a post code unit. The Ordnance Survey has no means to publish this *knowledge* about the URI design, but instead publishes 636,928 triples like those in Figure 1, explicitly asserting type information about all postcode URIs in the U.K.

POWDER allows making explicit the *intention* behind URI structure, which goes beyond exhaustively annotating all resources in a domain as it also represents the *knowledge* that currently unknown and future resources within the domain will have the properties implied by their position in the URI structure. That is to say, that any resource, currently existing *or added in the future*, under `data.ordnancesurvey.co.uk/id/postcodeunit` *must* be a postcode.

POWDER offers the advantage that, unlike various ad-hoc schemas for providing such information, it is designed with *extendability* and *interoperability* in mind, having a semantics that is compatible with RDF and OWL tools and a mechanism for developing, and for assigning formal semantics to, extensions to the basic POWDER constructs primarily targeting URLs and the HTTP Web. Furthermore, the protocol makes provisions for attributing POWDER documents with provenance data and validity period restrictions.

These features allow for a variety of possible applications to be explored besides the fundamental functionality of annotating resources. Such applications and extensions, both currently explored and potential, include resource discovery, repository compression, caching and cache revoking, social networking and web of trust applications, library science applications, and more.

In the remainder of this paper we will first present related approaches (Section 2) and then proceed to discuss the POWDER protocol (Section 3) and its formal semantics (Section 4). We then discuss and compare alternative approaches to implementing it (Section 5), present current POWDER applications (Sections 6 and 7), and conclude (Section 8).

2 Related Formalisms

The use cases for POWDER are centred on machine readable trustmarks, disambiguation of subject matter, declarations of accessibility compliance, mobile-friendliness, on-line safety, and so on. These all share the need to make statements about Web sites or parts of Web sites, which is a more generally recurring need.

Ad-hoc approaches, such as `robots.txt` files, can to some extent achieve this and have culminated in a general protocol for adding a variety of site-wide metadata (Nottingham and Hammer-Lahav, 2010). However, POWDER was designed not only to provide site-wide metadata, and a discovery mechanism for it, but to create a standard that could be seamlessly integrated into, and be a useful addition to, the Semantic Web technology stack.

POWDER can be summed up most succinctly as a solution to the `rdf:aboutEach` problem, as exposed by, among others, Gibbins and Shadbolt (2010): in the original RDF specification, `rdf:aboutEach` and `rdf:aboutEachPrefix` were *distributive referents* that allowed a predicate and object to be applied to multiple subjects (Lassila and Swick, 1999, Section 3.3).

During the development of the subsequent specification suite (Manola and Miller, 2004), it was shown that that the semantics of the `rdf:aboutEach` property were not at all clear and that parsers would need to be able retain information about one or more triples when parsing others in case new information came to light about what had already been processed.[1] As a consequence, `rdf:aboutEach` was removed from the specification, a decision that was greeted with loud cheers.[2]

The origins of the distributive referents themselves are a reflection of RDF's early genesis as a development of *the Platform for Internet Content Selection* (PICS, Lassila 1997). PICS was a labelling system for Web sites that allowed ratings to be applied to URLs that matched a given prefix, such as `http://www.example.com`. However, such a simple string-based match would not cover `http://example.com` (note the lack of 'www'), any of example.com's other sub-domains, or any resource served using HTTPS. A modern dynamic Web page, such as a news portal homepage, will easily draw content from upwards of 20 different domains and sub-domains so that labelling all the content using PICS would be very challenging. One of the main motivations for the development of POWDER was to make it possible to apply properties to any group if URIs including, in particular, `*.example.com` — i.e., any resource on example.com or any of its sub-domains but without inadvertently describing `fooexample.com`.

Furthermore, the recent rapid uptake of the publication of large-scale Linked Open Data has led W3C to publish *Cool URIs* (Sauermann and Cyganiak, 2008), best practice guidance for URI design. This underpins work on designing the URI structure of data published by, among many others, the U.K.

```
<http://data.ordnancesurvey.co.uk/id/postcodeunit/AL11AE> rdf:type
  <http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit> .
```

Figure 1 Example RDF data from the Ordnance Survey domain.

government[3] and the Ordnance Survey.[4] Cool URIs encourage encoding more complex structure in the path section of URIs than PICS is able to describe.

The old `rdf:aboutEachPrefix` would have only partially addressed this, allowing a prefix such as `http://data.ordnancesurvey.co.uk/id/postcodeunit` to be associated with postcode identifiers; it would, however, not have been able to express the two-dimensional structure of the Ordnance Survey URIs breaking resources down between identifiers vs. documents *and* between postcodes vs. counties, electoral divisions, and the other types of resources in the Ordnance Survey data.

POWDER enables engineers to retain and use the information that is embedded in carefully designed URIs without exhaustive and repetitive data publishing and processing, while at the same having the rigorous semantics that allow it to be placed in the semantic Web architecture.

3 Annotating by Name

3.1 Description Resources

The fundamental 'unit of information' in POWDER is the *Description Resource* (DR) (Archer et al., 2009). Using some syntactic sugar (and ignoring namespaces for clarity) we can re-cast the repetitive postcode triples from our Ordnance Survey example above as the POWDER document shown in Figure `fig:ex-pdr`,[5] which asserts that all IRIs on the `data.ordnancesurvey.co.uk` domain that have a path beginning with `id/postcodeunit` are of type `http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit`.[6]

Furthermore, this POWDER document makes it explicit that the assertion was made by the 'agent' described at `http://philarcher.org/foaf.rdf#me` on 1 September 2011, data that can be used by trust mechanisms to decide whether to use or ignore a POWDER document.

POWDER provides a number of shortcuts and syntactic elements designed to make it easy to create groups of URIs and make statements about them. For example, the descriptor set—the set of predicates and objects that can be applied to all the resources defined in the IRI set—makes use of POWDER's `typeof` element to express the `rdf:type` relationship without needing to declare the `rdf:` namespace. The <`includehosts/`> and <`includepathstartswith/`> elements shown in Figure 2 have complementary <`exclude/`> elements as well as ones covering path contains, path ends with, scheme, port and query string. The elements relevant to the latter, <`includequerycontains/`> and <`excludequerycontains/`>, are particularly expressive since they allow for name/value pairs to be expressed in any order so that the following two elements are equivalent:

```
<excludequerycontains>
  name1=value1&amp;name2=value2
</excludequerycontains>
<excludequerycontains>
  name2=value2&amp;name1=value1
</excludequerycontains>
```

This is a reflection of the fact that content management systems are rarely, if ever, concerned about the order in which query string parameters are expressed.

Processing of POWDER's <`iriset/`> element is done by converting the content of the elements into regular expressions. These (well tested) regular expressions are part of the POWDER specification itself such that we can re-cast the document in Figure 2 as shown in Figure 3.

All that has changed is that <`includehosts/`> and <`includepathstartswith/`> have both been replaced with the <`includeregex/`> element and the contents have been programmatically turned into regular expressions. The subset of POWDER that only uses <`includeregex/`> and <`excluderegex/`> as child elements of <`iriset/`> is known as POWDER-BASE. Given an input IRI, known as a candidate IRI, a processor can apply the regular expression(s) to decide whether or not it is a member of the IRI set and, if so, return RDF triples accordingly using the contents of the descriptor set.

A POWDER processor is an engine that can process POWDER-BASE documents and when queried about a URI responds with an RDF graph containing annotations about the URI as well as attribution for these annotations. For example, when queried with `http://data.ordnancesurvey.co.uk/id/postcodeunit/AL11AE` a POWDER Processor uses the document in Figure 3 (possibly derived from the one in Figure 2) to return the RDF graph in Fig 4.[7]

3.2 Extending POWDER

The POWDER protocol is *extensible* by means of *GRDDL transformations* (Connoly, 2007) that formally specify how to transform an extended POWDER document into a POWDER-BASE document, or, in fact, any other GRDDL-defined extension of POWDER-BASE, including POWDER. This makes it easy to develop domain-specific variants that allow POWDER

```
<powder xmlns="http://www.w3.org/2007/05/powder#">
  <attribution>
    <issuedby src="http://philarcher.org/foaf.rdf#me"/>
    <issued>2011-09-01T11:33:00</issued>
  </attribution>
  <dr>
    <iriset>
      <includehosts>data.ordnancesurvey.co.uk</includehosts>
      <includepathstartswith>/id/postcodeunit</includepathstartswith>
    </iriset>
    <descriptorset>
      <typeof src="http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit" />
    </descriptorset>
  </dr>
</powder>
```

Figure 2 POWDER document describing URIs that identify a resource of type `http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit`

```
<powder xmlns="http://www.w3.org/2007/05/powder#">
  <attribution>
    <issuedby src="http://philarcher.org/foaf.rdf#me"/>
    <issued>2011-11-20T11:33:00</issued>
  </attribution>
  <dr>
    <iriset>
      <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?
(data\.ordnancesurvey\.co\.uk)(\:([0-9]+))?\/</includeregex>
      <includeregex>\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?
(\/id\/postcodeunit)</includeregex>
    </iriset>
    <descriptorset>
      <typeof src="http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit" />
    </descriptorset>
  </dr>
</powder>
```

Figure 3 The POWDER-BASE form of the POWDER document in Figure 2.

```
<http://data.ordnancesurvey.co.uk/id/postcodeunit/AL11AE>
  rdf:type  <http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit> ;
  wdrs:describedby <http://philarcher.org/powder/postcodeunit.xml> .
<http://philarcher.org/powder/postcodeunit.xml>
  wdrs:issued 2011-09-01T11:33:00" ;
  wdrs:issuedby <http://philarcher.org/foaf.rdf#me> .
```

Figure 4 Response of a POWDER processor operating upon the document shown in Figure 2 when queried about `http://data.ordnancesurvey.co.uk/id/postcodeunit/AL11AE`

documents about this domain to be authored more succinctly.

For example, if we look again at `http://data.ordnancesurvey.co.uk/id/postcodeunit/AL11AE` we notice that the post code itself is the final path segment. Given that postcodes themselves have a two-part structure denoting a city and a postcode unit within that city, we would like to be able to directly refer to the postcode part of a URI or even to the city part of the postcode part of the URI, without cluttering our POWDER documents with the complex regular expressions necessary to achieve this.

We can easily define a new domain-specific POWDER namespace, say `http://data.ordnancesurvey.co.uk/geopdr`, and associate it with a GRDDL transform that turns domain-specific IRI elements into equivalent `<includeregex/>` elements. We can now define elements like `<includepostcodestartswith/>` or

```
<geopowder xmlns="http://data.ordnancesurvey.co.uk/geopdr#">
  <dr>
    <iriset>
      <includehosts>data.ordnancesurvey.co.uk</includehosts>
      <includepathstartswith>/id/postcodeunit</includepathstartswith>
      <includepostcodestartswith>AL</includepostcodestartswith>
    </iriset>
    <descriptorset>
      <gn:locatedIn rdf:resource="http://sws.geonames.org/2647043/" />
    </descriptorset>
  </dr>

  <dr>
    <iriset>
      <includehosts>data.ordnancesurvey.co.uk</includehosts>
      <includepathstartswith>/id/postcodeunit</includepathstartswith>
      <includecitycodes>AL1 AL2 AL3 AL4</includecitycodes>
    </iriset>
    <descriptorset>
      <gn:locatedIn rdf:resource="http://sws.geonames.org/2638867/" />
    </descriptorset>
  </dr>
</geopowder>
```

Figure 5 Description Resource asserting that all current and future postcodes beginning with ‘AL’ are/will be in the country of Hertfordshire and that all current and future postcodes where the first part is ‘AL1’ through ‘AL4’ are in the city of St. Albans.

`<includecitycodes/>` by publishing scripts, in a language like XSLT, implementing the appropriate transform into POWDER-BASE and using GRDDL to point to them.

Using these new elements we can conveniently and clearly encode that all U.K. post codes beginning with ‘AL’ are in the county of Hertfordshire (known by a GeoNames identifier of 2647043) and that all postcodes beginning with ‘AL1’ to ‘AL4’ are in the city of St. Albans, as shown in Figure 5.

In our post codes example we have built upon POWDER to retain HTTP-specific IRI set elements such as `<includehosts/>`. Although convenient in this particular extension, it should be noted that extensions can also be directly are derived from POWDER-BASE for applications that are not specific to the HTTP Web. Numbering schemes such as ISAN, ISBN, barcodes, and so on can be formulated as URIs in the `urn:` scheme and their structure can be encoded in POWDER extensions. Perego and Archer (2009, Section 3.3) demonstrate how to extend POWDER to cover ISAN numbers.

4 Formal Semantics

The logical foundations of POWDER are laid out in the *POWDER: Formal Semantics* recommendation (Konstantopoulos and Archer, 2009) and are based on a two-step transformation:

- POWDER documents, using the complete vocabulary for defining URI sets, are transformed to POWDER-BASE, equivalent XML documents that use only two kinds of `<iriset/>` elements: `<includeregexp/>` and `<excluderegexp/>`.
- POWDER-BASE documents are transformed to POWDER-S, equivalent OWL/RDF knowledge bases. In the presence of OWL inference and the POWDER-S extension, such knowledge bases infer the same annotations about resources as the original POWDER document.

The POWDER-S extension to RDF semantics circumvents the main difficulty in defining POWDER semantics: departing from the core position held by the logical foundations of Semantic Web technologies, in fact by computational logics in general, that URIs are meaningless labels to infer properties of resources premised on properties of the resources’ URIs.

In this section, we will first present the OWL/RDF semantics of POWDER documents and then proceed to present and discuss the POWDER-S extension.

4.1 DR Semantics

The core idea behind POWDER is that the relation between a IRI set and a descriptor set is that of OWL class subsumption:

Lemma 1: *The claim that the resources in the IRI set are described by the descriptor set can be expressed by* *asserting the* `rdfs:subClassOf` *relationship between the IRI set and the descriptor set.*

Consider, for example, the DR in Figure 2. If we see the <`iriset/`> element as a description of the class of resources that have URIs that match certain criteria and the <`descriptorset/`> element as a description of the 'postcode unit' class, then subsuming the former under the latter captures the intention that the resources described by the IRI set are members of 'postcode unit'.

Another important aspect of POWDER is that it facilitates the application of trust and caching mechanisms by providing for provenance and validity information in POWDER documents. More specifically, POWDER documents are required to provide authorship information and may also provide an issue date, a validity period, or any other attribution that could be useful to particular applications.

As these attributions characterize the POWDER document as a whole, they could be very naturally encoded as properties of a named RDF graph that holds the POWDER-S representation of the document. Although discussion is underway that could lead to RDF support for named graphs, at the time of preparing this article and, even more so, the POWDER Recommendation there was no formal way to separate and name graphs in a knowledge base or to assign properties to such named graphs.

As a result, it was decided to annotate the OWL knowledge base as a whole, represented by the `owl:Ontology` instance (Bechhofer et al., 2004, Section 7.2). The attribution of the POWDER document in Figure 2, for example, is equivalent to:

```
[] rdf:type owl:Ontology ;
   wdrs:issuedby
     <http://philarcher.org/foaf.rdf#me> ;
   wdrs:validuntil
     "2012-12-31T23:59:59"^^xsd:datetime .
```

This has the repercussion that in order to combine two or more POWDER documents, all trust and validity checks must be performed *before* mixing the statements from multiple documents into a single knowledge base, as in the combined base it is impossible to have different ontology headers for each sub-ontology that was added.

Finally, POWDER also allows ordered lists of increasingly generic DRs where the descriptor set of the first DR that matches the resources is applied and all subsequent DRs are ignored. We shall discuss how this is reconciled with OWL/RDF's open-world semantics in Section 4.3 below.

4.2 The POWDER-S Extension

While POWDER-S uses OWL classes to group resources, determining if a given resource belongs in one of these OWL classes involves testing the resource's URI against a regular expression.

The logical foundations of POWDER are based on an extension of RDF semantics that defines two datatype properties, `wdrs:matchesregex` and `wdrs:notmatchesregex`, such that their interpretation includes all pairs of URI-denoted resources and regular expression literals such that the resource's normalized string encoding matches (does not match) the regular expression.

These properties relate resources to regular expressions which that resource matches. Formally, they are defined as follows (Konstantopoulos and Archer, 2009, Section 4.3):

Definition 1: We define `wdrs:matchesregex` as:

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

with the further stipulation that $< x, reg >$ is in its extension 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 references and their string representation, as specified in Section 6.4 of the RDF Concepts document (Klyne and Carroll, 2004).
  - *sss* matches the regular expression *reg*, AND
  - $\mathrm{I}(uuu) = x$

where $\mathrm{I}(\cdot)$ is the interpretation function that maps URI references to resources.

Definition 2: We define `wdrs:notmatchesregex` as:

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

with the further stipulation that $< x, reg >$ is in its extension 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 references and their string representation, as specified in Section 6.4 of the RDF Concepts document (Klyne and Carroll, 2004).
  - *sss* does not match the regular expression *reg*, AND
  - $\mathrm{I}(uuu) = x$

where I(·) is the interpretation function that maps URI references to resources.

It is now possible to express <includeregex/> and <excluderegex/> as `owl:hasValue` restrictions on `wdrs:matchesregex` and `wdrs:notmatchesregex` respectively and build up an OWL Class to represent the set of resources that match (do not match) the expressions.

Under this extension, the semantics of POWDER documents can be formulated in OWL/RDF; Figure 6, for example, demonstrates the results of the POWDER to POWDER-S transform for the example in Figure 2. Formulating in OWL/DRF allows POWDER statements to be fully understood by and interoperable with other Semantic Web technologies and tools.

4.3 Discussion and Rationale

One key design choice in POWDER, and maybe the hardest to make, was the definition of the `wdrs:notmatchesregex` extension. Using Lemma 1 as a starting point, one can see that the main problem is defining the class of resources that have a URI that matches an IRI sets' criteria; annotating the members of this class with the properties in the descriptor set is then straight-forward using class subsumption and `owl:hasValue` restrictions.

Ideally, one would prefer to define POWDER in terms of OWL/RDF without any extension, but that was clearly not possible. Naturally one can always record data in RDF, but that does not automatically imply capturing the intention behind them. That is to say, simply using a datatype property to hold the regular-expression definition of an IRI set does not mean that the indention of the IRI set has been captured and that Semantic Web tools can decide whether any given resource is a member of the IRI set or not.

Consequently, POWDER had to introduce a minimal amount of new vocabulary; hopefully, of vocabulary generic enough to be usable in future use cases beyond POWDER. Achieving this makes POWDER fit easily in the Semantic Web stack, by lowering the entry bar for existing Semantic Web tools and by independently motivating the implementation of the POWDER extension. The `wdrs:matchesregex` extension to a large extend achieves this goal:

- it encodes the relationship between URI resources and their URIs absent from OWL/RDF but needed in order to define IRI sets;
- it encodes *only* the IRI set semantics that could not be stated in OWL/RDF, leaving DR semantics to be defined through the `rdfs:subClassOf` relation between IRI sets and OWL classes or `owl:hasValue` restrictions; and, finally,
- it is a vocabulary item that can prove useful in other applications, such as as formally describing HTTP interactions.[8]

Regarding the duality of Definitions 1 and 2, it should be noted that `wdrs:matchesregex` and `wdrs:notmatchesregex` are *not* complementary in the strict, open-world sense of OWL/RDF model-theoretic semantics and *could not* have been defined in terms of each other and OWL complementization. More specifically, for any regular expression *re* there are (unnamed) resources that belong to neither the `wdrs:matchesregex` *re* nor the `wdrs:notmatchesregex` *re* IRI set, hence these two IRI sets are *not* the complement or each other.

Besides allowing us to define the semantics of <excluderegex/> elements in IRI sets, Definition 2 also achieves another purpose. As POWDER foresees *DR lists* with *first-match-wins* semantics (cf. Section 4.1 above and Section 2.8.1, Archer et al. 2009), `wdrs:matchesregex` and `wdrs:notmatchesregex` allow us to create a *locally closed-world* semantics that captures exceptions. More specifically, exceptions to more general DRs can be encoded by conjoining the IRI set of the more general DR with the IRI set that does *not match* the regular expression in the exception (*matches* if the exception's IRI set is an <excluderegex/> element, cf. Section 3.1, Konstantopoulos and Archer 2009 for a detailed description). In retrospect, the explicit stipulation that DR lists are asserted atomically and no new exceptions can be asserted in an existing DR list should have been added to guarantee monotonicity.

4.4 Alternative Approaches

In the original OWL specification there was very little provision for manipulating the *concrete domain* of strings, numbers, and other literal values. Besides the pre-defined base types, user-defined datatypes could only be specified by explicit enumeration and there was no mechanism for concrete datatype descriptions analogous to abstract resource class descriptions.

The, recently finalized, OWL 2 recommendation would have been a better basis for establishing POWDER semantics, but was not available at the time. OWL 2 allows user-defined *literal types* to be constructed by using any of the *facets* defined by the *XML Schema Datatypes* (XSD) recommendation (W3C OWL Working Group, 2009, Section 7). XSD facets allow descriptions that specify a range within the base type the literal type is derived from. Regular expressions, called *patterns* in the XSD recommendation (Biron and Malhotra, 2004, Section 4.3.4), are allowed as facets over the string base type.

Using this, POWDER could have defined an IRI set by:

- defining the `wdrs:hasURI` datatype property that relates URI resources with the `xsd:string` representation of their URI;
- using existing XSD vocabulary to derive from `xsd:string` the literal type that includes all and

```
@prefix wdrs: <http://www.w3.org/2007/05/powder-s#> .
@prefix owl:  <http://www.w3.org/2002/07/owl#> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

[] rdf:type owl:Class ;
   rdfs:subClassOf <http://data.ordnancesurvey.co.uk/ontology/postcode/PostcodeUnit> ;
   owl:equivalentClass
     [ rdf:type owl:Class ;
       owl:intersectionOf (
         [ rdf:type owl:Restriction ;
           owl:hasValue "\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?
(data\.ordnancesurvey\.co\.uk)(\:([0-9]+))?\/"^^xsd:string ;
           owl:onProperty wdrs:matchesregex ]
         [ rdf:type owl:Restriction ;
           owl:hasValue "\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]*)(\:([0-9]+))?
(\/id\/postcodeunit)"^^xsd:string ;
           owl:onProperty wdrs:matchesregex ] .
```

Figure 6 POWDER-S formulation of the POWDER document in Figure 2.

only the strings that match the regular expression in the <iriset/> element;

- using existing OWL 2 vocabulary to define the set of resources that have the `wdrs:hasURI` property and that have a value for that property within the literal type above.

The advantage is that the regular expression matching is specified using existing vocabulary and the POWDER extension is restricted to realizing the straight-forward 1:1 relation between URI resources and their URIs.

As OWL 2 was, at the time, far from being finalized, let alone implemented in inference tools, it was decided to only include this alternative as a non-normative addendum in anticipation of the OWL 2 (Konstantopoulos and Archer, 2009, Section 4.6).

5 Implementations

A specification without implementations is fiction so, as with any standard worthy of serious attention, it has to be proved with running code.

If the context makes it appropriate, then POWDER can be processed entirely as XML. Two such proof-of-concept POWDER/XML processors were implemented during the POWDER Working Group's charter, the i-sieve POWDER Processor[9] and the 3P POWDER Processor.[10] Both process POWDER documents to provide a Web form interface for querying about resources to get RDF annotations. A further XML implementation is integrated in the AQUA system, described in Section 6.

In the remainder of this section, we will describe three alternative approaches to implementing an OWL/RDF processor that can be seamlessly integrated with other Semantic Web tools, such as triples stores and reasoners.[11]

All take a *layered inference* approach where semantic (or any other type of) inference is applied on the results of the POWDER layer, without needing to push later results back into the POWDER layer. This is in compliance with POWDER semantics: note the *if and only if* in Definitions 1 and 2, making the provision that neither explicit assertion nor semantic inference results can affect the interpretation of `wdrs:matchesregex`.[12] That is to say, a resource can only be the subject of a `wdrs:matchesregex` statement if it is a URI resource and its URI matches the regular expression in the object.

5.1 Model-theoretic semantics

The most straightforward implementation of a POWDER processor is to transform POWDER documents into their POWDER-S equivalent, make all applicable `wdrs:matchesregex` assertions for all URI resource × regular expression pairs[13] in the repository, and then use OWL inference.

This is the approach taken by SemPP, one of the reference implementations produced by the POWDER Working Group.[14] More specifically, SemPP is implemented within the Jena framework[15] by extending the standard Jena `OntModel` with one that intercepts the assertion of new URI resource properties or new POWDER-S restrictions to ensure that all the appropriate `wdrs:matchesregex` assertions are also made. This extended `OntModel` is used as the explicit triple back-end for the `InfModel` implementation provided by the Pellet[16] OWL reasoner.

Although a valid and successful proof of concept, there are two reasons why this approach is inefficient to the point of being unusable in real-world situations: although some optimization is achieved in cases where multiple properties are implied for each `wdrs:matchesregex`, in most cases asserting one statement (either `wdrs:matchesregex` or `wdrs:notmatchesregex`) for each URI × regular expression will make the repository *bigger* than if POWDER were not used.

The second reason is that the requirement to use OWL inference might be an unreasonable price for applications where only RDFS or, even more more so, only retrieval of explicit triples, is sufficient.

5.2 *Proof-theoretic semantics*

Looking at POWDER from another perspective, it is, effectively, a form of inference, not radically unlike semantic inference, except that the premises are the URI restrictions expressed in <`iriset`>s and the consequents are the properties expressed in <`descriptorset`>s.

Pursuing this angle, the extension in RDF semantics that POWDER defines can be applied to the proof-theoretic RDF and RDFS semantics (Hayes, 2004, Section 7) rather than the model-theoretic semantics used in the POWDER Recommendation and in Section 5.1 above.

The extension amounts, then, to augmenting the rules of RDFS inference as follows:

$$\frac{\text{covers}(dr, \texttt{xxx}) \quad \texttt{ppp} = \text{prop}(dr) \quad \texttt{yyy} = \text{value}(dr)}{\texttt{xxx ppp yyy}}$$

This rule infers triples from no premises other than the information in the DR, using a function that matches a resource's URI against the regular expressions in the DR's <`iriset/`> to decide if the resource falls within the scope of the DR. No other rule activates this rule (since no semantic inference can ever affect POWDER inference) and this rule activates the following RDFS rules:

$$\frac{\texttt{qqq rdfs:subPropertyOf ppp} \quad \texttt{xxx qqq yyy}}{\texttt{xxx ppp yyy}}$$

$$\frac{\texttt{zzz rdfs:subClassOf yyy} \quad \texttt{xxx rdf:type zzz}}{\texttt{xxx rdf:type yyy}}$$

This activation pattern does not, obviously, influence reasoning termination since there is no push-back of results from the other rules to the POWDER rule. Furthermore, time complexity for each step depends linearly on the number of DRs in the knowledge base and the number of URI resources, so is, in the worst case, quadratic to the size of the repository; since RDFS deduction is NP-complete in the first place (Baget, 2005), the extension does not lead to an increase in the complexity class. In practise, the increase in deduction time will be linear rather than quadratic since the number of DRs will typically be a very small fraction of the repository size (effectively, a constant).

This proof-theoretic extension was implemented by modifying the forward-chaining RDFS inference engine bundled with the Sesame 2 framework.[17] Although efficient, this approach has the main disadvantage that it is tied to a particular inference engine and that query-time performance is dependent on looking up indexed triples (although inferred ones rather than explicit data).

5.3 *Query Rewriting*

In the previous approaches POWDER either introduced a dependency on OWL inference (Section 5.1) or was implemented as an extension of a particular inference tool (Section 5.2). We have explored the possibility of implementing POWDER in a way that does not require any other form of inference but can be integrated into a stack that performs any (or none) semantic inference.

In order to achieve this, we have implemented a POWDER layer over a standard triple store. This layer is a full POWDER processor without any dependencies on OWL or RDFS reasoning; that is to say, the POWDER layer does not implement the `wdrs:matchesregex` extension, but instead it directly annotated resources in the triple store based on the DRs it operates upon.

Furthermore, this has been approached as *layered inference*, taking advantage of the *Storage And Inference Layers* (SAILs) architecture of the Sesame 2 framework. Sesame SAILs are 'stackable' components that infer implicit RDF triples from the (explicit or also implicit) data they receive from the SAIL immediately below. The Sesame distribution bundles a number of components for this framework, including a memory-based and a disk-based triple store and RDFS inference.

In this framework, a POWDER processor has been implemented as a SAIL that is stacked between the RDF store and the semantic inference SAIL. In this manner, from the perspective of the SAIL stacked over it, POWDER appears as if it were implemented inside the store, allowing for any inference engine implemented as a SAIL to be deployed without modifications (Konstantopoulos and Archer, 2011).

Triple retrieval is handled by wrapping the standard Sesame repository results implementation with one that augments the results returned by the bottom SAIL (RDF store) as follows:

- If the subject of the triple pattern is blank, then no triples are added.
- Otherwise:
  - If the predicate is blank, the subject's URI is tested against the regular expressions of all DRs.
  - Otherwise, the subject's URI is tested against the regular expressions of all DRs that assign the predicate specified in the pattern.

  Matching DRs provide potential triples, stored in the in-memory data structure in the repository results object.

  - If the object is blank, all potential triples are kept.
  - Otherwise, only potential triples with the same object as the one in the pattern are kept.

A point that should be made is that these assertions are never materialized (not even as inferred triples) but are rather 'simulated' by an approach that first extends and then filters query and triple retrieval results. In short, triple patterns involving POWDER-endowed properties are first ignored (to accept *any* resource as having the POWDER-assigned properties) and then the results are filtered so that only those bindings that satisfy the URI restrictions specified in the POWDER document are retained.

This approach is very efficient for large data stores, as it avoids asserting (either on the persistent store or in any in-memory structures used at query time) the POWDER-inferred triples. It has, however, the drawback that it is only complete for the instance-based inference needed to support querying and cannot yield any results requiring TBox reasoning. In becomes, furthermore, very inefficient for queries that have no triple patterns other than those with POWDER-assigned properties: in the absence of any indexed (non-POWDER) property to *guard* the query results, the query engine will have to check the regular expressions against each and every URI resource in the repository.

6 Web Content Accreditation

Different organizations around the world are working on establishing standards of quality in the accreditation of health-related web content. The European Council supported an initiative within eEurope 2002 to develop a core set of Quality Criteria for Health Related Websites (of the European Communities, 2002). These criteria may be used as a basis in the development of user guides, voluntary codes of conduct, trust marks, accreditation systems, adopted by relevant parties, at European, national, regional or organizational level.

There are two major mechanisms in medical quality labeling: filtering portals and third party accreditation. In *filtering portals* the web resources are classified according to predetermined criteria in order to facilitate a quick access to quality-reviewed information. Examples of this mechanism are the *Catalog and Index of French-speaking Medical Sites*,[18] the health and life sciences branch of the *Intute* service[19] in the U.K., and the *Agency for Quality in Medicine*[20] from Germany.

In *third-party accreditation*, on the other hand, an organization evaluates the quality of the web site according to a set of criteria and compliance is demonstrated to site visitors with a logo or trustmark on the homepage. The *HONCode* of the *Health on the Net* foundation,[21] the *URAC Accreditation Program*,[22] and the *Web Médica Acreditada*[23] trustmarks are the most well known quality seals of this type.

The main problem that these mechanisms face is the need for a continuous review and control of the accredited or classified web sites that requires a huge amount of human effort. This stress on content quality evaluation contrasts with the fact that most Web content annotation specifies how to layout content for the benefit of human readers.

Recent developments, such as the Semantic Web and Linked Open Data, have pushed forward the evolution of the Web from a repository of human-understandable information to a global knowledge repository or machine-readable and machine-processable information, enabling the use of advanced knowledge management technologies (Eysenbach, 2003).

Such metadata can be expressed in different ways using RDF. In order for the medical quality labeling mechanisms to be successful, they must be equipped with Semantic Web technologies that enable the creation of machine-processable labels as well as the automation of the labeling process.

6.1 AQUA System Overview

By analyzing the two main approaches of medical quality labeling (filtering portals and third party accreditation), we have identified the following key tasks, followed entirely or partially by most labeling agencies:

- Identification of new web resources: this could happen either by active web searching or by voluntary application from the information provider.
- Labeling of the web resources: this could be done with the purpose of awarding an accreditation seal or in order to classify and index the web resources in a filtering portal.
- Reviewing or monitoring the labeled web resources: this step is necessary to identify changes or updates in the resources as well as broken links and to verify if a resource still deserves to be awarded an accreditation seal.

As a result, the AQUA system (Mayer et al., 2011) was designed to support the main tasks of the web content accreditation process, that is: identification of unlabeled resources having health-related content; visit and review of the identified resources; generation of content labels for the reviewed resources, and monitoring the labeled resources.

Compared to other approaches that partially address the assessment process (Griffiths et al., 2005; Wang and Liu, 2006), the AQUA system is an integrated solution. AQUA aims to provide the infrastructure and the means to organize and support various aspects of the daily work of labeling experts by making them computer-assisted. More specifically, AQUA supports labeling experts in:

- Creating machine readable labels, by adopting the use of the POWDER model for producing machine-readable content labels.
- Automating the accreditation process by helping in the identification of unlabeled resources, extracting from these resources information relative to specific accreditation criteria, generating content labels from the extracted information and facilitating the monitoring of already labeled resources.

AQUA supports the labeling expert by integrating a series of content collection, analysis, and labelling components: the *Web Content Collection* (WCC) component that identifies, classifies and collects on-line content relative to the labeling criteria. The *Information Extraction Toolkit* (IET) analyses the web content collected by WCC and extracts attributes for the content labels. The *Label Management* (LAM) component generates, validates, modifies, and compares the content labels. All three components are supported by the *Multilingual Resources Management* (MRM) subsystem with access to health-related multilingual resources. Finally, the *Monitor-Update-Alert* (MUA) tool handles the configuration of monitoring tasks, database updates, and alerts to labeling experts.

6.2 The AQUA LAM Component: Creating Machine-Readable Labels

The label management interface and associated tools, together called LAM, allows experts to generate, update and compare content labels. From within the LAM user interface the user is able to

- generate new RDF labels from information automatically extracted by other AQUA tools;
- manually fill the relevant fields and generate new RDF labels;
- edit and update existing RDF labels; and
- compare RDF labels among themselves.

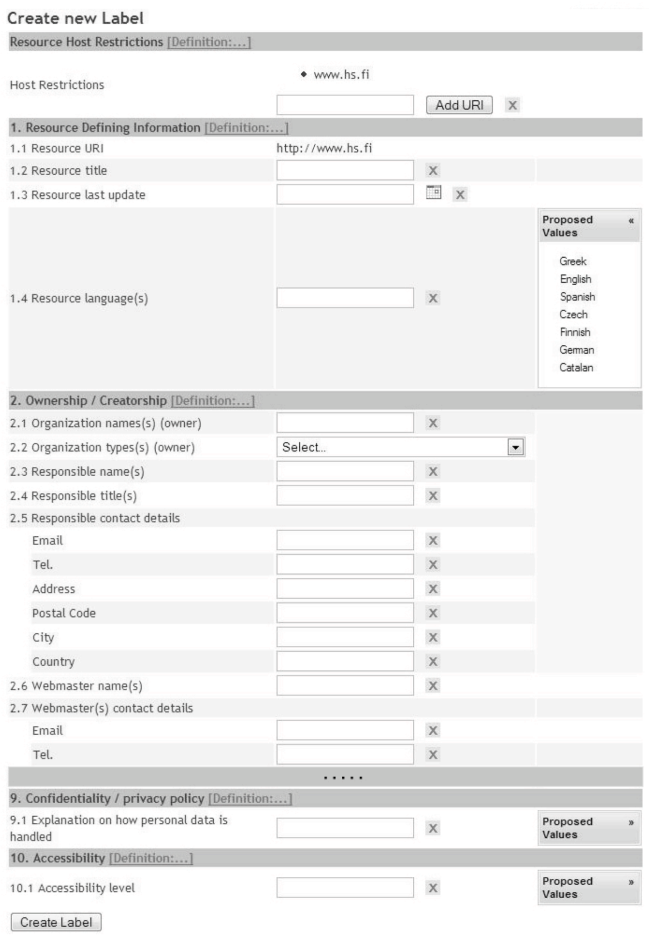
The user interface to generate/edit a label is a web form (see Figure 7) with input boxes, single and multiple select boxes, links and buttons. It is split into two distinct areas. The first part lets the user constrain the application of the label to certain hosts by explicitly declaring the host URIs or by adding regular expressions that properly identify them. Multiple hosts can be defined. Regular expressions for more fine-grained addressing can be defined as well. These definitions can be combined via the union and intersection operators and thus create rules that link different parts of a web resource with different labels.

The second part is where the label properties are assigned values. The label properties are the actual descriptors of a web resource, mapping the labeling criteria. A set of label descriptors can be linked with a set of host restrictions defined in the first part. Related properties are grouped to make the user filling them easier.

Once annotators have filled the label metadata, restrictions and properties, they can save the label in the internal AQUA database. This database supports version control and exporting RDF/XML serializations.

6.3 Benefits from POWDER adoption

The Protocol for Web Description Resources (POWDER) was selected as the driving technology for


Create new Label

Resource Host Restrictions [Definition:...]

Host Restrictions	• www.hs.fi

1. Resource Defining Information [Definition:...]

1.1 Resource URI	http://www.hs.fi
1.2 Resource title	
1.3 Resource last update	
1.4 Resource language(s)	

Proposed Values: Greek, English, Spanish, Czech, Finnish, German, Catalan

2. Ownership / Creatorship [Definition:...]

2.1 Organization names(s) (owner)	
2.2 Organization types(s) (owner)	Select..
2.3 Responsible name(s)	
2.4 Responsible title(s)	
2.5 Responsible contact details	
Email	
Tel.	
Address	
Postal Code	
City	
Country	
2.6 Webmaster name(s)	
2.7 Webmaster(s) contact details	
Email	
Tel.	

.....

9. Confidentiality / privacy policy [Definition:...]

9.1 Explanation on how personal data is handled	

10. Accessibility [Definition:...]

10.1 Accessibility level	

Figure 7 The AQUA LAM interface.

MedIEQ metadata description because it provides a consistent mechanism for the provision of descriptions and the means to authenticate and apply them to a group of on-line resources.

The AQUA prototype was evaluated by the labeling organizations participating in the MedIEQ project (namely, WMA and AQUMED). The primary goal of this evaluation was to conclude with a functional prototype that has the potential to be fully integrated within the day-to-day activities of a labeling organization. To this end, a parallel technical improvement action took place, refining given functionalities. The main objective of the extra technical improvement action was to enhance the overall system work-flow, so as to better match the day-to-day practice. The specifications for these technical refinements were given by an iterative feedback process with the MedIEQ labeling organizations, during the evaluation. It must be noted that the final version of AQUA was well received by both labeling organizations.

Below, we summarize some conclusions about the application of POWDER that came out from this evaluation:

- POWDER does not enforce any limitations on the descriptive vocabularies that can be used for the creation of a description, as long as the vocabulary is associated with a namespace. Thus, it provides maximal flexibility to ensure the accuracy and cohesiveness of the description.

- The mechanism for defining the scope of a description is one of the most powerful features of the protocol. POWDER allows the association of a description with a group of on-line resources instead of a single resource. The grouping can be achieved with various methods, for extremely simple such as a list of IRIs, to extremely powerful such as matching the IRI of a resource to a regular expression. Evidently, this method offers the capability to create a single description and associate it with a potentially huge amount of resources, thus reducing the complexity for the issuer of the POWDER document.
- Resources and the DRs that describe them may be connected to each other in a web of trust. Partners in this web may be certification authorities that verify the truthfulness of claims made in Description Resources and the reputation of their issuer. Therefore, POWDER facilitates the examination of the validity and veracity of the metadata associated with a resource.

6.4 Other POWDER Deployments

The *Wholesale Application Community* (WAC) supports the inclusion of links to POWDER instances to describe widgets. The use case provided in the (still evolving) standard is centred on child protection but could readily be extended into other areas. Unfortunately there are some inconsistencies in the way in which WAC suggests POWDER is used and authors are working to try and correct these.

MetaCert.com actively uses POWDER to provide information about sites on the `.xxx` top level domain and those that link to it. Although currently focused on child protection, the company will be rolling out additional products and services around accessibility, trustmarks, and mobile-friendliness, in essence building a suite of commercial services that reflect many of POWDER's original use cases. They are spearheading efforts to see POWDER supported in the major browsers, with a Firefox add-on as the exemplar.

7 Repository Compression

Although POWDER was originally envisioned as a protocol for publishing metadata, its versatility at mass-annotating resources has given rise to an unexpected, at the time of developing the protocol, application as a technology for *repository compression.*

7.1 Domain and Ontology

The SYNC3 domain is that of news and events described in news articles and blog posts, so that the concepts of a text *document* and of a news-worthy *event* reported in it are prominently situated in the SYNC3 model.

We shall not delve into the details of the linguistic processing pipeline of SYNC3 (Konstantopoulos and Archer, 2011; Sarris et al., 2011), but it suffices to say that at the end of this processing, the following information about documents and events has been extracted:

- Document metadata, including title, date of publication, and source.
- A breakdown of documents into *segments*, each being a list of consecutive syntactic elements of the document which document the same event.
- The resolution of the *abstract domain entity* that each concrete term, pronoun or other anaphora in the document refers to.
- The *semantics* of each segment, which comprises the event that is being documented as well as the subset of all terms appearing in the segment that refer to entities participating in the event.
- The geographical and temporal grounding of an event, as well as a numerical valuation of the participation of the various domain entities in an event.

The SYNC3 ontology[24] extends the DOLCE+DnS UltraLite top-level ontology[25] to capture and index this information in a semantic repository supporting multi-faceted and semantic querying. Documents, segments, named entities, and events in SYNC3 receive a unique numerical ID as soon as they are crawled or extracted by the linguistic processing tools. This ID can be used to retrieve from an XML database information about them via Web services located at URLs constructed from their IDs as shown in the following examples:

- News article with ID 10123:
  `http://sync3.atc.gr/Sync3MultiMediaInfo/resources/newsArticles/10123`
- The second segment of the document above:
  `http://sync3.atc.gr/Sync3MultiMediaInfo/resources/newsArticles/10123/segments/10123_2`
- Named entity of type *person* with ID 42:
  `http://sync3.atc.gr/Sync3MultiMediaInfo/resources/persons/42`
- Event with ID 1990:
  `http://sync3.atc.gr/Sync3MultiMediaInfo/resources/events/1990`

Although membership in some of these concepts (for example, `dul:Event`) can be directly inferred from property domain and range restrictions, some distinctions cannot be made by ontological axioms, but are useful information for answering queries. So, for example, both news items and blog posts have the same structure and are, as a consequence, identically modelled as instances of the `sync3:Content` concept. The distinction, however, between news items and blog posts must be retrievable by repository clients, and so `sync3:Content` instances are `dul:describedBy` either the resource `sync3:newsItem` or the resource `sync3:blogPost`, both instances of `dul:Description`.

Similarly for the sub-types of named entities (person, location, organization, etc.) which also need to be available to queries, membership in particular DUL types cannot be inferred but needs to be explicitly asserted.

Such resource groupings can be very naturally expressed in POWDER, as shown in Figure 8. The full POWDER document for the XML database is available at `http://sync3.atc.gr/powder.xml`

7.2 Results

The amount of triples saved by using POWDER to mass-annotate resources depends on the application and the density of the URI-predictable properties. In our application, the savings on content type assertions (news or blog) is linearly related to the amount of content stored in the repository, but the savings on domain entity type assertions (as extracted by the Named Entity Recognizer, cf. Section 7.1) depends heavily on the density of newly encountered domain entities, as these require a new domain entity instance to be typed.

In order to estimate the amount of triples saved, we used SYNC3 data to instantiate two Sesame2 Native Store instances, one full and one where POWDER-inferable triples were excluded. This data represents a week's worth of extracted and analysed news articles. As can be seen in Table 1, the compression is substantial but also shows a diminishing gains effect; the rate, however, at which the compression rate drops is also falling, converging to a steady compression rate of around 13%.

The initially higher compression is explained by the re-use of domain entities as they repeatedly appear in articles, so that fewer domain entity-type assertions are required. The exact compression rate is, naturally, specific to each particular ontological schema, URI design, and application, and not broad generalization can be made.

Table 1 Compression, in terms of number of triples and as percentage of the original size, achieved by applying POWDER to the SYNC3 repository.

Size in Mtriples		Difference	
Raw	POWDER	Absolute	Percent
4.21	3.23	1,031,501	-23.4%
63.29	55.34	8,327,897	-12.6%
92.58	80.42	12,744,179	-13.1%
155.23	134.90	21,311,993	-13.1%
208.40	181.22	28,496,983	-13.0%

What should also be noted is that querying time is practically unaffected as shown on Table 2 (reported by Konstantopoulos 2011). This demonstrates how having to search through smaller indexes compensates for the overhead of the POWDER preocessing, especially in the case of the query re-writing implementation that binds some of the patterns in the query without any access to the store.

8 Conclusions

The POWDER protocol was originally conceived as a simple and intuitive, human-editable XML schema for annotating web content for the purposes of content accreditation and annotation (e.g., is this medical site reliable?) and resource discovery (e.g., where is the mobile-friendly part of this site?).

Near the end of the POWDER working group's original charter, the idea was conceived that in order to fit nicely into the Semantic Web architecture POWDER should have a formal semantics founded on OWL/RDF and Lemma 1 was formulated.[26]

This has necessitated the definition of the `wdrs:matchesregex` extension (Section 4.2), which has successfully minimized the amount of new vocabulary introduced in order to express POWDER, relying mostly on existing OWL/RDF terms to define POWDER terms; while at the same time the introduced vocabulary can be motivated independently of POWDER, as demonstrated by recent discussion on formalizing HTTP interactions (cf. Section 4.3).

Working out an OWL/RDF formal semantics for POWDER has given it the versatility and interoperability that has allowed it to be deployed in use cases that have not been foreseen during the protocol's development, like the repository compression method presented in Section 7.

Further promising use cases that have emerged include annotating resources with references to related data, pushing forward the linked data idea. More specifically, and returning to our earlier Ordnance Survey example, given a URI such as `http://data.`

Table 2 Querying time for three repositories holding one month of SYNC3 data (38Mtriples). The 'vanilla' repository stacks the Sesame RDFS reasoner SAIL over the Native Store SAIL.

Tuples	Time (in seconds)		
retrieved	Vanilla	POWDER	RDFS/P
62	0.9	0.57	0.75
538	32	32	13
706	22	19	14
2176	123	100	117
2233	109	86	106
3418	117	96	114

```
<dr>
  <iriset>
    <includehosts>sync3.atc.gr</includehosts>
    <includepathstartswith>Sync3MultiMediaInfo/resources/news</includepathstartswith>
  </iriset>
  <descriptorset>
    <typeof src="http://sync3.eu/rdf/sync3#Content" />
    <dul:isDescribedBy rdf:resource="http://sync3.eu/rdf/sync3#NewsItem"/>
  </descriptorset>
</dr>

<dr>
  <iriset>
    <includehosts>sync3.atc.gr</includehosts>
    <includepathstartswith>Sync3MultiMediaInfo/resources/persons</includepathstartswith>
  </iriset>
  <descriptorset>
    <typeof src="http://www.loa-cnr.it/ontologies/DUL.owl#Person"/>
  </descriptorset>
</dr>
```

Figure 8 Sample Description Resources from the POWDER document used to describe content and domain entity instances in SYNC3. The full POWDER document for the XML database is available at `http://sync3.atc.gr/powder.xml`

`ordnancesurvey.co.uk/id/postcodeunit/AL11AE` where does one go to run SPARQL queries to find out more? One way to find out is to look at the VoID (Alexander et al., 2011) file at the root of `http://data.ordnancesurvey.co.uk` which gives the address to SPARQL endpoint for querying about resources within the domain. This, however, assumes that all data publishers provide a VoID description of their data and that they link to relevant data by other publishers.

By contrast, publishing a POWDER document such as the one in Figure 9 offers a more flexible way to publishing VoID descriptions than looking at well-known locations. Among other advantages, POWDER would allow not only the Ordnance Survey but also other entities to annotate the Ordnance Survey site with VoID metadata, linking, for example, to third-party endpoints with relevant data. POWDER not only provides the vocabulary to annotate the complete Ordnance Survey site, but also provides the provenance information that each data consumer needs in order to decide if such third-party metadata is to be trusted,

Further use cases can also be investigated, such as publishing regular expressions that map URIs to other URIs, allowing a publisher to *formally* and *concisely describe a URI restructuring* and alleviating the pressure for URI persistency; or using POWDER to describe the kinds of resources and properties that an endpoint holds data about, enabling novel *resource discovery* methods to be developed for large-scale distributed repositories.

Acknowledgements

S. Konstantopoulos wishes to acknowledge the partial support of the European FP7-ICT project SYNC3. Please see `http://www.sync3.eu` for more information.

References

Alexander, K., R. Cyganiak, M. Hausenblas, and J. Zhao (2011, March). Describing linked datasets with the VoID vocabulary. W3C Interest Group Note, 3 March 2011. URL `http://www.w3.org/TR/void`

Archer, P., K. Smith, and A. Perego (2009, September). Protocol for web description resources (POWDER): Description resources. W3C Recommendation, 1 September 2009. URL `http://www.w3.org/TR/powder-dr`

Baget, J.-F. (2005). RDF entailment as a graph homomorphism. In *Proceedings of the 4th International Semantic Web Conference (ISWC 2005)*, Number 3729 in Lecture Notes in Computer Science, Berlin/Heidelberg, pp. 82–96. Springer Verlag.

Bechhofer, S., F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, and L. A. Stein (2004, February). OWL Web Ontology Language: Reference. W3C Recommendation, 10 February 2004. URL `http://www.w3.org/TR/owl-ref`

Biron, P. V. and A. Malhotra (2004, October). XML Schema part 2: Datatypes second edition. W3C Recommendation, 28 October 2004. URL `http://www.w3.org/TR/xmlschema-2`

Connoly, D. (2007, September). Gleaning resource descriptions from dialects of languages (GRDDL). W3C Recommendation, 11 September 2007. URL `http://www.w3.org/TR/grddl`

```
<dr>
  <iriset>
    <includehosts>data.ordnancesurvey.co.uk</includehosts>
  </iriset>
  <descriptorset>
    <void:sparqlEndpoint
rdf:resource="http://api.talis.com/stores/ordnance-survey/services/sparql"/>
  </descriptorset>
</dr>
```

Figure 9 An example DR (minus namespace declarations) giving the SPARQL endpoint for any URI on the data.ordnancesurvey.co.uk domain.

Eysenbach, G. (2003). The Semantic Web and healthcare consumers: A new challenge and ppportunity on the horizon? *J Healthc Techn Manag 5*, 194–212.

Gibbins, N. and N. Shadbolt (2010). Resource Description Framework (RDF). In *Encyclopedia of Library and Information Sciences* (Third Edition ed.). CRC Press.

Griffiths, K. M., T. T. Tang, D. Hawking, and H. Christensen (2005, December). Automated assessment of the quality of depression websites. *J Med Internet Res 7*(5).

Hayes, P. (2004, February). RDF semantics. W3C Recommendation, 10 February 2004. URL `http://www.w3.org/TR/rdf-mt`

Klyne, G. and J. J. Carroll (2004, February). RDF concepts and abstract syntax. W3C Recommendation, 10 February 2004. URL `http://www.w3.org/TR/rdf-concepts`

Konstantopoulos, S. (2011). Three ways to sprinkle powder. In *Proceedings of the Posters Session of the 10th International Semantic Web Conference (ISWC 2011), Bonn, 2011.*

Konstantopoulos, S. and P. Archer (2009, September). Protocol for web description resources (POWDER): Formal semantics. W3C Recommendation, 1 September 2009. URL `http://www.w3.org/TR/powder-formal`

Konstantopoulos, S. and P. Archer (2011). POWDER and the multi million-triple store. In *Proceedings of the International Workshop on Semantic Web Information Management (SWIM 2011), SIGMOD/PODS 2011, Athens, 12 June 2011*, New York, USA. ACM.

Lassila, O. (1997, May). PICS-NG metadata model and label syntax. W3C Note, 14 May 1997. URL `http://www.w3.org/TR/NOTE-pics-ng-metadata`

Lassila, O. and R. R. Swick (1999, February). Resource Description Framework (RDF): Model and syntax specification. W3C Recommendation, 22 February 1999. URL `http://www.w3.org/TR/1999/REC-rdf-syntax-19990222`

Manola, F. and E. Miller (2004, February). RDF primer. W3C Recommendation, 10 February 2004. URL `http://www.w3.org/TR/rdf-primer`

Mayer, M., P. Karampiperis, A. Kukurikos, V. Karkaletsis, K. Stamatakis, D. Villarroel, and A. Leis (2011). Applying semantic web technologies to improve the retrieval, credibility and use of health-related web resources. *Health Informatics Journal 17*(2), 95–115. Special Issue on Semantic Descriptions of Medical Web Resources: Technologies to support their Creation, Maintenance and Access.

Nottingham, M. and E. Hammer-Lahav (2010, April). Defining well-known Uniform Resource Identifiers. IETF RFC 5785, April 2010. URL `http://tools.ietf.org/html/rfc5785`

of the European Communities, C. (2002). eEurope 2002: Quality criteria for health related websites. *Journal of Medical Internet Research (JMIR) 4*(3).

Perego, A. and P. Archer (2009, September). Protocol for web description resources (POWDER): Grouping of resources. W3C Recommendation, 1 September 2009. URL `http://www.w3.org/TR/powder-grouping`

Sarris, N., G. Potamianos, J.-M. Renders, C. Grover, E. Karstens, L. Kallipolitis, V. Tountopoulos, G. Petasis, A. Krithara, M. Gallé, G. Jacquet, B. Alex, R. Tobin, and L. Bounegru (2011). A system for synergistically structuring news content from traditional media and the blogosphere. In *Proceedings of the eChallenges e-2011 Conference, Florence, 26–28 October 2011.*

Sauermann, L. and R. Cyganiak (2008, December). Cool URIs for the Semantic Web. W3C Interest Group Note, 3 December 2008. URL `http://www.w3.org/TR/cooluris`

W3C OWL Working Group (2009, October). OWL 2 web ontology language: Primer. W3C Recommendation, 27 October 2009. URL `http://www.w3.org/TR/owl2-primer`

Wang, Y. and Z. Liu (2006, May). Automatic detecting indicators for quality of health information on the web. *Int J Med Inform.*

Notes

[1]Issue raised by D. Connoly, available at `http://www.w3.org/2000/03/rdf-tracking/#rdfms-abouteach`

[2]Item 16 of the minutes of the working group meeting on 7th December 2001. URL: `http://lists.w3.org/Archives/Public/w3c-rdfcore-wg/2001Dec/0089.html`

[3]Jeni Tennison, `http://www.jenitennison.com/blog/node/135`

[4]John Goodwin, `http://johngoodwin225.wordpress.com/2009/10/25/location-location-location-exploring-ordnance-survey-linked-data`

[5]Available at `http://philarcher.org/powder/postcodeunit.xml`

[6]POWDER refers to Internationalised URIs (IRIs) and the specifications include relevant information about canonicalisation. In this article, we generally use the term

URI, but retain references to IRIs when directly quoting the POWDER protocol or using POWDER terminology such as 'IRI set'.

[7]Attribution properties are from the POWDER namespace `http://www.w3.org/2007/05/powder-s#` abbreviated hereafter to `wdrs:`

[8]The *Architecture of the World Wide Semantic Web* task group is currently looking into developing an OWL ontology that expresses the meaning of the HTTP request/response interactions specified in RFC 2616. See `http://www.w3.org/2001/tag/awwsw` and, in particular about the potential of POWDER, the report documenting the development of the ontology, `http://www.w3.org/wiki/AwwswVocabulary` retrieved in November 2011.

[9]Created by P. Archer, `http://i-sieve.com/powderprocessor`

[10]Created by A. Perego, `http://dawsec.dicom.uninsubria.it/andrea/ppp`

[11]Available at `http://powder.sourceforge.net`

[12]For brevity, we shall subsequently say 'the `wdrs:matchesregex` property' to mean 'the `wdrs:matchesregex` and `wdrs:notmatchesregex` properties'. We shall explicitly warn the reader when referring to only one or the other.

[13]The string literals that denote pertinent regular expressions can be trivially separated from other string literals by the fact that they appear as `owl:hasValue` fillers in `owl:Restriction`S on the `wdrs:matchesregex` property.

[14]See section *Semantic POWDER Processor* at `http://www.w3.org/2007/powder` and also `http://powder.sourceforge.net`

[15]See `http://jena.sourceforge.net`

[16]See `http://clarkparsia.com/pellet`

[17]See `http://www.openrdf.org`

[18]CISMEF, see `http://www.cismef.org`

[19]See `http://www.intute.ac.uk`

[20]AQUMED, see `http://www.aezq.de`

[21]See `http://www.hon.ch`

[22]See `http://www.urac.org`

[23]See `http://www.comb.es`

[24]`http://www.sync3.eu/rdf/sync3` abbreviated hereafter to `sync3:`

[25]`http://www.loa-cnr.it/ontologies/DUL.owl` abbreviated hereafter to `dul:`

[26]During the POWDER Face-to-Face meeting in Boston, 8-9 November 2007, `http://www.w3.org/blog/powder/2007/11/10/summary_of_face_to_face_meeting_held_dur_2007`