SISSVoc: A Linked Data API for access to SKOS vocabularies

Simon J D Cox[a*], Jonathan Yu[a] and Terry Rankine[b]
[a] *CSIRO Land and Water, PO Box 56, Highett, Vic. 3190 Australia*
[b] *CSIRO Mineral Resources, PO Box 1130, Bentley WA, 6102 Australia*
{simon.cox|jonathan.yu|terry.rankine}@csiro.au

Abstract. The Spatial Information Services Stack Vocabulary Service (SISSVoc) is a Linked Data API for accessing published vocabularies. SISSVoc provides a RESTful interface via a set of URI patterns that are aligned with SKOS. These provide a standard web interface for any vocabulary which uses SKOS classes and properties. The SISSVoc implementation provides web pages for human users, and machine-readable resources for client applications (in RDF, JSON, and XML). SISSVoc is implemented using a Linked Data API façade over a SPARQL endpoint. This approach streamlines the configuration of content negotiation, styling, query construction and dispatching. SISSVoc is being used in a number of projects, mainly in the environmental sciences, where controlled vocabularies are used to support cross-domain and interdisciplinary interoperability. SISSVoc simplifies access to vocabularies for end users, and provides a web API to support vocabulary applications.

Keywords: Vocabulary, SKOS, API, Linked data

1. Introduction

Controlled vocabularies are a key element of many classification systems. They are typically published by specific organisations, domains, or communities of practice. The web has encouraged and enabled consolidation of vocabulary use, such that common vocabularies are now more likely to be maintained and published at a community level than only within an agency or project team, thus improving interoperability of scientific datasets. Examples include chemical entities [11,16], bio-medical terminology [30,39], environmental science topic or subject headings [21,24,40] and geological classifications (see compilation at [23]). Vocabularies such as EuroVoc [41] and the International Chronostratigraphic Chart [6] have very well-defined governance and authority, i.e. the Publications Office of the European Union, and the International Commission for Stratigraphy, respectively.

While many vocabularies are openly available on the web, they are formalized and published in a variety of generally incompatible ways, including databases and spreadsheets, text documents, page and image formats. Some of the most fundamental vocabularies are made available on the web by their official custodian only as browser pages or PDFs for download (e.g SI units of measure[1], geologic timescale[2]).

The emergence of *Semantic Web* technologies has provided some powerful tools for formalizing definitions, vocabularies, and ontologies, in forms that also support reasoning and inferencing. In this context, the Simple Knowledge Organization System (SKOS) [20][1] was designed to allow easy formalization of existing multilingual vocabularies that have flat or hierarchical structures, to smooth the transition towards the richer logic-based tools from ontology modelling.

SKOS provides a standard vocabulary for representing thesauri, classifications, taxonomies and controlled vocabularies, using RDF. SKOS has a simple model with few key constructs, focussing on labelling and basic hierarchies. While it lacks the expressivity and rigour of languages such as OWL, its simplicity allows a broad range of vocabularies and classifiers to be ported from a diverse set of formats to RDF, promoting ease of sharing and cross-linking between vocabularies. Many existing vocabularies

[1] http://www.bipm.org/en/si/base_units/ , http://www.bipm.org/en/si/si_brochure/
[2] http://stratigraphy.org/index.php/ics-chart-timescale

have being ported to SKOS [19] including large vocabularies such as AGROVOC [26] and the Library of Congress Subject Headers (LCSH) [35]. SKOS is now one of the most commonly used vocabularies for structured data on the web [19].

Vocabularies are likely to be adopted and shared if they are made available easily. Nevertheless, despite successes in the use of SKOS for *encoding* vocabularies, current standards provide only very low-level *interfaces* to vocabulary data. For example, many vocabularies are published as an RDF document for download. However, if the vocabulary is large then the download will be commensurately large, and if the user only wants to retrieve a single vocabulary term or select a few terms, this option requires processing on the client side. Alternatively, access to vocabularies is often provided at a SPARQL endpoint. SPARQL [15,25] is the generic RDF query language. While this is very powerful, it is a low-level language similar to the relational database query language SQL and normally is only used by database administrators. Some SKOS vocabularies are published via other HTTP interfaces. However, each implementation uses different protocols and supports a varied set of features e.g. content-negotiation provided by the GEMET [42] REST interface, and NERC Data Grid's Vocabulary Server [17,43] SOAP interface. In some cases, one or both of human-readable formats and machine-readable formats is not available. Thus, discovery and access across vocabulary endpoints becomes challenging and ad-hoc.

There is a clear opportunity here, to design an API to match the SKOS vocabulary, taking advantage of the fact that much modern vocabulary content is structured using SKOS classes and predicates. This API can then be used as the basis for various higher level vocabulary applications.

Linked Data has been proposed as a means of publishing and interlinking structured data on the web. Linked Data proposes the use of RDF for describing structured data and allows relationships and links between resources to be defined [4]. This allows both human-readable and machine-readable content/interaction to access data resources and their descriptive metadata using existing web technologies simply by dereferencing HTTP URIs. A number of SKOS vocabulary services are available that utilise Linked Data approaches, such as Semantic Technologies for Archaeological Resources (STAR) Project's semantic terminology services[3], Library of Congress Authorities and Vocabularies service[4], and the Coastal and Marine Spatial Planning Vocabularies (CMSPV) SKOS API[5] [44]. However, each service has a different interface to access the content. Technologies such as Pubby [9], D2R server [3] and Epimorphics Linked Data API Implementation (ELDA) [12] are available for publishing RDF resources as Linked Data, but there is no standard pattern for access to SKOS vocabulary resources. The fundamental issue is that RESTful approaches rely only on URIs, HTTP, and content-types [13,29], yet SKOS is not recognised as a 'content-type' in this context.

In this paper, we describe a standard interface called SISSVoc through which SKOS vocabularies can be provided to web users. SISSVoc provides a level of abstraction for the end users corresponding to the SKOS content model, supporting access to vocabularies without specific knowledge of the underlying technologies and semantic web languages used, such as SPARQL endpoints and queries, SKOS and RDF. A human interface in the form of web pages and forms is provided when HTML is requested. SISSVoc also allows for machine-to-machine use, so that data providers can use HTTP links to vocabularies, data applications can be configured with standard terminology, and data clients can retrieve definitions or verify the existence of items claimed to be in particular vocabularies.

SISSVoc v1 and v3 have been briefly introduced previously [7,14]. In this paper, we present the SISSVoc v3 design in detail and describe the current implementation. We point to its use in some environmental domains, and some client applications built on SISSvoc. We evaluate it in terms of the URI design, and compare it with some other products with similar scope.

SISSVoc is a key component of the Spatial Information Services Stack (SISS) developed by CSIRO through the AuScope project [37].

2. Motivation: different interfaces for different users

Using standard semantic web technologies, a vocabulary can be usefully published through at least four distinct interfaces:

[3] http://hypermedia.research.southwales.ac.uk/resources/terminology/

[4] http://id.loc.gov/search/

[5] http://tw.rpi.edu/web/project/CMSPV/KeyConcepts

1. The complete vocabulary formalized in RDF, formatted using one of the standard RDF serializations (RDF/XML, TTL) and bundled as a **single document** (file), delivered from the "Ontology URI". This is for users and services who wish to harvest the whole vocabulary in one transaction, for local processing. For example
http://sissvoc.ereefs.info/vocab/ereefs/wq

2. At a **SPARQL endpoint,** for access to subsets and views of the vocabulary through the standard RDF query language. This is for expert users, and to support applications that require a highly capable, though low-level interface: e.g.
http://sissvoc.ereefs.info/ereefs/sparql

3. A **vocabulary service** supporting queries on the standard properties of vocabulary items, with options for what is included and the result format. This provides for general users who want to explore a vocabulary without having to know RDF or SPARQL, and to provide an API that insulates developers from SPARQL or from having to load a complete vocabulary: e.g.
- query for all concepts in a vocabulary:
http://sissvoc.ereefs.info/sissvoc/ereefs/concept
- query for concepts broader than those in the vocabulary with the label "nitrogen":
http://sissvoc.ereefs.info/sissvoc/ereefs/concept/broader?anylabel=nitrogen

4. For each **item** in the vocabulary, its {URI} should resolve to a description of the item. This is suitable for direct reference to vocabulary items, and in-line links within datasets. For example, an item in a vocabulary of chemical substances and taxa published on behalf of the Australian government: is denoted
http://environment.data.gov.au/def/object/nitrogen

There are accepted standards for three of these layers, based on generic web technologies (HTTP/URI, RDF, SKOS, SPARQL). SISSVoc has been developed to address the gap at level 3. It has a HTTP-based interface, following RESTful web services [29] and Linked Data [2,4] principles. Standard operations are thus defined as a set of URI patterns. The patterns use the SKOS vocabulary, so as to facilitate discovery and access to resources formalized using SKOS.

3. Design and implementation

3.1. SISSVoc API

SISSVoc provides access to resource descriptions using the following general URI pattern:

```
http://{server}/{vocabulary}/{type}[/{relation}]
   [?{selection-parameters}[&view-parameters]]
```

Tables 1-5 show the details of the various parameters in this pattern, and the corresponding SPARQL queries. A key SISSvoc pattern is the *resource description pattern* (Table 1), in which the description of a SKOS or non-SKOS resource, whose URI is known, is obtained using

```
http://{server}/{vocabulary}/resource
   ?uri={resourceURI}
```

i.e. type== “resource”,
selection-parameters== “uri={resourceURI}”.

Note that in this pattern the resourceURI for a vocabulary item does not necessarily include the SISSVoc server URI.

A basic set of SISSVoc URI patterns provides interfaces to query lists of resources of the SKOS classes. Table 2 lists URI Patterns for querying the set of SKOS ConceptScheme, Collection and Concept respectively.

Another set of SISSVoc URI Patterns provide filtering and selection operations to specific SKOS Concepts. Table 3 lists URI patterns for obtaining a list of concepts based on partial or exact matches on text in labels (rdfs:label, skos:prefLabel, skos:altLabel) for a given vocabulary. Text matching is across all SKOS labels as well as rdfs:label, and is language neutral.

The final set of SISSVoc URI Patterns provides interfaces to allow access to a list of concepts that are related to a selected concept through the predicates defined in SKOS for structuring vocabularies, i.e. the /broader and /narrower properties. Table 4 lists URI patterns for /broader, /broaderTransitive, /narrower, /narrowerTransitive related to a specific SKOS Concept, denoted by its URI. Table 5 lists URI patterns for obtaining a list of concepts that are /broader, /broaderTransitive, /narrower, /narrowerTransitive than SKOS Concepts discovered by the text searches.

Table 1 - SISSVoc URI Pattern for Resource description

ID	URI pattern	Description	SPARQL	Example
1	/resource?uri={URI}	Resource description identified by URI (not limited to any specific type).	DESCRIBE {URI}	http://sissvoc.ereefs.info/sissvoc/ereefs/resource?uri=http://environment.data.gov.au/def/object/nitrogen

Table 2 - SISSVoc URI Patterns for SKOS Concept, ConceptScheme and Collection

ID	URI pattern	Description	SPARQL	Example
2	/conceptscheme	List of all concept schemes	SELECT ?item WHERE { ?item a skos:ConceptScheme }	http://sissvoc.ereefs.info/sissvoc/ereefs/conceptscheme
3	/collection	List of all concept collections	SELECT ?item WHERE { ?item a ?type . FILTER (?type = skos:Collection || ?type = skos:OrderedCollection)}	http://def.seegrid.csiro.au/sissvoc/cgi201211/collection
4	/concept	List of all concepts	SELECT ?item WHERE { ?item a skos:Concept }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept

Table 3 - SISSVoc URI Patterns for SKOS Concept discovery by label

ID	URI pattern	Description	SPARQL	Example
5	/concept?anylabel={text}	List of concepts where a label matches text	SELECT ?item WHERE { ?item a skos:Concept . ?item ?label ?l . FILTER (?label = skos:prefLabel || ?label = skos:altLabel || ?label = skos:hiddenLabel || ?label = rdfs:label) FILTER regex(str(?l) , {text} , 'i') }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept?anylabel=ammonia
6	/concept?labelcontains={text}	List of concepts where a label contains text	SELECT ?item WHERE { ?item a skos:Concept . ?item ?label ?l . FILTER (?label = skos:prefLabel || ?label = skos:altLabel) FILTER regex(str(?l) , {text} , 'i') }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept?labelcontains=ammonia

Table 4 - SISSVoc URI patterns for SKOS Concept broader and narrower by URI

ID	URI pattern	Description	SPARQL	Example
7	/concept/broader?uri={URI}	List of concepts skos:broader than the concept identified by URI	SELECT ?item WHERE { ?item a skos:Concept . {URI} skos:broader ?item }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept/broader?uri=http://environment.data.gov.au/def/property/ammonia_ammonium_concentration
8	/concept/narrower?uri={URI}	List of concepts skos:narrower than concept identified by URI	SELECT ?item WHERE { ?item a skos:Concept . {URI} skos:narrower ?item }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept/narrower?uri=http://environment.data.gov.au/def/property/polyatomic_ion_concentration
9	/concept/broaderTransitive?uri={URI}	List of concepts skos:broaderTransitive than concept identified by URI	SELECT ?item WHERE { ?item a skos:Concept . {URI} skos:broaderTransitive ?item }	http://def.seegrid.csiro.au/sissvoc/isc2014/concept/broaderTransitive?uri=http://resource.geosciml.org/classifier/ics/ischart/Coniacian
10	/concept/narrowerTransitive?uri={URI}	List of concepts skos:narrowerTransitive than concept identified by URI	SELECT ?item WHERE { ?item a skos:Concept . {URI} skos:narrowerTransitive ?item }	http://def.seegrid.csiro.au/sissvoc/isc2014/concept/narrowerTransitive?uri=http://resource.geosciml.org/classifier/ics/ischart/Cretaceous

Table 5 - SISSVoc URI pattern for SKOS Concept discovery broader/narrower by label

ID	URI pattern	Description	SPARQL	Example
11	/concept/broader?anylabel={text}	List of concepts skos:broader than a concept with a label that matches text	SELECT ?item WHERE { ?item a skos:Concept . ?i0 skos:broader ?item . ?i0 ?label ?l . FILTER (?label = rdfs:label || ?label = skos:prefLabel || ?label = skos:altLabel || ?label = skos:hiddenLabel) FILTER regex(str(?l) , {text} , 'i') }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept/broader?anylabel=sulfite%20concentration
12	/concept/narrower?anylabel={text}	List of concepts skos:narrower than a concept with a label that matches text	SELECT ?item WHERE { ?item a skos:Concept . ?i0 skos:narrower ?item . ?i0 ?label ?l . FILTER (?label = rdfs:label || ?label = skos:prefLabel || ?label = skos:altLabel || ?label = skos:hiddenLabel) FILTER regex(str(?l) , {text} , 'i') }	http://sissvoc.ereefs.info/sissvoc/ereefs/concept/narrower?anylabel=non-metal%20concentration
13	/concept/broaderTransitive?anylabel={text}	List of concepts skos:broaderTransiti	SELECT ?item WHERE { ?item a skos:Concept .	http://def.seegrid.csiro.au/sissvoc/isc2014/concept/broaderTransitive?anylabel=Homerian

		ve than a concept with a label that matches text	?i0 skos:broaderTransitive ?item . ?i0 ?label ?l . FILTER (?label = rdfs:label || ?label = skos:prefLabel || ?label = skos:altLabel || ?label = skos:hiddenLabel) FILTER regex(str(?l) , {text} , 'i') }	
14	/concept/narrowerTransitive?anylabel={text}	List of concepts skos:narrowerTransitive than a concept with a label that matches text	SELECT ?item WHERE { ?item a skos:Concept . ?i0 skos:narrowerTransitive ?item . ?i0 ?label ?l . FILTER (?label = rdfs:label || ?label = skos:prefLabel || ?label = skos:altLabel || ?label = skos:hiddenLabel) FILTER regex(str(?l) , {text} , 'i') }	http://def.seegrid.csiro.au/sissvoc/isc2014/concept/narrowerTransitive?anylabel=Cretaceous

Table 6 – Summary of coverage of SKOS vocabulary by current SISSVoc implementations

SKOS Meta-model	**SKOS class/property**	**SISSVoc 3.0**
Class	**Concept, ConceptScheme, Collection**	**Yes**
	OrderedCollection	No
Property (Labels)	**prefLabel, altLabel, hiddenLabel**	**Yes**
Property (Semantic)	semanticRelation, related	No
	broader, narrower	**Yes**
	broaderTransitive, narrowerTransitive	**Yes**
Property (ConceptScheme)	inScheme, topConceptOf, hasTopConcept	No
Property (Collection)	member, memberList	No
Property (notes)	note, changeNote, editorialNote, historyNote, scopeNote	No
	definition, example	No

The exact behavior of a SISSVoc instance depends on the content and behavior of the SPARQL endpoint. For example, the *Transitive queries require that the vocabulary is processed using SKOS inference rules to completion. On the other hand, pattern 3 explicitly includes OrderedCollection, even though it is formally a subclass of Collection. The maintainer of any particular vocabulary content must consider the likely query modes when preparing content.

The SISSVoc API is currently limited to a subset of the SKOS vocabulary and predicates, as required to satisfy specific applications that had emerged in a set of environmental science applications. Table 6 summarizes which elements of the SKOS vocabulary are reflected in the current version of SISSVoc. Nevertheless, a comprehensive SKOS API could easily be developed following the general URI pattern also applied to the other elements of the SKOS vocabulary. And while SKOS data is the application considered here, in principle the general pattern is also extensible to other RDF applications.

SISSVoc is currently specified for HTTP GET operations only.

3.2. SISSVoc implementation

SISSVoc 3.0 was conceived as a Linked Data façade over a vocabulary exposed at a SPARQL endpoint. Use of the Linked Data API [28] streamlines the configuration of content negotiation, styling, query construction and dispatching, and also provides some standard result handling, including paging and language selection.

We have implemented SISSVoc using ELDA [12], an open source implementation of the Linked Data API [28]. An ELDA configuration is bound to a single RDF triple store (shown in Figure 1). Each URI pattern (or "HTTP endpoint") is a few lines in the configuration file, including the corresponding SPARQL query pattern[6]. SISSVoc presents vocabulary content as human-readable resources (HTML), and as machine-readable resources (RDF, JSON, and XML) for client applications, controlled by HTTP content negotiation or by Linked Data API arguments.

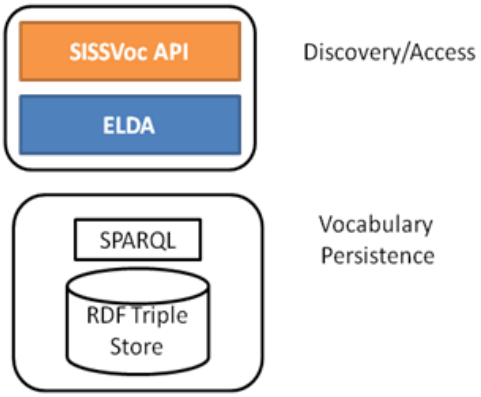

Fig. 1 - SISSVoc Implementation using ELDA

Since the SPARQL endpoint is independent of the SISSVoc deployment, it is not necessary for them to be co-located. This separates concerns with regards to vocabulary maintenance and persistence, and the discovery and access interface. SISSVoc can thus be deployed to provide an interface to any SKOS vocabulary that is published at a SPARQL endpoint. Figure 2 shows an example of this where a SISSVoc deployed at CSIRO[7] provides a standard interface to the NERC Vocabulary Server [17,43] using its SPARQL interface[8].

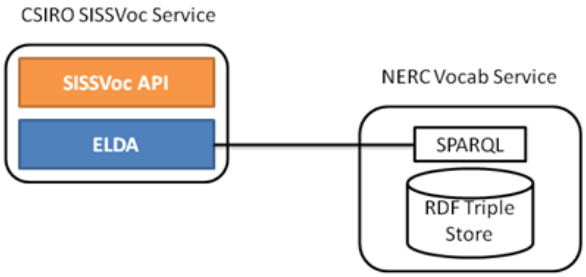

Fig. 2 - Example of a SISSVoc deployment to externally governed vocabulary service

3.3. SISSVoc deployment

In section 2 we included a set of examples to illustrate different vocabulary interfaces in the context of a single actual vocabulary. They are available to be used separately by the different classes of user, but in the context of a vocabulary deployment each interface can, and probably should, use the next one up in

[6] The CSIRO implementation is documented at https://www.seegrid.csiro.au/wiki/Siss/SISSvoc3Overview. Also see https://github.com/jyucsiro/sissvoc-runner for a tool to install locally for testing ELDA configurations.

[7] http://auscope-services-test.arrc.csiro.au/elda-demo/nerc/collection

[8] http://vocab.nerc.ac.uk/sparql/

either configuration or real-time operation. Figure 3 shows how this works in practice in configuration of a typical SISSVoc instance. Resolving the URI for the item in the vocabulary (**interface 4**) involves a call to the SISSVoc API (**interface 3**), which issues a SPARQL query (**interface 2**), to a triple store which was loaded from the vocabulary document (**interface**). The server for vocabulary item URIs is configured to redirect requests to the resource-description URI hosted by this SISSvoc.

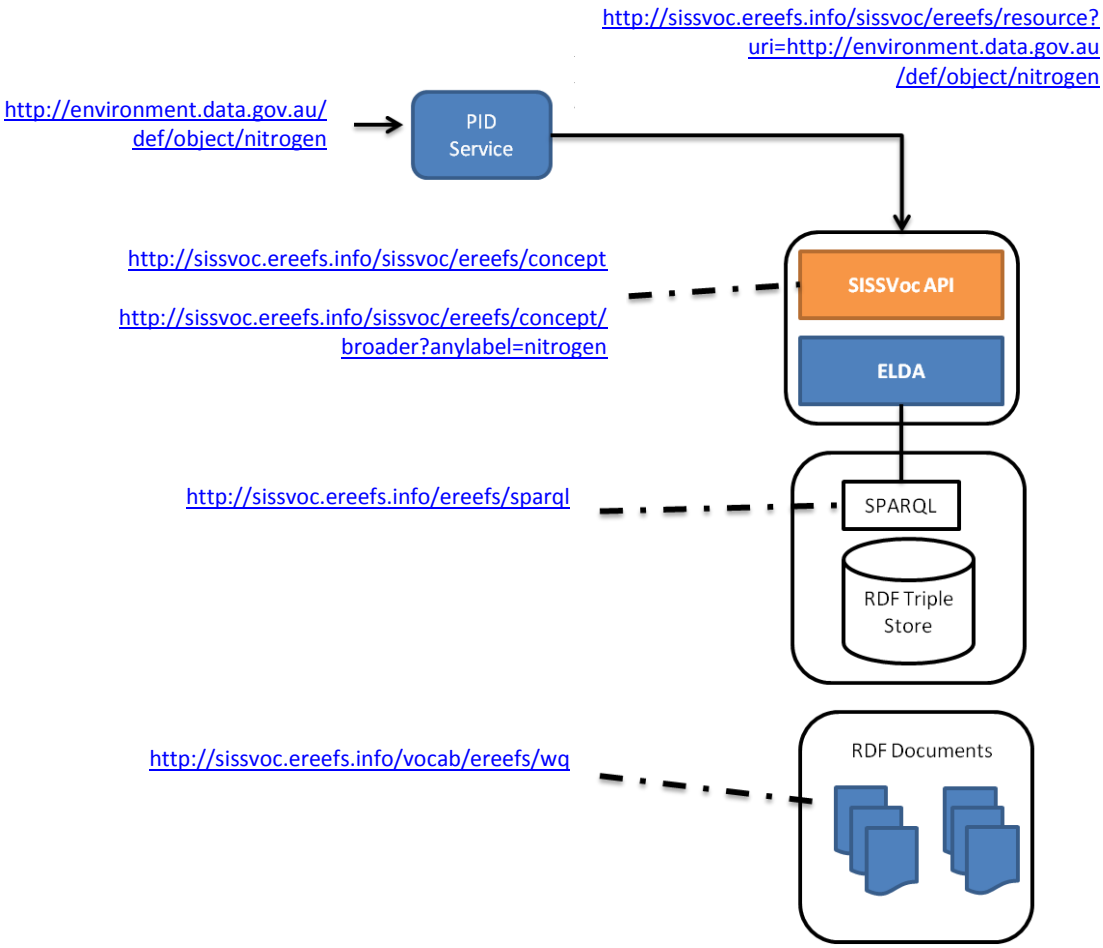

Fig. 3 – A typical SISSVoc Deployment complemented by PID service and a web service hosting RDF documents. Each box represents one of the 4 different interfaces to support the various vocabulary access use cases. An additional component (labeled PID Service[9]) redirects vocabulary URIs to the SISSVoc.

[9] https://www.seegrid.csiro.au/wiki/Siss/PIDServiceUserGuide

3.4. Example Deployments

SISSVoc has been deployed in support of a number of projects, mainly in the earth and environmental sciences, where controlled vocabularies are used to support cross-domain and interdisciplinary interoperability. Examples of vocabularies currently that are being served through various SISSVoc instances are listed in Table 6.

Table 6 -Examples of SISSVoc deployments and their service endpoints

OGC Definitions	http://www.opengis.net/def/
Geological Timescale	http://resource.geosciml.org/classifier/ics/ischart/
Environmental monitoring definitions	http://environment.data.gov.au/def/property/ http://environment.data.gov.au/def/object/ http://environment.data.gov.au/def/unit/
Water and energy supply and consumption (WESC) definitions	http://wescml.org/sissvoc/vocab/collection
ANZSRC Socio-Economic Objective	http://researchdata.ands.org.au:8080/vocab/api/anzsrc-seo/concepts
NERC Vocabularies **(no URI redirection, so links return to the NVS)**	http://vocab.nerc.ac.uk/ via http://auscope-services-test.arrc.csiro.au/elda-demo/nerc/collection

4. SISSVoc Applications

SISSVoc presents a simple interface to any controlled vocabulary that is structured using, or at least decorated with, SKOS classes and properties. A standard vocabulary interface allows listboxes and other User Interface widgets to be populated via HTTP requests to standard vocabularies. It also makes possible the development of common applications such as search clients and validation clients. Here we describe two developed by the authors.

4.1. Water Data Transfer Format validation service

The Water Data Transfer Format (WDTF) is an XML-based exchange standard that was developed for transfer and ingestion of data into the Australian Bureau of Meteorology's information systems from over 200 data providers. A WDTF validation service was implemented using two standard schema languages together with a vocabulary service to check both structure and content (Figure 4) [38].

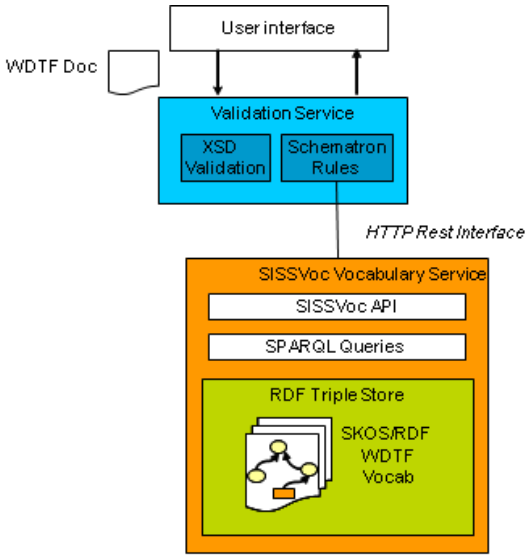

Figure 4 - WDTF Validation Service Leveraging SISSVoc Vocabulary Service

Data structure is validated using a XML schema validation component. Schematron[10] is used to perform co-constraint checking and vocabulary checking. Vocabulary checking includes API calls to a

[10] http://www.schematron.com/

SISSVoc service presenting WDTF SKOS vocabularies.

The listing in Figure 5 shows a XPath function used in the WDTF Validation service to perform vocabulary checking for valid water parameters. The function 'checkParameterExists' calls the function 'getConceptByIdentifier' which uses the SISSVoc resource API to retrieve the resource description for the URI. The function then checks that it has a skos:broader Concept that matches a specified URI which means that the value is in the correct hierarchy (http://www.bom.gov.au/std/water/xml/wio0.2/property/wdtf-parameters/Parameter). The function 'checkParameterExists' is used in a Schematron rule that checks wdtf:TimeSeriesObservation elements, specifically, to verify that the observedProperty is a valid WDTF water parameter.

```
<xsl:function name="wdtffunc:checkParameterExists" as="xs:boolean"
      xmlns:wdtffunc="http://www.csiro.au/wdtf/functions">
  <xsl:param name="parameterUri" as="xs:string"/>
  <xsl:variable name="doc" select="wdtffunc:getConceptByIdentifier($parameterUri)" />
  <xsl:variable name="skosBroaderValue" select="$doc//skos:broader" />
  <xsl:variable name="expectedUri" select="string('
           http://www.bom.gov.au/std/water/xml/wio0.2/property/wdtf-parameters/Parameter')" />
  <xsl:choose>
    <xsl:when  test="$skosBroaderValue[@rdf:resource = $expectedUri]" >
      <xsl:value-of select="true()"/>     </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="false()"/>    </xsl:otherwise>
  </xsl:choose>
</xsl:function>


<xsl:function name="wdtffunc:getConceptByIdentifier">
  <xsl:param name="identifier" as="xs:string"/>
  <!-- SISSVOC_ENDPOINT is the URL for the specific SISSVoc endpoint -->
  <xsl:variable name="queryUrl" select="concat('&SISSVOC_ENDPOINT;/resource?uri=',$identifier)"/>
  <xsl:variable name="doc" select="document($queryUrl)" />
  <xsl:copy-of select="$doc"/>
</xsl:function>

<sch:rule context="wdtf:TimeSeriesObservation/om:observedProperty[(string-length(normalize-
space(@xlink:href)) > 0)">
  <sch:let name="id" value="normalize-space(.)"/>
  <sch:let name="propertyName" value="wdtffunc:getParamValueFromUri($id)"/>
  <sch:let name="location" value="wdtffunc:getLocationMessage(.)"/>
  <sch:let name="isException" value="wdtffunc:isUriException($id)"/>
  <sch:let name="vocabLookup" value="wdtffunc:checkParameterExists($id)
                                  or wdtffunc:checkSurveyTypeExists($id)"/>
  <sch:assert test="$isException or  $vocabLookup" flag="error">
         Parameter ' <sch:value-of select="$propertyName"/>' is not valid
         at <sch:value-of select="$location"/>.
  </sch:assert>
</sch:rule>
```

.Figure 5 - XPath functions and a Schematron rule used to support a validation which uses the SISSVoc API

4.2. SISSVoc Search

SISSVoc Search is built on the SISSVoc API to provide a simple search for vocabulary entries, using terms in the vocabulary labels and descriptions. It includes

- a web-based search interface to support search via HTML form interface (http://sissvoc.ereefs.info/search),
- HTTP GET requests, with the term and SISSVoc endpoint in the query string of the URI (e.g. http://sissvoc.ereefs.info/search?q=water&endpoint=http://wescml.org/sissvoc/vocab).

The user interface allows a user to switch between endpoints, so users may search for vocabulary terms knowing only the SISSvoc deployment URL. Figure 6 shows a screenshot of the user interface. This application has focused on the search use case, though there are some obvious complementary features that could be considered to enhance the user experience of the tool, such as faceted browsing to support discovery of vocabulary terms.

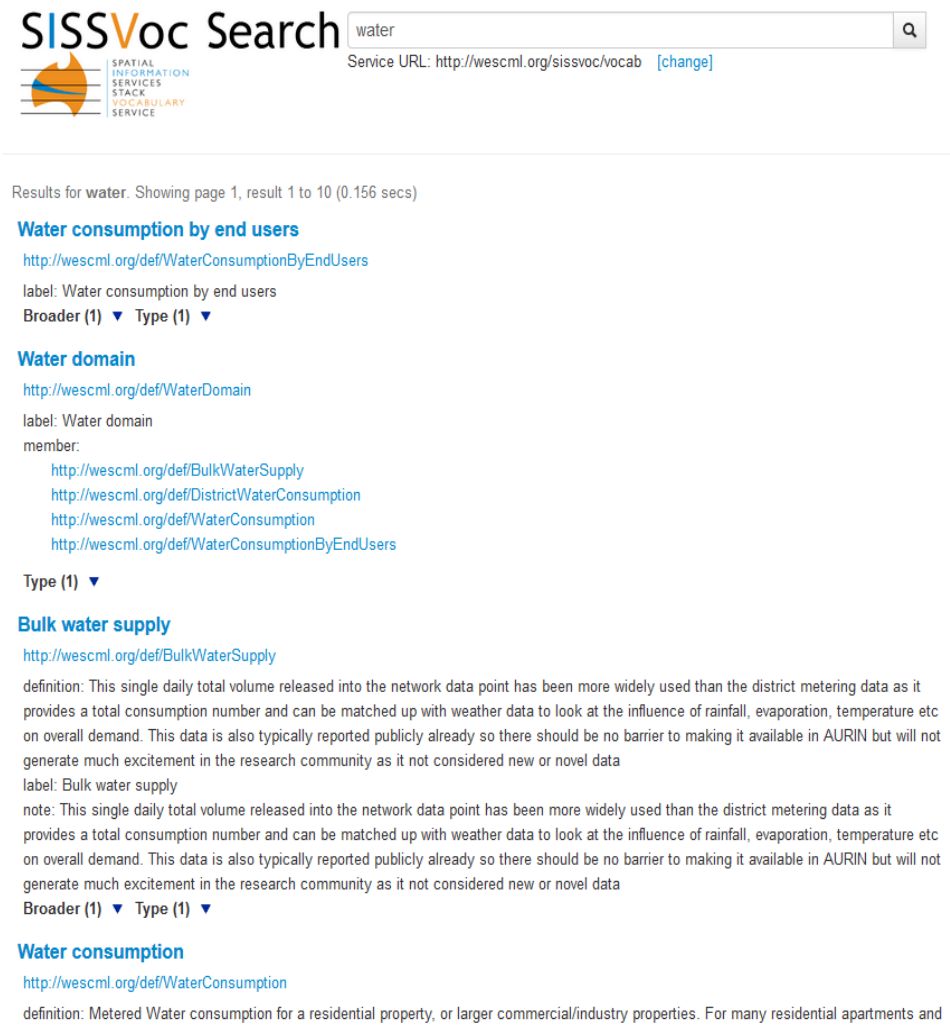

Figure 6 - Screenshot of the SISSVoc search tool

5. Analysis

5.1. URI Pattern for resource descriptions

The URI pattern for resource descriptions is

```
http://{server}/{vocabulary}/resource
          ?uri={resourceURI}
```

This should be read as

> *what http://{server}/{vocabulary} knows about the resource denoted {resourceURI}.*

The pattern is a very explicit implementation of the principles defined in 'Cool URIs for the semantic web' [31], making a very clear distinction between the *information resources* and *non-information resources* involved. In this case

- a 'concept', denoted by {resourceURI}, is understood as an abstract (non-information) resource, which may have various definitions and representations;
- the RDF graph, denoted by the complete resource description pattern, is a corresponding information resource.

A number of URI patterns have been proposed that distinguish between resources and their descriptions [5,10,31,33]. These typically combine special tokens in the URI with HTTP parameters and response codes to indicate to the user how to understand the resource. Two patterns are directly comparable with the SISSVoc resource description pattern, in that they have distinct but related URIs for the non-information resource or concept, and for a description of it:

The Cool URIs for the Semantic Web pattern [10,31,33]:

- http://example.org/id/{id} denotes a non-information resource
- http://example.org/doc/{id} a corresponding description or information resource

In DBPedia [5]:

- http://dbpedia.org/resource/{id} denotes a concept, a non-information resource
- http://dbpedia.org/data/{id} an rdf graph describing the concept, an information resource
- http://dbpedia.org/page/{id} an html page describing the concept, an information resource

The SISSVoc pattern:

- http://example.net/{name} denotes a resource (optionally a non-information resource)
- http://example.org/sissvoc/resource?uri=http://example.net/{name} a description or information resource (format selected through content-negotiation)

The resource description URI pattern does not use the SKOS vocabulary. The properties included in a resource description are those provided by the SPARQL endpoint through its implementation of the DESCRIBE operation, which is typically an approximation of the Concise Bounded Description [34]. Thus, while the list endpoints described in Tables 2-5 use SKOS predicates, the representations return may describe a concept within which the SKOS elements are a minor aspect or 'decoration' of a more specific ontology. The SISSVoc SKOS API is a 'generic' vocabulary access point, and the response may be supplemented by more specific interfaces relevant to a specialized vocabulary. For example, an RDF representation of the 2014 version of the geologic timescale is identified as

```
http://resource.geosciml.org/classifier/ics/ischart/2014
```

and is delivered by a SISSVoc service in the form of a graph that mixes SKOS predicates with predicates from an ontology designed for the geological timescale [8]. The non-SKOS elements can be accessed through the underlying SPARQL endpoint, but are out of scope for the SISSVoc interface.

This vocabulary also links to a number of external sources (including DBpedia [18] and SWEET [27]), but the external content is not used in the SISSVoc interface, which relies on a single SPARQL endpoint.

5.2. URI Patterns for lists

The URI pattern for requesting resources of a particular type is

```
http://{server}/{vocabulary}/{type}
   [?{selection-parameters}[&{view-parameters}]]
```

SISSvoc follows the Linked Data API [28] in using the singular form of the resource type in the URI (i.e. concept, conceptscheme, collection), although the result will sometimes be a list.

5.3. URI Patterns for queries

Earlier versions of SISSvoc[11] took the traditional approach of appending a query-string to a single service URI. In SISSVoc 3.0 each query pattern is a distinct HTTP endpoint. Tables 3-5 focus on relationships between SKOS concepts, so all patterns are based on

```
http://{server}/{vocabulary}/concept/{relation}
    ?{selection-parameters}
```

From a REST point of view, this is oriented around queries, rather than the concepts themselves, but is a necessary consequence of the design goal of separating the service location from concept URIs. As noted above (section 3.3) concept URIs may be redirected to a SISSVoc service if the concept owner endorses a particular service to provide descriptions, but this is strictly a separate concern to SISSVoc which provides a query function. Nevertheless, other redirections could support a REST interface oriented around the concept URIs. For example, a URI redirection pattern

```
{conceptURI}/{relation}  →
http://{server}/{vocabulary}/concept/{relation}
    ?uri={conceptURI}
```

could access concepts related to a primary concept through the {relation} predicate.

SISSVoc development is ongoing in support of a number of environmental science projects, through which this and other refinements of the API are being explored for future versions.

5.4. REST behavior and vocabulary maintenance

The SISSVoc API is currently only specified for HTTP GET operations. It is not a full API, as it does not support HTTP operations for insertion, update and deletion [13,29]. SISSvoc is a lightweight search and retrieval SKOS API. An extensive search did not reveal any similar products which included a full REST API.

Managing vocabulary content is a challenging task, for which the technical aspect involves encoding not only the basic concept description, but also all the relationships within and between vocabularies. Maintaining the integrity of these in the face of fine-grained update operations is a significant task. RDF editors (such as Protégé[12] or TopBraid Composer[13]), ensure that the consistency of relationships between resources is maintained, and support generation of RDF *documents* to transfer a set of vocabulary content from the maintenance to publication environment, as outlined above. For complete web-hosted vocabulary maintenance, as well as publication (commercial) tools like TopQuadrant's Enterprise Vocabulary Net[14], and the PoolParty Thesaurus Server[15] are available.

5.5. Related Work

A number of other vocabulary APIs have been described, either in the context of a single vocabulary or as a general purpose re-usable API aligned with SKOS. In this section, we compare the capabilities of APIs based on similar premises and scope to SISSvoc. Candidates should explicitly leverage the SKOS model, and be a reusable API, so we excluded tools that focus on content management (e.g. Topbraid EVN), or are a service with an API that hosts a specific vocabulary (e.g. GEMET API [42]). A summary comparison matrix is provided in Table 7.

5.5.1. SKOSAPI

SKOSAPI [45] is a Java-based API for SKOS vocabularies. It specializes the OWL API libraries, thus providing logics-based reasoning to applications built on SKOSAPI. Vocabularies can be loaded via HTTP, but interaction with vocabularies is via programmatic interfaces, and SKOSAPI does not provide any web-based APIs for access.

In comparison, while SISSVoc does not explicitly feature logics-based reasoning in the API, this functionality could be included in a SISSVoc deployment by building on triple store implementations which support reasoning (e.g. OWLIM).

5.5.2. ASKOSI

ASKOSI [46] is a framework with coupled SKOS API to vocabularies loaded into a localized datastore (either via files or triple store endpoint). This is primarily to support human-readable web-based views

[11] https://www.seegrid.csiro.au/wiki/Siss/VocabularyService#SISSvoc_versions

[12] http://protege.stanford.edu/

[13] http://www.topquadrant.com/tools/modeling-topbraid-composer-standard-edition/

[14] http://www.topquadrant.com/products/topbraid-enterprise-vocabulary-net/

[15] http://www.poolparty.biz/portfolio-item/poolparty-thesaurus-server/

on the loaded vocabularies and supports machine-readable views in RDF/XML and customized XML.

In comparison, SISSVoc provides a broader set of machine-readable views which includes JSON-LD, and RDF in turtle notation (provided by ELDA). SISSVoc provides a layer of abstraction over a specified SPARQL endpoint, and is decoupled from the datastore. However, ASKOSI provides a broader set of SKOS relationship APIs, also covering the *Match mappings.

5.5.3. Skosprovider

Skosprovider [47] is a Python-based API for creating, loading and interfacing with SKOS vocabularies. A limitation of Skosprovider is that it only allows a mono-hierarchy view of the SKOS Concept model.

5.5.4. Poolparty

Poolparty [32,48] is a commercial thesaurus maintenance and publishing environment. The focus is on interactive content management, and the complete suite provides a range of capabilities, including wiki functionality for community discussion of a thesaurus, some tools for automating creation of a new thesaurus from existing non-SKOS sources, and integration with external semantic-web resources like DBpedia.

The REST API for accessing SKOS content is similar to SISSVoc. For example, the Poolparty URI pattern for 'concepts broader than {conceptURI}' is

```
http://{server}/api/thesaurus/{project}/broaders
   ?concept={conceptURI}
```

which may be compared with the SISSVoc pattern shown in Table 4.

A HTTP POST endpoint is specified to support a 'suggestNewConcept' function, though somewhat surprisingly the submission payload is JSON. Otherwise, update and other maintenance is through a SPARQL Update endpoint.

5.5.5. SKOSMOS

SKOSMOS [22] is closest to SISSVoc, of the products tabulated, as it is an open-source, REST-based SKOS access API. The capabilities of SISSVoc and SKOSMOS are very similar, and implemented using almost identical URI patterns: for example, compare the SKOSMOS pattern for 'concepts broader than {conceptURI}':

```
http://{server}/rest/v1/{vocabulary}/broader
   ?uri={conceptURI}&lang=en
```

with SISSVoc:

```
http://{server}/{vocabulary}/concept/broader
   ?uri={conceptURI}&_lang=en
```

SKOSMOS also supports content/lifecycle management, though not through the REST API.

Note that SKOSMOS is the successor project to ONKI SKOS Server [36]. At the time that SISSVoc design was initiated (2009) ONKI was based on SOAP Web Service technologies with AJAX web interfaces.

Table 7 – Comparison of general purpose SKOS APIs

API / Feature	**SKOS API** [45]	**ASKOSI** [46]	**Skosprovider** [47]	**Poolparty** [48]	**SKOSMOS** [22]	**SISSVoc**
API / Programming Language	Java / OWL API	Java	Python	RESTful HTTP-based API available	PHP Implementation / Fuseki / RESTful HTTP-based (GET)	LDA Implementation / RESTful HTTP-based (GET)
Decoupled from datastore/ SPARQL endpoint	No	Partial (localizes copy of SKOS content via file or SPARQL endpoint)	Partial (localizes copy of SKOS content via file or SPARQL endpoint using RDFLib)	No	Yes	Yes
External SPARQL support	No	Partial	Yes	Yes	Yes	Yes
Open Source License	Apache 2.0	Gnu GPL v3	Yes	No	MIT License	CC-BY (LDA configuration)
SKOS Concept API	Yes	Yes	Yes[16]	Yes	Yes	Yes
SKOS Collection API	Yes	Yes	Yes	Yes	No	Yes
SKOS ConceptScheme API	Yes	Yes	Yes	Yes	Yes	Yes
SKOS relationship API	Yes	Partial (exactMatch, closeMatch, broader, narrower, related, broadMatch, narrowMatch, relatedMatch)	Partial (broader, narrower, related)	Yes	Partial (broader, broaderTransitive, narrower, narrowerTransitive, related)	Partial (broader, broaderTransitive, narrower, narrowerTransitive)
SKOS Label API	Yes	Yes	Yes	Partial	Yes[17]	Partial API for access via label (any label, label contains)
Multilingual	Yes	Yes	Yes	Yes	Yes	Yes (via LDA *_lang* parameter)
Content negotiation	No	SKOS/RDF or XML	No	Yes	Yes	Yes
Content / lifecycle management	Local only	Local only	No	Yes	Yes	No – independent SPARQL endpoint

[16] Single hierarchy
[17] hiddenLabel support not yet implemented

6. Conclusion

SISSvoc provides a lightweight search and retrieval API for RDF datasets based on SKOS. SISSVoc provides an abstracted view for end users and client applications to access and query SKOS vocabulary resources without necessarily knowing any of the underlying technologies and semantic web languages used. The current design of SISSVoc (v3) is based on the Linked Data API and the deployments described are implemented by configuring a Linked Data API endpoint. Since the triple-store hosting the content is coupled to the SISSVoc layer through a (usually public) SPARQL endpoint, the deployment pattern is flexible, and multiple interfaces to the content are usually published, with each used as the basis of the interface next higher in the stack. This supports a range of application approaches. The Linked Data API provides significant capability out-of-the-box, including content negotiation for both human interfaces and machine readable interfaces.

The evaluation includes an analysis of the URI patterns, and a comparison with other SKOS APIs. The three HTTP-based APIs in the comparison have independently converged on essentially identical URI patterns for query, though their current implementations use very different technologies. SISSVoc is probably the lightest-weight, as it is implemented purely by configuring existing components coupled through standard HTTP and SPARQL. We have demonstrated that the availability of a simple REST API to SKOS resources provides a basis for useful applications, in search and validation scenarios.

Acknowledgements

Jackie Stewart (nee Githaiga) (CSIRO) did the development work on earlier versions of SISSvoc; Stuart Williams & Chris Dollin (Epimorphics) provided advice and debugging during the initial ELDA implementation. Particular thanks to four reviewers from Semantic Web Journal whose thorough and constructive comments have led us to make many significant improvements in the paper. SISSVoc development was supported by AuScope under the Australian Government's National Collaborative Research Infrastructure Strategy, and by CSIRO and the Bureau of Meteorology under the Water Information Research and Development Alliance (WIRADA).

References

[1] T. Baker, S. Bechhofer, A. Isaac, A. Miles, G. Schreiber, E. Summers, Key choices in the design of Simple Knowledge Organization System (SKOS), Web Semant. Sci. Serv. Agents World Wide Web. 20 (2013) 35–49 DOI:10.1016/j.websem.2013.05.001.

[2] T. Berners-Lee, Linked Data - Design Issues, *W3C Design Issues*. [Online]. Available at: http://www.w3.org/DesignIssues/LinkedData.html. [Accessed: 13-Feb-2014].

[3] C. Bizer, R. Cyganiak, D2r server-publishing relational databases on the semantic web, in: 5th Int. Semant. Web Conf., 2006: p. 26.

[4] C. Bizer, T. Heath, T. Berners-Lee, Linked Data - The Story So Far, Int. J. Semant. Web Inf. Syst. 5 (2009) 1–22 DOI:10.4018/jswis.2009081901.

[5] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, et al., DBpedia - A crystallization point for the Web of Data, Web Semant. 7 (2009) 154–165 DOI:10.1016/j.websem.2009.07.002.

[6] K.M. Cohen, S.C. Finney, P.L. Gibbard, J.-X. Fan, (updated), The ICS International Chronostratigraphic Chart, Episodes. 36 (2013) 199–204.

[7] S.J.D. Cox, K. Mills, F. Tan, Vocabulary services to support scientific data interoperability, in: Geophys. Res. Abstr. Proc. EGU Gen. Assem., European Geoscience Union, 2013.

[8] S.J.D. Cox, S.M. Richard, A geologic timescale ontology and service, Earth Sci. Informatics. (2014) DOI:10.1007/s12145-014-0166-2.

[9] R. Cyganiak, C. Bizer, Pubby – A Linked Data Frontend for SPARQL Endpoints. [Online]. Available at: http://wifo5-03.informatik.uni-mannheim.de/pubby/. [Accessed: 13-Feb-2014].

[10] P. Davidson, Designing URI Sets for the UK Public Sector, London, 2009, Available at: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/60975/designing-URI-sets-uk-public-sector.pdf.

[11] K. Degtyarenko, P. de Matos, M. Ennis, J. Hastings, M. Zbinden, A. McNaught, et al., ChEBI: a database and ontology for chemical entities of biological interest., Nucleic Acids Res. 36 (2008) D344–50 DOI:10.1093/nar/gkm791.

[12] Epimorphics, Elda: the linked-data API in Java. [Online]. Available at: http://www.epimorphics.com/web/tools/elda.html. [Accessed: 13-Feb-2014].

[13] R.T. Fielding, R.N. Taylor, Principled design of the modern Web architecture, ACM Trans. Internet Technol. 2 (2002) 115–150 DOI:10.1145/514183.514185.

[14] J. Githaiga, G. Duclaux, S.J.D. Cox, J. Yu, Spatial Information Services Stack (SISS) Vocabulary Service – A Tool For Managing Earth & Environmental Sciences Controlled Vocabularies., in: eResearch Australasia, 2010.

[15] S. Harris, A. Seaborne, SPARQL 1.1 Query Language, World Wide Web Consortium, 2013, Available at: http://www.w3.org/TR/sparql11-query/.

[16] J. Hastings, P. de Matos, A. Dekker, M. Ennis, B. Harsha, N. Kale, et al., The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013., Nucleic Acids Res. 41 (2013) D456–63 DOI:10.1093/nar/gks1146.

[17] A. Leadbetter, R. Lowry, D.O. Clements, The NERC Vocabulary Server: Version 2.0, in: Geophys. Res. Abstr. Proc. EGU Gen. Assem., European Geoscience Union, 2012.

[18] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mende, et al., DBpedia – A Large-scale , Multilingual Knowledge Base Extracted from Wikipedia, Semant. Web J. 1 (2012) 1–29.

[19] N.A.A. Manaf, S. Bechhofer, R. Stevens, The Current State of SKOS Vocabularies on the Web, in: Semant. Web Res. Appl. Lect. Notes Comput. Sci., Springer, Berlin Heidelberg, 2012: pp. 270–284.

[20] A. Miles, S. Bechhofer, SKOS Simple Knowledge Organization System Reference, World Wide Web Consortium, 2009, Available at: http://www.w3.org/TR/skos-reference/.

[21] K.S. Nagendra, O. Bukhres, S. Sikkupparbathyam, M. Areal, Z.B. Miled, L.M. Olsen, et al., NASA Global Change Master Directory: an implementation of asynchronous management protocol in a heterogeneous distributed environment, in: 3rd Int. Symp. Distrib. Objects Appl., IEEE Comput. Soc, 2001: pp. 136–145.

[22] National Library of Finland, Skosmos REST API. [Online]. Available at: https://github.com/NatLibFi/Skosmos/wiki/REST-API. [Accessed: 25-Aug-2014].

[23] G. Ogg, Geologic TimeScale Foundation - Stratigraphic Lexicons. [Online]. Available at: https://engineering.purdue.edu/Stratigraphy/resources/lexicons.html. [Accessed: 13-Feb-2014].

[24] L.M. Olsen, G. Major, K. Shein, J. Scialdone, S. Ritz, T. Stevens, et al., NASA/Global Change Master Directory (GCMD) Earth Science Keywords. Version 8.0.0.0.0. [Online]. Available at: http://gcmd.nasa.gov/learn/keyword_list.html. [Accessed: 13-Feb-2014].

[25] E. Prud'hommeaux, A. Seaborne, SPARQL Query Language for RDF, World Wide Web Consortium, 2008, Available at: http://www.w3.org/TR/rdf-sparql-query/.

[26] S. Rajbhandari, J. Keizer, The AGROVOC Concept Scheme : A Walkthrough, J. Integr. Agric. 11 (2012) 694–699 DOI:10.1016/S2095-3119(12)60058-6.

[27] R.G. Raskin, M.J. Pan, Knowledge representation in the semantic web for Earth and environmental terminology (SWEET), Comput. Geosci. 31 (2005) 1119–1125 DOI:10.1016/j.cageo.2004.12.004.

[28] D. Reynolds, J. Tennison, L. Dodds, I. Dickinson, linked-data-api - API and formats to simplify use of linked data by web-developers. [Online]. Available at: https://code.google.com/p/linked-data-api/. [Accessed: 13-Feb-2014].

[29] L. Richardson, S. Ruby, RESTful Web Services, O'Reilly Media, 2008, Available at: http://medcontent.metapress.com/index/A65RM03P4874243N.pdf.

[30] F.B. Rogers, Medical Subject Headings (MeSH), Bull. Med. Libr. Assoc. 51 (1963) 114–116 DOI:10.1038/205236a0.

[31] L. Sauermann, R. Cyganiak, Cool URIs for the Semantic Web. [Online]. Available at: http://www.w3.org/TR/cooluris/. [Accessed: 13-Feb-2014].

[32] T. Schandl, A. Blumauer, The Semantic Web: Research and Applications, in: L. Aroyo, G. Antoniou, E. Hyvönen, A. ten Teije, H. Stuckenschmidt, L. Cabral, et al. (Eds.), Semant. Web Res. Appl. Lect. Notes Comput. Sci., Springer Berlin Heidelberg, Berlin, Heidelberg, 2010: pp. 421–425.

[33] J. Sheridan, J. Tennison, Linking UK Government Data, in: C. Bizer, T. Heath, T. Berners-Lee, M. Hausenblas (Eds.), Linked Data Web Work., CEUR, Raleigh, North Carolina, USA, April 27, 2010, 2010.

[34] P. Stickler, CBD - Concise Bounded Description, World Wide Web Consortium, 2005, Available at: http://www.w3.org/Submission/CBD/.

[35] E. Summers, A. Isaac, C. Redding, D. Krech, LCSH, SKOS and Linked Data, Web Semant. Sci. Serv. Agents World Wide Web. 20 (2013) 35–49 DOI:10.1016/j.websem.2013.05.001.

[36] J. Tuominen, M. Frosterus, K. Viljanen, E. Hyvönen, ONKI SKOS Server for Publishing and Utilizing SKOS Vocabularies and Ontologies as Services, Semant. Web Res. Appl. Lect. Notes Comput. Sci. 5554 (2009) 768–780 DOI:10.1007/978-3-642-02121-3_56.

[37] R. Woodcock, B.A. Simons, G. Duclaux, S.J.D. Cox, Auscope's use of standards to deliver earth resource data, in: Geophys. Res. Abstr. Proc. EGU Gen. Assem., European Geoscience Union, 2010.

[38] J. Yu, S.J.D. Cox, G. Walker, P.J. Box, P. Sheahan, Use of standard vocabulary services in validation of water resources data described in XML, Earth Sci. Informatics. 4 (2011) 125–137 DOI:10.1007/s12145-011-0084-5.

[39] ICD-10-CM - International Classification of Diseases, Tenth Revision, Clinical Modification. [Online]. Available at: http://www.cdc.gov/nchs/icd/icd10cm.htm. [Accessed: 13-Feb-2014].

[40] GEMET Thesaurus. [Online]. Available at: http://www.eionet.europa.eu/gemet/. [Accessed: 27-Mar-2014].

[41] Eurovoc, the EU's multilingual thesaurus. [Online]. Available at: http://eurovoc.europa.eu/. [Accessed: 13-Feb-2014].

[42] GEMET WebService API. [Online]. Available at: http://taskman.eionet.europa.eu/projects/zope/wiki/GEMETWebServiceAPI. [Accessed: 13-Feb-2014].

[43] NERC Vocabulary Server version 2.0 (NVS2.0) at BODC. [Online]. Available at: http://vocab.nerc.ac.uk/. [Accessed: 13-Feb-2014].

[44] Coastal and Marine Spatial Planning Vocabularies. [Online]. Available at: http://tw.rpi.edu/web/project/CMSPV. [Accessed: 13-Feb-2014].

[45] SKOS API. [Online]. Available at: http://skosapi.sourceforge.net/documentation.html. [Accessed: 28-Aug-2014].

[46] ASKOSI. [Online]. Available at: http://www.destin-informatique.com/ASKOSI/. [Accessed: 28-Aug-2014].

[47] Skosprovider. [Online]. Available at: http://skosprovider.readthedocs.org/en/latest/api.html. [Accessed: 28-Aug-2014].

[48] PoolParty API - Guide - Thesaurus Services. [Online]. Available at: https://grips.semantic-web.at/display/POOLDOKU/Thesaurus+Services. [Accessed: 28-Aug-2014].