A Mechanism for the Representation of Versions in Linked Administrative Geographic Data

Alex Lohfink[i], Duncan McPhee[ii]

University of Glamorgan, faculty of Advanced Technology, Dept. of Computing and Maths,
Pontypridd,CF37 1DL
Tel. +44443 482950
(alohfink, dmcphee)@glam.ac.uk

Abstract. The Resource Description Framework (RDF) is a base technology of the semantic web, providing a web infrastructure that links distributed resources with semantically meaningful relationships at the data level. RDF is a data model that is composed of simple *subject-predicate-object (or resource-property-value)* "triples", where the predicate represents the semantic link between the subject and its object. Triples combine to form a graph. The referenced resources in RDF triples have unique identifiers called uniform resource identifiers (URIs) that are web-compatible and web-scalable. These identifiers can point to pre-defined predicates, or specific web pages, objects, or other network-accessible resources, leading to less ambiguity and clearer meaning. An issue with RDF data is that resources are subject, as with all data, to evolution through change, and this can lead to linked resources being either removed or outdated. In this paper we describe a model to address such an issue in administrative geography RDF data. The model version-enables such data at the instance (triple) level without disturbing the inherent simplicity of the RDF structure imposed by the triple, and this translates to the querying of data implemented using the model, which retrieve version-specific data by matching simple graph-patterns.

Keywords: Key words: administrative geography; semantic web; versioning; RDF; linked data

1 Introduction

The Resource Description Framework (RDF [1]) has become an important language in the representation of distributed data on the semantic web. It has been successful in representing relationships between web resources at the data level, as opposed to the presentation (web page) level, enabling websites to publish machine-readable information about relationships between distributed resources, rather than relying on relational database-driven web pages that express relationships within queries.

RDF uses Uniform Resource Identifiers, or URIs, to identify resources, their

properties, and property values, on the web. These values are represented by the nodes and arcs of a graph. So, for example, the graph shown in Fig. 1 could be used to represent the statements "there is civil parish identified by http://data.ordnancesurvey.co.uk/id/7000000000000005, which is called Chelmsley Wood, and has the census code 00ct006."

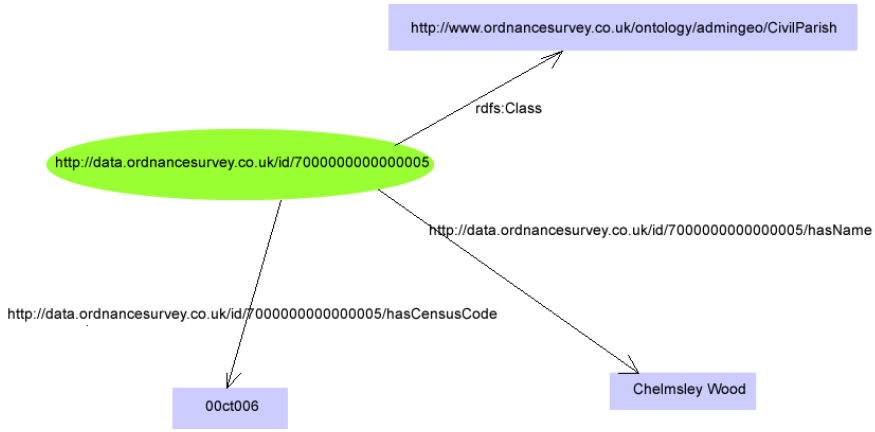

Fig. 1. RDF graph describing Chelmsley Wood

At the heart of the RDF data model is the RDF triple. This is a simple structure based on three parts: subject: predicate: object, or alternatively, resource: property: value. An example of a triple from Fig. 1 would be:

subject: Civil Parish (URI http://data.ordnancesurvey.co.uk/id/7000000000000005)

predicate: hasCensusCode (URI http://data.ordnancesurvey.co.uk/id/7000000000000005/hasCensusCode)

object: literal (00ct006).

This triple represents the statement "The civil parish identified by http://data.ordnancesurvey.co.uk/id/7000000000000005 has the census code 00ct006". Statements can thus be represented and linked to form a directed graph.

The example shown in Fig. 1 is drawn from Ordnance Survey[1]Administrative Geography data. The Ordnance Survey (OS) have published some of this data in RDF format, and an issue with this data is that the administrative units represented change frequently (boundaries, for example, are released twice a year), leading to linked

[1] This research is sponsored by Ordnance Survey

resources being either removed or out-dated. In this paper we describe a mechanism that could solve this problem by version-enabling RDF administrative geography data.

The rest of this paper is organised as follows: in the next section we describe the problem which our model addresses and previous work, in section 3 we discuss some of the possible mechanisms to represent versions in RDF and justify our methodology, in section 4 we describe the model, and in section 5 we describe the querying of data stored in a small dataset based on the model. In section 6 we present our conclusions.

2 The need for versioning in administrative geographic RDF data

Versioning for RDF can be viewed from two perspectives: web ontology versioning (classes) and instance versioning (triples). Instance versioning can be further divided into model-based and statement-based (triple) versioning. Model-based versioning applies to a group of triples that form part of a logical unit. Statement-based versioning applies to individual statements (triples). It is within the field of instance versioning that this paper is concerned. Specifically, we aim to address the aforementioned issue that has arisen in OS Administrative Geography data. Here, the relationship between administrative units in the UK is described by RDF data, but the many changes that have occurred to these units over the years are not adequately represented as there is no logical organisation between different versions of units. Because of this, different users of the administrative geography datasets could link to different versions of administrative units represented in whichever version of the data they are accessing, meaning that inconsistencies will be apparent between different RDF datasets. It would, therefore, be beneficial to provide a versioning mechanism that would allow linkage to a default version of an administrative unit, but allow access to alternative versions if specified. At present, to our knowledge, there is no proposed system or model that provides this capability.

There have been some attempts at introducing version mechanisms to RDF graphs, mainly centred on ontology versioning and determining differences between graphs. The SemVersion [2] model focuses on managing change in ontologies where users can suggest different classes to include in the ontology. SemVersion can manage such changes and reconcile them into a new version of the ontology. SemVersion employs model-based versioning.

Delta [3] is a system designed to identify differences between RDF graphs, and uses functions to compute these differences. Differences between graphs are produced in the form of a delta which represents the changes only. This means that a delta derived from a knowledge base can be applied to a subset of this knowledge base and update it, with accurate results.

Another version model described by [4] uses an extension to the Topic Maps data model [5] to potentially implement versions in RDF. Topic Maps represent topics (or subjects), attributes, and associations as an entity-relationship model. The model described by [4] uses a structure called the VersionInfo Object, or VIO, to record start and end dates for a specific version of a topic map object. This model is stated as

being applicable to RDF triples by grouping triples into logical units and linking them to a VIO.

HTTP content negotiation has been described by [6] as a method to represent versioned resources, where the default linkage is always the current version. Previous versions are timestamped and accessed by specifying a time in the HTTP request, using a *timegate*, which supports date-time content negotiation.

Both Delta and SemVersion are aimed at managing change to web ontologies or specific RDF graphs rather than addressing the need to be able to reference different versions of the same statements or classes within the same RDF dataset. [4] attempts to provide a method for achieving this, by linking logical units of triples to VIOs. In this case, the suggestion is that the VIO would contain start and end dates relevant to the statements in the referenced logical unit, in effect extending the graph to incorporate version objects. However, no implementation is specified, and it is not clear how alternatives would be handled. It also organises VIOs according to a sequence, the organisation of which is not specified. The model described by [6] versions at the resource level, and has the advantage that the default URI is always current. However, this would be problematical if linkage was required to a non-default version. Also, this model only distinguishes versions using date-time values, and would therefore not disambiguate versions which are not distinguished by time (for example, where two identically named, but distinct resources exist at the same time).

3 Possible mechanisms for versioning RDF data

There are several mechanisms within RDF and RDF Schema (RDFS, the specification of the classes and properties of RDF [7]) that offer the potential to model instance-level versions in administrative geography data, and these will now be discussed.

3.1 Inferencing

RDFS allows inferencing based on defined properties such as *subClassOf* and *subPropertyOf* . At the simplest level, this provides a mechanism to create a version hierarchy based on inheritance, where new versions of an item are defined using the *subClassOf* (or 'is_a') relationship. Inferencing allows RDF to infer from the *subClassOf* relationship that the resource is a member of the superclass. For example, In Ordnance Survey Administrative Geography data, a Civil Parish is defined as a *subClassOf* a Civil Administrative Area. It can thus be inferred that the Civil Parish of Chelmsley Wood is also a Civil Administrative Area. This feature provides type propagation, and could be used to define a version hierarchy of RDFS classes.

It is also possible with some RDF implementation environments to define inferencing rules. This kind of inferencing goes beyond the scope of the RDFS inferencing capabilities, and allows the definition of specific, text-based rules by which implicit relationships can be inferred. This could allow version-specific rules to enhance a version hierarchy, such as *version_of, derivative_of, alternative_to* based on criteria derived from the differences between versions of an administrative unit.

3.2 Named graphs

Named graphs allow groups of triples to be identified as belonging to a specified graph within a larger RDF graph. This is achieved by tagging the triples with an identifier that specifies the named graph to which it is associated, in effect making the triples "quads". This means that a group of triples could be coupled together as a named version graph. The RDF query language, called the SPARQL protocol and RDF query language (SPARQL [8]), has a *FROM NAMED* clause which can query named graphs.

3.3 RDF containers

RDF containers provide the capability to represent collections of triples as single entities, and link to them forming new triples. This means that it is possible to model relationships between multiple versions of data by defining appropriate RDF container classes. An RDF container simply uses a blank node from which to link resources that belong to that container. Of particular interest here is the *rdf:Alt* container, which is used to describe a list of alternative values of a resource.

It is our contention that of these three mechanisms, RDF containers provide the most appropriate structures to represent versions. RDFS inferencing provides a simple mechanism to deduce versions, but does not provide any version-specific relationships between versioned resources. Rule-based inferencing, on the other hand, is mainly aimed at getting more meaning from existing relationships between resources, and would require the definition of specific conditions upon which inferences could be made, which would not be expressed within triples. Named graphs disturb the inherent simplicity of the triple by tagging each one with an identifier to identify it with a particular graph, and would also require some kind of logical naming convention to facilitate querying. In addition, relationships between versions within the named graph would still need to be defined, negating the need to name the version graph. RDF containers, on the other hand, provide a simple mechanism that integrates seamlessly into an RDF graph without disturbing its simplicity, and version-specific properties can be defined within this structure. Containers also provide flexibility in the mode of the representation, as they can be defined as rdf:Alt, rdf:Bag, or rdf:Sequence. Rdf:Alt denotes that the contained resources are alternatives, rdf:Bag denotes that there is no significance to the order in which the contained resources are represented, and rdf:Sequence denotes that the contained resources are set in sequential order. It should be noted that there is no implicit behaviour associated with any of these containers, and that these conventions exist to provide consistency in their use.

4 A model for versioning administrative geographic data

The use of the rdf:Alt container to represent versions within an rdf graph is shown in Fig. 2. The model shown represents two versions of a resource described in two

separate RDF datasets produced by the OS. The diagram shows resources as elipses, literals as rectangles, and properties as arcs. URIs use the following prefixes:

dcterms: *http://purl.org/dc/terms/*

admingeo: *http://www.ordnancesurvey.co.uk/ontology/AdministrativeGeography/v2.0/AdministrativeGeography.rdf#*

rdfs: *http://www.w3.org/2000/01/rdf-schema#*

rdf: *http://www.w3.org/1999/02/22-rdf-syntax-ns#*

The dcterms URIs make use of the Dublin Core metadata [9] terms version properties.

The versioned resource has the URI *http://data.ordnancesurvey.co.uk/id/7000000000010769*, and the property rdfs:label with the value of *The London Borough of Bexley*. The property *dcterms:hasVersion* links the versioned resource to the rdf:Alt container, specifically a blank node defined as type *rdf:Alt*. The property *rdf:_1* of this container specifies the alternate version for the *The London Borough of Bexley* resource, which has the URI *admingeo:osr7000000000010759*. By the convention defined by [7], all container members are identified by the properties rdf:_1, rdf:_2, and so on, but the significance of the ordering is defined by the container type, as specified in the previous section. Further versions would be represented using such properties. The property *dcterms:isVersionOf* gives the reverse property, showing which resource this resource is a version of. The versioned resource is given the property *dcterms:isPartOf* to show which administrative authority this version was/is a part of. In this example, it denotes that this version of *The London Borough of Bexley* is part of the Greater London authority which has the URI *admingeo:osr7000000000041441*. Although in the example OS data used in this work the alternative version of *The London Borough of Bexley* is part of the same Greater London authority, the model allows for a version to be part of a different administrative unit.

It can be seen that date values have been linked to each administrative unit resource using the property *dcterms:date*. There is currently little consistency or agreement on the definition of date values that are included in OS administrative data. The values provided here represent the dates on which the datasets were first assembled. Boundary lines for administrative units are typically released twice a year and it is intended that these dates will be included in the data in future releases. Another possibility is that the dates when these boundaries came into existence could be included.

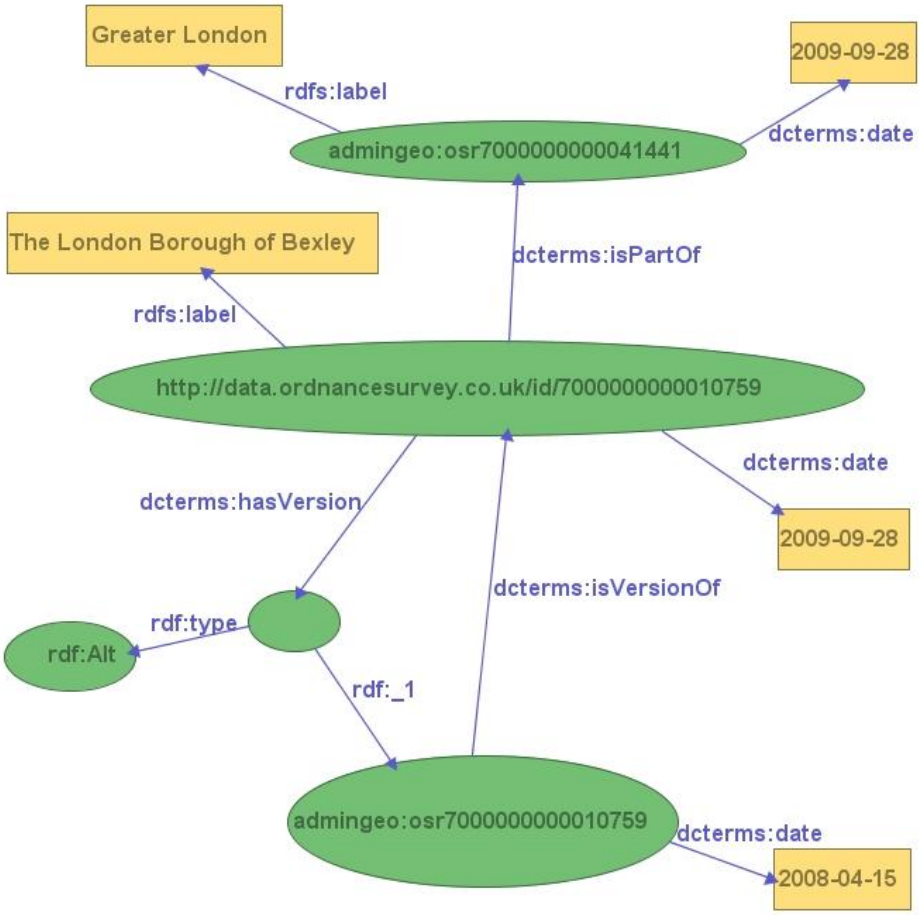

Fig. 2. RDF graph showing the use of the rdf:Alt container to represent two versions of the resource called 'The London Borough of Bexley'.

5 Querying the versioned data

To evaluate querying of RDF data based on our model, we implemented a small dataset consisting of OS administrative data representing a Greater London Authority unit, and two of its constituent boroughs, each with a version. Included were representative properties for each resource.

Querying illustrates some of the proposed benefits of the model. Firstly, because the model uses standard RDF containers, the implemented data could be queried using standard SPARQL queries. Secondly, version-related properties can be retrieved along with other properties, meaning that version-specific predicates need not necessarily be defined in the query. Thirdly, because the inherent simplicity of the data structure imposed by the RDF triple has not been disturbed, version-specific

attributes can be retrieved by matching simple graph patterns.

For example, the following query retrieves all the properties and their values for the resource with the URI *http://data.ordnancesurvey.co.uk/id/7000000000041441* :

```
SELECT ?x ?y
WHERE
{<http://data.ordnancesurvey.co.uk/id/7000000000041441>
?x  ?y};
```

In this example the variable *?x* retrieves all properties including the *hasVersion*, *isVersionOf*, and *isPartOf* properties if present. These can in turn be queried using a similarly simple query. For example, the following query retrieves the version of a resource and its parent version (prefixes are as previously defined):

```
SELECT ?x ?y
WHERE {?x  dcterms:isVersionOf ?y};
```

Here the variable ?x retrieves the version, and the variable ?y retrieves the parent version.

The exception is when querying the *hasVersion* property, which must access the version of the resource via the container, which is a blank node. The query is shown in the following example:

```
SELECT ?y ?z
WHERE {?y  dcterms:hasVersion ?x.?x rdf:_1 ?z};
```

This query retrieves the URIs of a versioned resource, and its first version, represented by the variables *?y* and *?z* respectively. The variable *?x* represents the blank node of the container, and the graph pattern matches two triples, delineated by a period (*?y dcterms:hasVersion ?x and ?x rdf:_1 ?z)*. Replacing the *rdf:_1* value with a variable would retrieve all versions of the resource.

6 Conclusions

In this paper we have given a brief background to RDF, and discussed some of the possible methods which may be adopted for the purposes of versioning RDF at the instance (triple) level in administrative geography data. We have described a versioning model based on RDF containers that addresses a specific issue that has arisen in the RDF representation of administrative geographic data, that is, the requirement of a linkage to a default version of an administrative unit, and logical

access to previous or alternative versions of that unit. The proposed model utilises a standard rdf:Alt container in its structure, and exploits industry standard Dublin Core metadata for its version-specific properties. The model represents versions without interfering with the simplicity of the graph structure imposed by the RDF triple, and this simplicity is evident in the queries, which are able to utilise standard SPARQL syntax and retrieve version-specific properties by matching simple graph patterns.

Future work will involve a larger scale implementation of the model to establish if the model will scale. Also, more precise date definitions would enable the possibility of temporal operations against versions. Although these are not yet adequately supported by SPARQL, certain RDF APIs such as AllegroGraph [10] provide specialised temporal literals and operations.

References

1. W3C. *RDF/XML Syntax Specification (Revised)*. 2004 [cited 3/11/2009]; W3C Recommendation 10 February 2004]. Available from: http://www.w3.org/TR/rdf-syntax-grammar/.
2. Volkel, M. *SemVersion – Versioning RDF and Ontologies*. 2005 [cited 3/11/2009]; Available from: http://semversion.ontoware.org/kwebd233a.pdf.
3. Berners-Lee, T. and D. Connolly. *Delta: an ontology for the distribution of differences between RDF graphs*. 2001 [cited 3/11/2009]; www document]. Available from: http://www.w3.org/DesignIssues/Diff.
4. Ludwig, C., M.W. Kuster, and G. Moore, *Versioning in Distributed Semantic Registries*, in *iiWAS2008*. 2008. p. 493-499.
5. ISO. *Information Technology - Topic Maps - Part 2: Data Model* 2008 [cited 3/11/2009]; Available from: http://www.isotopicmaps.org/sam/.
6. Sompel, H.V.d., et al. *An HTTP-Based Versioning Mechanism for Linked Data*. in *LDOW2010*. 2010.
7. W3C. *RDF Vocabulary Description Language 1.0: RDF Schema*. 2004 [cited 3/11/2009]; W3C Recommendation 10 February 2004]. Available from: http://www.w3.org/TR/rdf-schema/.
8. W3C. *SPARQL Query Language for RDF*. 2008 [cited 3/11/2009]; W3C Recommendation 15 January 2008]. Available from: http://www.w3.org/TR/rdf-sparql-query/.
9. *The Dublin Core Metadata Initiative*. 2010 [cited 2010 26th April]; Available from: http://dublincore.org/.
10. *AllegroGraph RDFStore*. 2009 [cited 3/11/2009]; www document]. Available from: http://www.franz.com/agraph/allegrograph/.

[i] Alex Lohfink is a lecturer at the University of Glamorgan, and gained a PhD there in

December 2008. His current research interests are the semantic web and spatio-temporal databases.

[ii] Duncan McPhee is a senior lecturer at the University of Glamorgan. His research interests are in computer-based learning, databases, data mining, and the semantic web.