

# **SCHEMAS**

**Contract: N° IST-1999-10100**

## **Forum for Metadata Schema Implementers**

**User guide on designing interoperable schemas using RDF**

# **D62**

**Document number:**

**SCHEMAS-GMD-WP6-D62-FINAL-20010809**

<b>General Information</b>
----------------------------

<b>Title</b>	SCHEMAS: User guide on designing interoperable schemas using RDF
<b>Creator</b>	Thomas Baker
<b>Creator</b>	Gauri Salokhe
<b>Subject-Keywords</b>	Deliverable D62; WP6; Formulating and documenting good-practice guidelines on how to use RDF schema assertions to integrate diverse modules in multiple languages; Application profiles
<b>Description</b>	This document provides guidance to implementers on using standard modules to design interoperable schemas and on disclosing these schemas using the RDF format distributed registries.
<b>Publisher</b>	GMD
<b>Date</b>	9 August 2001
<b>Type</b>	Text Manuscript
<b>Format</b>	application/msword
<b>Identifier-</b>	
<b>Document Number</b>	SCHEMAS-GMD-WP6-D62-FINAL-20010809
<b>Language</b>	English
<b>Rights</b>	European Commission; Internal circulation within project; External circulation via SCHEMAS Web site

**Dublin Core Metadata for this document**

```
<META NAME="DC.Title" CONTENT="SCHEMAS: User guide on designing interoperable schemas
using RDF ">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#title">

<META NAME="DC.Creator" CONTENT="Thomas Baker ">
<META NAME="DC.Creator" CONTENT="Gauri Salokhe">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#creator">

<META NAME="DC.Subject" CONTENT="Deliverable D62">
<META NAME="DC.Subject" CONTENT="WP6">
<META NAME="DC.Subject" CONTENT="Formulating and documenting good-practice guidelines
on how to use RDF schema assertions to integrate diverse modules in multiple languages ">
<META NAME="DC.Subject" CONTENT="Application Profiles">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#subject">

<META NAME="DC.Description" CONTENT="This document provides guidance to implementers
on using standard modules to design interoperable schemas and on disclosing these schemas using the
RDF format distributed registries.">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#description">

<META NAME="DC.Publisher" CONTENT="GMD ">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#publisher">

<META NAME="DC.Date" CONTENT="(SCHEME=ISO8601) 2001-08-09">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#date">

<META NAME="DC.Type" CONTENT="Text.Manuscript">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#type">

<META NAME="DC.Format" CONTENT="(SCHEME=IMT) application/msword">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#format">
<LINK REL=SCHEMA.imt HREF="http://sunsite.auc.dk/RFC/rfc/rfc2046.html">

<META NAME="DC.Identifier" CONTENT="http://www.schemas-forum.org/folder/filename">
<META NAME="DC.Identifier" CONTENT="(SCHEME=URN) SCHEMAS-GMD-WP6-D62-
FINAL-20010809">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#identifier">

<META NAME="DC.Language" CONTENT="(SCHEME=ISO639-1) en">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#language">

<META NAME="DC.Rights" CONTENT="European Commission; Internal circulation within project;
External circulation via SCHEMAS Web site">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#rights">

<META NAME="DC.Date.X-MetadataLastModified" CONTENT="(SCHEME=ISO8601) 2001-08-
09">
<LINK REL=SCHEMA.dc HREF="http://purl.org/metadata/dublin_core_elements#date">
```

**Dublin Core metadata for this document**

```
<?xml version="1.0"?>
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:smes="http://www.schemas-forum.org/registry/schemas/SCHEMAS/1.0/smes#"
>
<rdf:RDF rdf:resource = " SCHEMAS-GMD-WP6-D62-FINAL-20010809">
<dc:title> SCHEMAS: User guide on designing interoperable schemas using RDF
</dc:title>
<dc:creator> Thomas Baker </dc:creator>
<dc:creator> Gauri Salokhe </dc:creator>
<dc:subject> Deliverable D62 </dc:subject>
<dc:subject> WP6 </dc:subject>
<dc:subject> Formulating and documenting good-practice guidelines on how to use RDF
schema assertions to integrate diverse modules in multiple languages </dc:subject>
<dc:subject> Application profiles </dc:subject>
<dc:description> This document provides guidance to implementers on using standard
modules to design interoperable schemas and on disclosing these schemas using the RDF
format distributed registries. </dc:description>
<dc:publisher> GMD </dc:publisher>
<dc:date> 2001-08-09 </dc:date>
<dc:type> Text </dc:type>
<dc:format> application/MSWord </dc:format>
<dc:identifier> SCHEMAS-GMD-WP6-D62-FINAL-20010809 </dc:identifier>
<dc:language> en </dc:language>
<dc:rights> European Commission; Public distribution <dc:rights>
</rdf:RDF>
```

## Table of Contents

<b>1</b>	<b>INTRODUCTION</b> .....	<b>6</b>
<b>2</b>	<b>THE CONCEPT OF RDF SCHEMAS</b> .....	<b>6</b>
<b>3</b>	<b>APPLICATION PROFILES</b> .....	<b>6</b>
<b>4</b>	<b>CONTENTS OF AN APPLICATION PROFILE?</b> .....	<b>7</b>
<b>5</b>	<b>RUNNING QUERIES ON RDF PROFILES</b> .....	<b>13</b>
<b>6</b>	<b>MOVING AHEAD WITH GROUPING MECHANISMS</b> .....	<b>14</b>
	<b>REFERENCES</b> .....	<b>16</b>
	<b>APPENDIX</b> .....	<b>17</b>

## 1 Introduction

This deliverable is part of workpackage 6 which addresses the issue of designing interoperable schemas using RDF and multilingual registry. The objective of the multilingual registry is to provide guidance to implementers on using standard modules to design interoperable schemas and on disclosing these schemas using the Resource Description Framework (RDF) format for distributed registries. The emphasis is on using the RDF schema format to link multiple translations of a schema in different languages.

## 2 The concept of RDF Schemas

As part of the more general registry, the Multilingual RDF Registry focuses specifically on the use of the RDF, a new standard for supporting the exchange of metadata on the Web that is up for recommendation by the World-Wide Web Consortium (W3C). One aspect of RDF, called RDF Schemas, is a format for encoding metadata schemas using explicit Web links to related parent schemas. RDF was designed to solve, among other things, the problem of schemas in multiple languages. This capability is particularly useful, indeed crucial, for linking translations of schemas in multiple languages – hence the focus of this registry activity on multilinguality.

The Dublin Core Element Set, which is already available in twenty-five languages, provides a convenient and manageable starting point for this registry. Data-model conventions for expressing Dublin Core schemas in RDF are being discussed and agreed in the Dublin Core Metadata Initiative. The focus in SCHEMAS is on explaining and documenting for the benefit of implementers how to create and use RDF schemas for linking locally designed schemas (in any language) to the standards on which they are based (in English).

## 3 Application Profiles

Application profiles as a type of schema have become topical over the past year or so, but the concept itself is not new. The Z39.50 community, for example, has used "profiles" for constraining potential options and parameter values, left open by standards specifications, to those required by a particular application (e.g., GILS or WAIS), function (e.g., simple author-title-subject searching), or user group (e.g., chemists or musicians). According to the "Framework and Taxonomy of International Standardized Profiles" (ISO TR 10000), a profile specifies how standards, particularly protocols, can be used in combination for meeting such requirements. [1]

In IEEE standardization committees for learning technology, a "standards profile" is "a technique of referencing (in contrast to defining) technical specifications... [permitting] the creation of a bundle of standards, each one tailored, extended, or constrained to meet the needs of the committee developing a standards profile... The point of using standards profiles is to *reuse* existing standards wording without having to recreate the words..." [2]

To users of the Digital Object Identifier, a DOI Application Profile is "the functional specification of an application (or set of applications) of the DOI System to a class of intellectual property entities that share a common set of attributes" for the purpose of

enabling particular applications, from simple resource discovery to complex rights management. [3]

Jane Hunter reports that "Significant new initiatives such as TV-Anytime, MPEG-21, and the Open Archives Initiative are demanding application profiles which combine elements from a number of different existing standardized metadata schemas whilst maintaining interoperability and satisfying their own specific requirements through refinements, extensions and additions." [4] Similarly, the Federal Geographic Data Committee distinguishes between its the Content Standard for Digital Geospatial Metadata, and a profile based on that standard, which "describes the application of the Standard to a specific user community". A profile "always contains the Standard, plus modifications to the optionality or repeatability of non-mandatory elements in the Standard" and "may also contain extended elements"; it may be formalized through the FGDC process or used informally by a user community.[5]

In the European project DESIRE, which developed and tested new techniques for resource discovery and network management between July 1998 and June 2000, an "application profile" was a set of elements with usage information on associated element values, schemes, or controlled vocabularies used for particular projects, computer programs, interchange formats, or information services. In the DESIRE style, an application profile cannot introduce new data elements; it must take each element from an associated namespace. A profile can group together data elements from multiple vocabularies; and it may declare a scheme of valid values appropriate for a particular application. [6,7]

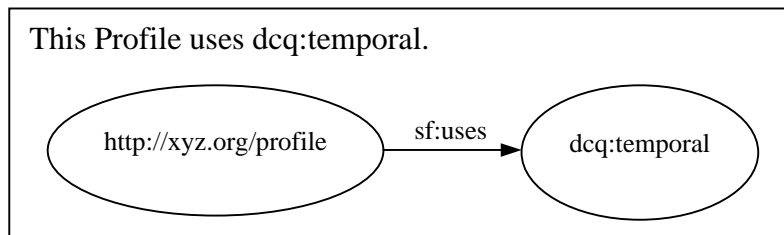
On the basis of this experience, Rachel Heery and Manjula Patel have defined application profiles as "schemas which consist of data elements drawn from one or more namespaces, combined together by implementors, and optimised for a particular local application". By definition, such profiles depend for their elements on namespaces. Namespaces, in this context, are element sets maintained as stable points of reference. They serve to "identify the management authority for an element, support definition of unique identifiers for elements, [and] uniquely define particular data element sets or vocabularies". Management authorities can range from internationally recognized standards bodies, to maintainers of unofficial or de-facto standards, down to projects or services with special data elements defined primarily for local use. [8] This contrast between "namespaces that declare" and "profiles that reuse" provided the starting-point for our discussion of application profiles in the SCHEMAS context.

## **4 Contents of an Application Profile?**

The style of Application Profile presented here is an answer to the question: "What does your metadata say?", or more precisely: "What terms does your metadata use, and how does it use them?". This answer is best described as a set of statements of certain fixed patterns. W3C's Resource Description Framework specification provides the basic grammar of these statements -- a word order of Subject - Predicate - Object, where the Predicate is a verb phrase characterizing the relationship between the Subject and Object.

In practical terms, the sum of such statements is a page or two of XML-formatted metadata looking something like Appendix A (below); this is what gets parsed and indexed by a registry database. But this XML encoding is only intended for consumption by database software (or XML geeks). The logic of the RDF statements

is easier to explain with "node-and-arc" diagrams, where the Subject and Object are nodes and the Predicate is an arc.



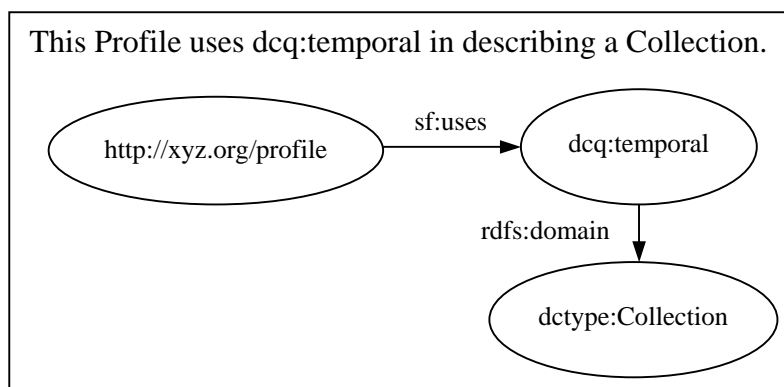
**Figure 1. Use a term from a namespace.**

Figure 1, for example, says in effect: "This profile uses the term Temporal from the DCQ namespace". Note that each part of the statement -- Subject (<http://xyz.org/profile>, in this case the URI of the application profile), Predicate (`sf:uses`), and Object (`dcq:temporal`) -- has a unique Web address, as the prefixes "sf:" and "dcq:" resolve to "<http://www.schemas-forum.org/terms/>" and "<http://purl.org/dc/terms/>" respectively.[9] These addresses identify the namespace schemas where the terms "uses" and "temporal" are declared and defined. Figure 1 represents the most basic statement of an application profile: "This profile uses this term from this namespace". Contents of Figure 1 can be coded into following RDF statements.

```

<!-- Use a term (see Figure 1) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      </rdf:Description>
    </sf:uses>
  </sf:ApSchema>
  
```

Figures 2 to 7 will now illustrate various types of additional information that can be associated with these terms when they are adapted for a particular application environment.



**Figure 2. Specify a class of object to which it refers.**

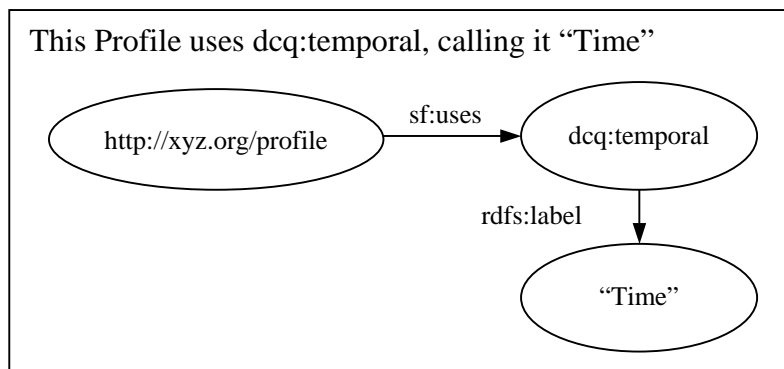
In Figure 2, the Object of the statement in Figure 1 becomes the Subject of a second statement: "This profile uses `dcq:temporal`, and `dcq:temporal` is used specifically in reference to collections". In other words, the metadata is not about "resources" in a

generic sense, but refers specifically to things like manuscript collections, museums, or archives.

```

<!-- Use a term, specify referred class (see Figures 1-2) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection"/>
    </rdf:Description>
  </sf:uses>
</sf:ApSchema>

```

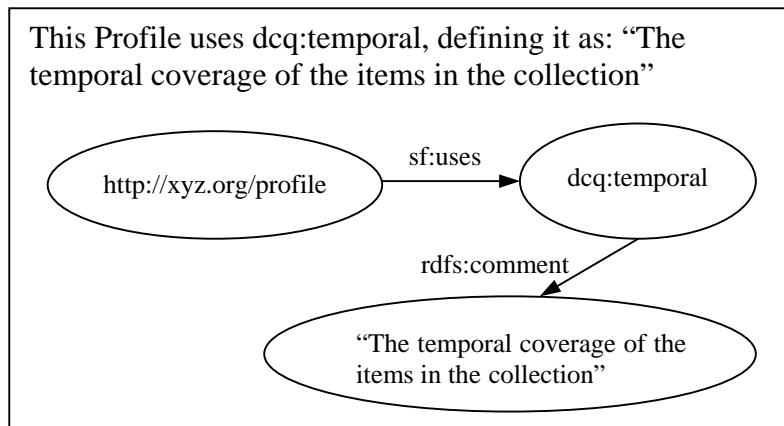


**Figure 3. Override a default with a local label**

```

<!-- Use a term, overriding default label (see Figures 1-3) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection"/>
      <rdfs:label>Time</rdfs:label>
    </rdf:Description>
  </sf:uses>
</sf:ApSchema>

```



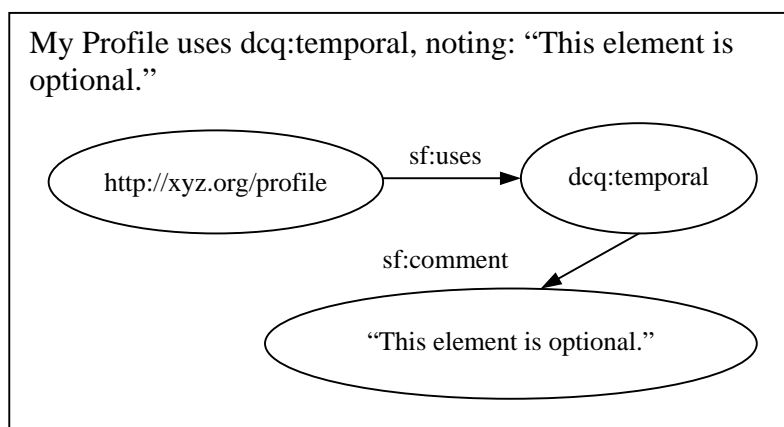
**Figure 4. Override a default with a local definition**

```

<!-- Use a term, overriding default label and definition (see Figures 1-4) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection"/>
      <rdfs:label>Time</rdfs:label>
      <rdfs:comment>The temporal coverage of the items in the collection.
    </rdfs:comment>
    </rdf:Description>
  </sf:uses>
</sf:ApSchema>

```

In the official documentation of its namespace, the Dublin Core qualifier "`dcq:temporal`" is labeled "Temporal" and defined as "Temporal characteristics of the intellectual content of the resource". Figures 3 and 4 show how these "default" labels and definitions can be replaced, or overridden, with labels and definitions that are more appropriate or understandable for users in a particular application context. In this example, the term "`dcq:temporal`" is labelled "Time" and defined as "The temporal coverage of items in the collection". Figure 5 simply adds a local usage guideline ("This element is optional").



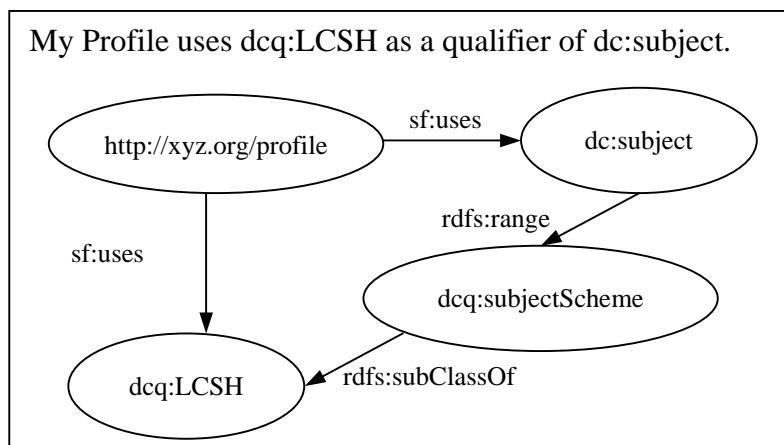
**Figure 5. Add a usage guideline**

```

<! -- Use a term, overriding defaults and adding notes (see Figure 1-5) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection"/>
      <rdfs:label>Time</rdfs:label>
      <rdfs:comment>The temporal coverage of the items in the collection. </rdfs:comment>
      <sf:comment>This element is optional </sf:comment>
    </rdf:Description>
  </sf:uses>
</sf:ApSchema>

```

A profile might also include information about permissible element values. In Figure 6, the profile uses the Dublin Core element Subject (`dc:subject`), and it also uses a qualifier of Subject (`dcq:LCSH`) for specifying that the value of `dc:subject` is a term taken from the Library of Congress Subject Headings (LCSH). (The diagram shows an additional construct: it says that the range of acceptable values for `dc:subject` is restricted to the value set signified by `dcq:subjectScheme`. It then defines `dcq:LCSH` as a sub-set of that value set. In this case, `dcq:LCSH` is related to `dc:subject` through `dcq:subjectScheme` in the DCQ namespace itself, so the additional declaration here may be redundant. This is an example of where clarification is needed, from research and implementation experience on the division of labor between namespaces and profiles.)



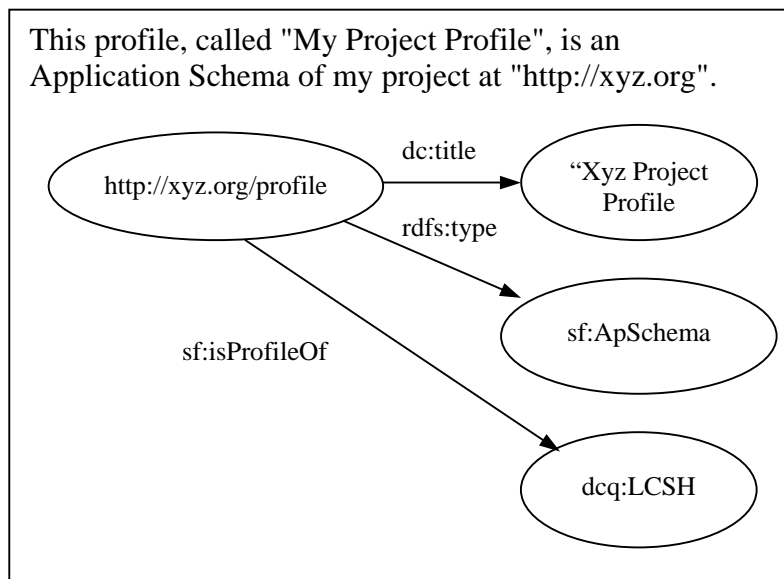
**Figure 6. Use an encoding scheme for a term**

```

<!-- Use an encoding scheme (Figure 6) -->
<sf:ApSchema rdf:about = "http://xyz.org/ profile">
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/elements/1.1/subject" >
      <rdfs:range rdf:resource = "http://purl.org/dc/terms/SubjectScheme" />
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection" />
    </rdf:Description>
  </sf:uses>
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/LCSH">
      <rdfs:subClassOf rdf:resource = "http://purl.org/dc/terms/SubjectScheme"/>
    </rdf:Description>
  </sf:uses>
</sf:ApSchema>

```

Notice that the Subject of the statements in Figures 1 to 6 is the Profile itself. Figure 7 tells us more about the Profile itself: its name ("XYZ Project Profile"), its type ("sf:ApSchema"), and the application of which it is a profile ("http://xyz.org").



**Figure 7. Describe the profile itself, citing the application to which it refers**

```

<!-- Description of this profile (see Figure7) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile">
  <dc:title> Xyz Project Profile </dc:title>
  <sf:isProfileOf rdf:resource= "http://xyz.org" />
  ...RDF on used terms...
</sf:ApSchema>

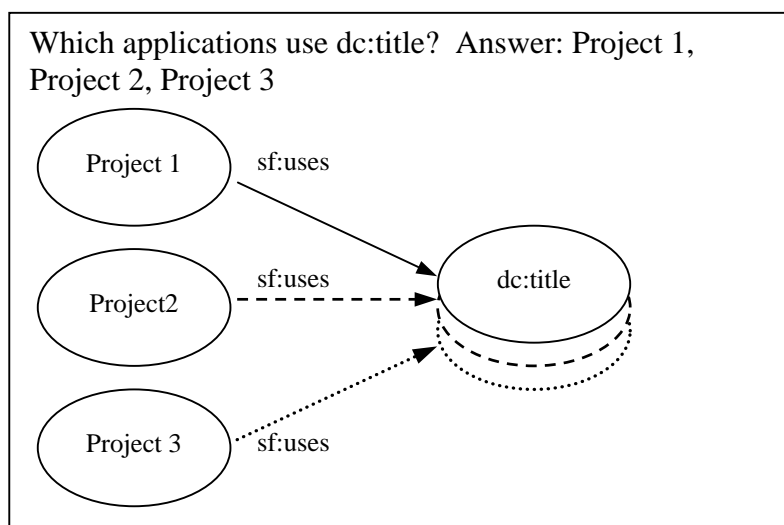
```

Namespace inclusion block should contain the xml version statement and the URI's of namespaces from which the elements of the application profile have been derived.

```
<!-- Namespace inclusion block -->
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
  xmlns:dc = "http://purl.org/dc/elements/1.1/"
  xmlns:dcq = "http://purl.org/dc/terms/"
  xmlns:sf = "http://www.schemas-forum.org/terms" >
```

## 5 Running Queries on RDF Profiles

The practical usefulness of defining a profile as a set of simple sentence patterns is shown by the queries it supports. Creating a searchable index of RDF statements may be pictured as a process of superimposing (joining) multiple statements via their shared nodes. The URIs that associate each part of the sentence with a unique Web address – the Subject (a resource), Predicate (a vocabulary term from a namespace schema), and the Object (another resource or a string literal) – serve as fixed anchor points for merging data from a diversity of sources.

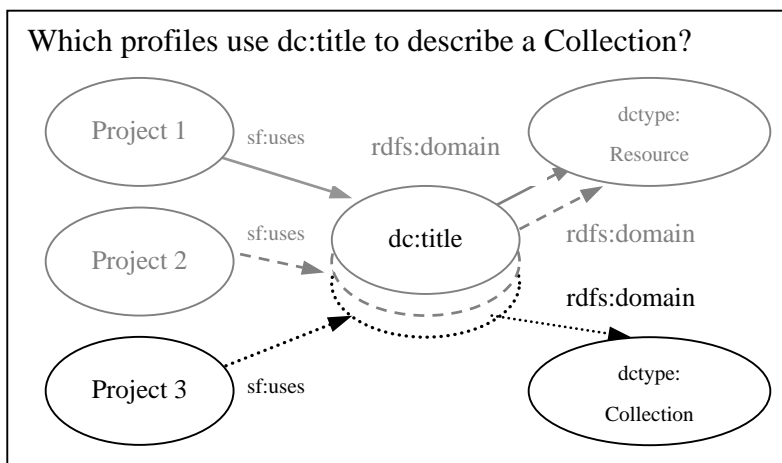


**Figure 8. Joining statements as a basis for queries**

In Figure 8, three sentences sharing `sf:uses` as the Predicate and `dc:title` as the Object yield an answer to the question: "Which applications use `dc:title`?" Figure 9 takes the query one step further and narrows the search result of Figure 8 to those projects that use `dc:title` specifically in reference to collections, as opposed to resources more generically.

The joining of sentences in this manner makes clear that the simple model presented in Figures 1 through 7 may require one further improvement. RDF sentences, also known as triples, stand on their own, and it is through joining that they are placed into a context. If an application profile asserts local labels, definitions, and usage notes to be properties of a term defined in a namespace somewhere, then each such local property will appear in a joined graph as a separate property of the namespace term –

independently of the other local properties associated with that namespace term in a particular profile.

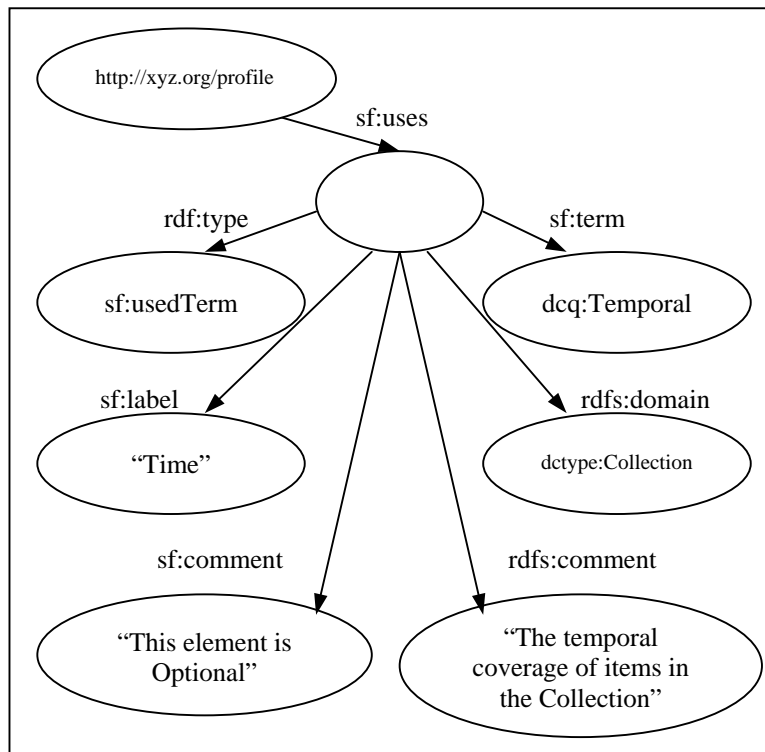


**Figure 9. Narrowing a search**

## 6 Moving ahead with grouping mechanisms

From a modeling point of view, it may be preferable for the triples not to refer directly to a namespace term, but to an entity representing the term "as used" in the local context. This can be done with an "intermediate node" – a modeling construct that groups all of the locally defined properties of a namespace term in a way that allows them to appear as a package when the RDF graphs are joined. In Figure 10 (and Appendix B), the intermediate node is "anonymous" – it does not itself have a unique identifier that would allow it to be referenced as such, in this case as a particular adaptation of dcq:temporal.

In RDF, one can however assign an identifier to the node, giving it in effect a URI and allowing the locally adapted term to be referenced by other metadata like any other namespace term. In principle, this would allow one application profile to use a namespace term indirectly, by using an adapted term from another profile. Whether such practice should be promoted is an open question. It is easy to picture this getting out of hand, with profiles based on profiles based on profiles, threatening semantic drift. But this is perhaps unavoidably an issue in the linguistics of a Semantic Web generally.



**Figure 10. Next step: grouping the properties of a “term used”**

```

<!-- Use a term, overriding defaults and adding notes (see Figure 10) -->
<sf:ApSchema rdf:about = "http://xyz.org/profile ">
  <sf:uses>
    <sf:usedTerm>
      <sf:term rdf:resource = "http://purl.org/dc/terms/temporal" />
      <rdfs:label>Time</rdfs:label>
      <rdfs:comment>The temporal coverage of the items in the collection. </rdfs:comment>
      <sf:comment>This element is optional. </sf:comment>
      <rdfs:domain rdf:resource=" http://purl.org/dc/dcmitype/Collection" />
    </sf:usedTerm>
  </sf:uses>
</sf:ApSchema>

```

## References

- [1] Z39.50 International Standard Maintenance Agency, About Profiles, <http://lcweb.loc.gov/z3950/agency/profiles/about.html>
- [2] IEEE 1484.18: Platform and Media Profiles, <http://edutool.com/pmp/>.
- [3] Digital Object Identifier System, DOI Application Profile, <http://www.doi.org/doi-ap.html>.
- [4] Jane Hunter and Carl Lagoze, Combining RDF and XML Schemas to enhance interoperability between metadata application profiles, May 2001, <http://www10.org/cdrom/papers/572/index.html>.
- [5] Federal Geographic Data Committee, Guidelines for creating a profile for the Content Standard for Digital Geospatial Metadata, 1998, <http://www.fgdc.gov/metadata/csdlgm/profile.html>.
- [6] DESIRE Metadata Registry Framework, March 2000, <http://www.desire.org/html/research/deliverables/D3.5/d35.html>.
- [7] DESIRE Registry Data Model, December 2000, <http://desire.ukoln.ac.uk/registry/docs/datamodel.html>.
- [8] Rachel Heery and Manjula Patel, Application profiles: mixing and matching metadata schemas, *Ariadne 25*, September 2000, <http://www.ariadne.ac.uk/issue25/app-profiles/intro.html>.
- [9] DCMI, Namespace policy for the Dublin Core Metadata Initiative, <http://dublincore.org/documents/dcmi-namespace>.

## Appendix

```
<! -- Namespace inclusion block -->
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
  xmlns:dc = "http://purl.org/dc/elements/1.1/"
  xmlns:dcq = "http://purl.org/dc/terms/"
  xmlns:sf = "http://www.schemas-forum.org/terms" >

  <! -- Description of this profile (see Figure7) -->
  <sf:ApSchema rdf:about = "http://xyz.org/profile">
  <dc:title> Xyz Project Profile </dc:title>
  <sf:isProfileOf rdf:resource= "http://xyz.org" />

  <! -- Use a term, overriding defaults and adding notes (see Figure 1-5) -->
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/temporal">
      <rdfs:label>Time</rdfs:label>
      <rdfs:comment>The temporal coverage of the items in the collection. </rdfs:comment>
      <sf:comment>This element is optional </sf:comment>
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection"/>
    </rdf:Description>
  </sf:uses>

  <! -- Use an encoding scheme (Figure 6) -->
  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/elements/1.1/subject" >
      <rdfs:range rdf:resource = "http://purl.org/dc/terms/SubjectScheme" />
      <rdfs:domain rdf:resource = "http://purl.org/dc/dcmitype/Collection" />
    </rdf:Description>
  </sf:uses>

  <sf:uses>
    <rdf:Description about = "http://purl.org/dc/terms/LCSH">
      <rdfs:subClassOf rdf:resource = "http://purl.org/dc/terms/SubjectScheme"/>
    </rdf:Description>
  </sf:uses>

  </sf:ApSchema>
</rdf:RDF>
```

```
<!-- RDF Code for Figure 8 -->
<!-- Namespace inclusion block -->
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
  xmlns:dcq = "http://purl.org/dc/terms/"
  xmlns:sf = "http://www.schemas-forum.org/terms/">

  <!-- Use a term, overriding defaults and adding notes (see Figure 8) -->
  <sf:ApSchema rdf:about = "http://xyz.org/profile ">
  <sf:uses>
    <sf:usedTerm>
    <sf:term rdf:resource = "http://purl.org/dc/terms/temporal" />
    <rdfs:label>Time</rdfs:label>
    <rdfs:comment>The temporal coverage of the items in the collection. </rdfs:comment>
    <sf:comment>This element is optional. </sf:comment>
    <rdfs:domain rdf:resource=" http://purl.org/dc/dcmitype/Collection" />
    </sf:usedTerm>
  </sf:uses>

  </sf:ApSchema>
</rdf:RDF>
```