NERC Vocabulary Server version 2.0

Author: Adam Leadbetter (alead@bodc.ac.uk)
Intended audience: NERC Vocabulary Server Version 2.0 users

Document revision history

Revision	Revision date	Summary of changes
1.1	20/06/207	Modified to more accurately describe the GetRelatedConcepts method [QTL]
1.0	13/04/2012	First issue

Contents

Introduction

The NERC Vocabulary Server provides access to groupings of standardised terms that cover a broad spectrum of disciplines of relevance to the oceanographic and wider environmental sciences communities.

Using standardised sets of terms (otherwise known as "controlled vocabularies") in metadata and to label data solves the problem of ambiguities associated with data markup and also enables records to be interpreted by computers. This opens up data sets to a whole world of possibilities for computer aided manipulation, distribution and long term reuse.

An example of how computers may benefit from the use of controlled vocabularies is in the summing of values taken from different data sets. For instance, one data set may have a column labelled "Temperature of the water column" and another might have "water temperature" or even "temperature". To the human eye, the similarity is obvious but a computer would not be able to interpret these as the same thing unless all the possible options were hard coded into its software. If data are marked up with the same term, this problem is resolved.

In the real world, it is not always possible or agreeable for data providers to use the same terms. In such cases, controlled vocabularies can be used as a medium through which data centres can map their equivalent terms.

The controlled vocabularies delivered by the NERC Vocabulary Server contain the following information for each term:

- Key — a compact permanent identifier for the term, designed for computer storage rather than human readability
- Label — the text string representing the term in human-readable form
- Abbreviation — a concise text string representing the term in human-readable form where space is limited
- Definition — a full description of what is meant by the term

Both labels and definitions may be delivered in multiple human-readable languages.

All of the vocabularies are fully versioned and a permanent record is kept of all changes made.

Version 2.0 of the server (NVS2.0) represents a complete rewrite of the internal software with both increased functionality and performance. Although the V1.1 code will remain operational for the foreseeable future, existing users are urged to convert to 2.0 as soon as possible and the development of new V1.1 applications is strongly discouraged. The V1.0 and V1.1method specifications are maintained as historical documents.

Terminology

W3C	The World Wide Web Consortium, the main international standards organisation for the World Wide Web
RDF	The Resource Description Framework is a family of W3C specifications for making statements about resources on the World Wide Web in the form of "subject-predicate-object" expressions, known as triples.
SKOS	Simple Knowledge Organization System. A W3C recommendation for the representation of knowledge in a format understandable to computers. SKOS is built on top of RDF.
Concept	A SKOS concept can be viewed as an idea or notion; a unit of thought. The notion of a SKOS concept is useful when describing the conceptual or intellectual structure of a knowledge organization system, and when referring to specific ideas or meanings established within that system.
Concept Collection	A concept collection is useful where a group of concepts shares something in common, and it is convenient to group them under a common label. In NVS2.0, concept collections are synonymous with controlled vocabularies or code lists.
Concept Scheme	A concept scheme can be viewed as an aggregation of one or more SKOS concepts. Semantic relationships (links) between those concepts may also be viewed as part of a concept scheme. A concept scheme is therefore useful for containing the concepts registered in multiple concept collections but which are related to each other as a single semantic unit, such as a thesaurus.
API	An Application Programming Interface is specification intended to be used as an interface by software components to communicate with each other
ReST / ReSTful	Representational State Transfer is a design of API in which web services are viewed as resources and can therefore be identified by their Uniform Resource Locators (URLs).
SOAP	Is a design of API for exchanging structured information across computer networks as the result of calls to web services. It relies upon XML (eXstensible Markup Language) documents for passing messages.

Connectivity

Consumers may access the vocabulary server either using the ReSTful URLs described below or via SOAP.

SOAP consumers should generate their client implementation from the WSDL available at http://vocab.nerc.ac.uk/vocab2.wsdl.

Collection, concept and scheme URIs

Collections, concepts and schemes are presented to the server as Uniform Resource Identifiers (URIs) (in this case actually URLs) having the syntax

Collections:	http://vocab.nerc.ac.uk/collection/
	http://vocab.nerc.ac.uk/collection/colRef/colVer/
	http://vocab.nerc.ac.uk/collection/colRef/colVer/status/
Concepts:	http://vocab.nerc.ac.uk/collection/colRef/colVer/conRef/
Schemes:	http://vocab.nerc.ac.uk/scheme/
	http://vocab.nerc.ac.uk/scheme/schemeRef/

where

http://vocab.nerc.ac.uk/collection/ and *http://vocab.nerc.ac.uk/scheme/* respectively provide catalogues of the available concept collections and concept schemes.

colRef is an internal opaque identifier for the concept collection, e.g. `P02` for the SeaDataNet Parameter Discovery Vocabulary.

colVer may be a valid concept collection version number or '`current`' to specify the latest version of the collection.

status may be '`all`', '`accepted`' or '`deprecated`' to indicate whether all concepts related to a collection should be returned, or only the accepted or deprecated concepts.

conRef is an internal opaque identifier for the concept within the concept collection, e.g. `TEMP` for 'Temperature of the water column' in the SeaDataNet Parameter Discovery Vocabulary.

schemeRef is an internal opaque identifier for the concept scheme, e.g. `ICANCOERO` for the International Coastal Atlas Network Coastal Erosion Thesaurus.

ReSTful and SOAP API Method Details

GetCollections

The `GetCollections` method allows the client to retrieve a list of the available SKOS concept collections from NVS2.0. This allows a client to discover the content of NVS2.0 which is available through the concept collection paradigm.

API	***Method Call Details***
ReST	**Base URL:** http://vocab.nerc.ac.uk/ **URL suffix:** collection/ **Example fully encoded URL:** http://vocab.nerc.ac.uk/collection/ **Returns:** A SKOS concept collection RDF XML document
SOAP	**Method:** getCollections **Input Parameters**: No Parameters needed **Returns:** ConceptCollection complex data type

GetConceptCollection

The `GetConceptCollection` method allows the client to retrieve all of the available metadata and all of the concepts and associated information for a given SKOS concept collection identified by its URL.

API	***Method Call Details***
ReST	**Base URL:** http://vocab.nerc.ac.uk/collection/ **URL suffix:** collectionID/versionID/status versionID is optional. If it is omitted, the versionID defaults to “current”, which may also be used as a valid versionID, and returns the most up to date version of the concept collection. status is also optional. If omitted the status defaults to “all” which returns all concepts registered to the specified concept collection. Other values for status which are valid are “accepted” and “deprecated”. **Example fully encoded URLs:** http://vocab.nerc.ac.uk/collection/A01/ http://vocab.nerc.ac.uk/collection/C19/2/ http://vocab.nerc.ac.uk/collection/A01/current/ http://vocab.nerc.ac.uk/collection/A01/current/all/ http://vocab.nerc.ac.uk/collection/A01/current/accepted/ http://vocab.nerc.ac.uk/collection/A01/current/deprecated/ **Returns:** A SKOS concept collection RDF XML document
SOAP	**Method:** getConceptCollection(*collectionURL,status*) **Input Parameters:** *collectionURL*: String - concept collection URL e.g. http://vocab.nerc.ac.uk/collection/A01/ *status*: String of value “all”, “accepted” or “deprecated” **Returns:** ConceptCollection complex data type

GetConcept

The `GetConcept` method allows the client to retrieve all available information about a given concept, identified by its URL.

API	***Method Call Details***
ReST	**Base URL:** http://vocab.nerc.ac.uk/collection/ **URL suffix:** collectionID/versionID/conceptID versionID may either be the string "current" to return the most up to date version of the concept, or an integer number to return the version of the concept from a given version of the concept collection. **Example fully encoded URLs:** http://vocab.nerc.ac.uk/collection/C18/current/72/ http://vocab.nerc.ac.uk/collection/A01/current/Human_Responses_to_Coastal_Change **Returns:** A SKOS concept RDF XML document
SOAP	**Method:** GetConcept(*conceptURL*) **Input Parameters:** *conceptURL*: String - concept URL e.g. http://vocab.nerc.ac.uk/collection/C18/current/72/ **Returns:** concept complex data type

GetSchemes

The `GetSchemes` method allows the client to retrieve a list of and the descriptions of the concept schemes available through NVS2.0.

API	***Method Call Details***
ReST	**Base URL:** http://vocab.nerc.ac.uk/ **URL suffix:** scheme/ **Example fully encoded URL:** http://vocab. nerc.ac.uk/scheme/ **Returns:** A SKOS concept scheme RDF XML document
SOAP	**Method:** GetSchemes **Input Parameters**: No Parameters needed **Returns:** ConceptScheme complex data type

GetConceptScheme

The `GetConceptScheme` method allows the client to retrieve all of the available metadata and all of the concepts and associated information for a given SKOS concept scheme, as identified by its URL.

API	***Method Call Details***
ReST	**Base URL:** http://vocab.nerc.ac.uk/scheme/ **URL suffix:** schemeID/ **Example fully encoded URL:** http://vocab.nerc.ac.uk/scheme/ICANCOERO/ **Returns:** A SKOS concept scheme RDF XML document
SOAP	**Method:** GetConceptScheme(*schemeURL*) **Input Parameters:** *schemeURL*: String - concept Scheme URL e.g. http://vocab.nerc.ac.uk/scheme/ICANCOERO/ **Returns:** ConceptScheme complex data type

GetRelatedConcepts

The `getRelatedConcepts` method allows the client to access all of the concepts which are related to a given concept, identified by that concept's URL.

A relationship type mask is provided to the method call to determine which types of relationship are returned by the method call. This will allow, for example, the selection of only narrower matches or only broader matches facilitating relationship tree building in client interfaces. The mask is built from four flags where each flag value may be 1 or 0 to determine if the relationship should be returned or not. e.g.:

broader	narrower	sameAs	related
0	0	1	0

searches only for related concepts which are synonyms to the specified concept.

The method returns a representation of the input concept along with individual concept records of the related concepts.

This method is not available through the ReST API, only through the SOAP API.

API	*Method Call Details*
ReST	This method is unavailable through the ReST API.
SOAP	**Method:** getRelatedConcepts(*conceptURL*, *relationshipType*, *status*) **Input Parameters:** *conceptURL*: String - concept URL e.g. http://vocab.nerc.ac.uk/collection/P01/current/PSALCU01/ *relationshipType*: String indicating level of relationship to return as defined above e.g. "0010" *status*: String of value "all", "accepted" or "deprecated" **Returns:** RelatedConcepts complex data type

GetTopConcepts

The `getTopConcepts` method allows the client to access the concepts which are explicitly stated to be the entry points of a given SKOS concept scheme, identified by its URL.

This method is not available through the ReST API, only through the SOAP API.

API	***Method Call Details***
ReST	This method is unavailable through the ReST API.
SOAP	**Method:** getTopConcepts(*schemeURL*) **Input Parameters:** *schemeURL*: String - concept Scheme URL e.g. http://vocab.nerc.ac.uk/scheme/ICANCOERO/ **Returns:** A list of Concept complex data type objects enclosed by <getTopConcepts><topConcepts>…</topConcepts></getTopConcepts> tags

SearchVocab

The `searchVocab` method allows the client to search the knowledge encoded within NVS2.0. The input incorporates:

This method is not available through the ReST API, only through the SOAP API. Note, there is no guaranteed consistency to the order in which the concepts are returned.

API	***Method Call Details***
ReST	This method is unavailable through the ReST API.
SOAP	**Method:** searchVocab(*query*, *case_sensitivity*, *term_type*, *max_results*, *multilang*, *uri_list*, *status*) **Input Parameters:** *query***:** The search term to be acted on. Valid wildcard characters are: * = 1 or more characters e.g. Searches for "Salinity*" and "*alinity*" on http://vocab.nerc.ac.uk/collection/P01/current/ will yield different result sets. *case_sensitivity***:** Optional Boolean value: *true* or *false*. Default action is *false*. *term_type*: String of value "uri", "preflabel" or "altlabel" *max_results*: An optional integer to limit the number of resturned results *multilang***:** Optional Boolean value to search on non-English labels: *true* or *false*. Default action is *false*. This option is included to significantly reduce the response time for searches in which multilingual functionality is not required. *uri_list*: A list of the concept collection URLs to search e.g. http://vocab.nerc.ac.uk/collection/P01/current/,http://vocab.nerc.ac.uk/collection/P02/current *status*: String of value "all", "accepted" or "deprecated" **Returns:** SearchResults complex data type

VerifyConcept

The `verifyConcept` method is used to check the existence of a given concept within NVS2.0, as identified by its URL, its preferred label or its alternative label. This is of particular use to a client that is validating the markup of its metadata or data. The return of this method is a Boolean value, equal to `true` if the concept in question exists in NVS2.0 and `false` if it does not.

This method is not available through the ReST API, only through the SOAP API.

API	***Method Call Details***
ReST	This method is unavailable through the ReST API.
SOAP	**Method:** verifyConcept(*concept*, *collectionURI*, *conceptType*, *status*) **Input Parameters:** String - concept Collection URL, String – Concept label or URL *concept*: String of one of the following: The full URL to the concept to be verified – use with *conceptType* = *"*uri" e.g. http://vocab.nerc.ac.uk/collection/P01/current/PSALCU01 or the concept preferred label or alternative label to be verified – use with *conceptType="*preflabel*" or conceptType="*altlabel" e.g. "Practical salinity of the water body by CTD and computation using UNESCO 1983 algorithm and NO calibration against independent measurements" collectionURI: String - The URL to the concept collection against which the concept should be verified e.g. http://vocab.nerc.ac.uk/collection/P01/ *conceptType*: String of value "uri", "preflabel" or "altlabel" *status*: String of value "all", "accepted" or "deprecated" **Returns:** Boolean value, i.e.: <verifyConcept><verified>true</verified></verifyConcept> <verifyConcept><verified>false</verified></verifyConcept>

ReSTful Interface XML Payload Details

The ReSTful URL access methods return XML documents conforming to the W3C's Simple Knowledge Organization System model. The use of XML tags outside the scope of that specification within these payload documents is explained in detail below.

Namespaces used

DC	http://purl.org/dc/elements/1.1/
GRG	http://www.isotc211.org/schemas/grg
OWLXML	http://www.w3.org/2006/12/owl2-xml#
RDF	http://www.w3.org/1999/02/22-rdf-syntax-ns#
RDFS	http://www.w3.org/2000/01/rdf-schema#
SKOS	http://www.w3.org/2004/02/skos/core#
XSD	http://www.w3.org/2001/XMLSchema#

skos:Collection

A SKOS concept collection is a group of related concepts. Each controlled vocabulary served from NVS2.0 are formalised in their representation as a `skos:Collection`. Each concept which is a member of a `skos:Collection` is enclosed by a `skos:member` tag.

dc:title and skos:prefLabel

- Mandatory
- Number
    - One per concept collection document

The Dublin Core metadata element set provides the `dc:title` tag to present the formal name given to a resource. In this instance, the `dc:title` and `skos:prefLabel` tags will carry the title of the concept collection.

dc:alternative and skos:altLabel

- Optional
- Number
    - One per concept collection document

Where a resource has more than one title by which it is known, the `dc:alternative` and `skos:altLabel` tags provides a method of encoding the alternative titles.

dc:description

- Optional
- Number
    - One per concept collection document

Often, the formal name (or names) of a resource cannot carry enough information to make the resource both discoverable and usable. In this case, a plain text description of the resource can aid in the usage of the resource. In this case, the account of the content of the resource shall be contained within `dc:description` tags.

dc:date

- Mandatory
- Number
    - One per concept collection document

The Dublin Core metadata element `dc:date` allows the inclusion of an important point in the lifecycle of the resource. In this case we use the time and date of creation of the version of the concept collection requested.

```
<dc:date>
        2011-05-31T08:00:20.136+0000
</dc:date>
```

owlxml:versionInfo

- Mandatory
- Number
    - One per concept collection document

The `owlxml:versionInfo` tag gives the published version number of the concept collection.

```
<owlxml:versionInfo>
        2
</owlxml:versionInfo>
```

dc:creator

- Mandatory
- Number
    - One or more per concept collection document

The Dublin Core metadata elements provide the creator element defined as “the entity primarily responsible for making the resource”. The Dublin Core guidelines give examples of a creator as “include a person, an organization, or a service. Typically, the name of a Creator should be used to indicate the entity.” Therefore a `dc:creator` tag will be used to store the content governance body for a given concept scheme.

```
<dc:creator>
        SeaVox: SeaDataNet and MarineXML Vocabulary Content Governance Group
</dc:creator>
```

grg:RE_RegisterOwner

- Mandatory
- Number
  - One or many per concept collection document

The `grg:RE_RegisterOwner` tag allows the inclusion of ISO19135 compatible information regarding the person or body who owns a concept collection.

```
< grg:RE_RegisterOwner>
      SeaVox: SeaDataNet and MarineXML Vocabulary Content Governance Group
</ grg:RE_RegisterOwner>
```

rdfs:comment

- Optional for inclusion in concept collection documents
- Number
  - Zero, one or many per concept collection document

An RDF Schema (RDFS) comment is added to the NVS2.0 payload in order to provide further information about the body in charge of the content governance for the concept collection or concept scheme.

```
<rdfs:comment>
      Group set up under the joint auspices of the SeaDataNet project and the
      Intergovernmental Oceanographic Commission MarineXML Steering Group for
      controlled vocabulary governance in the marine domain
</rdfs:comment>
```

grg:RE_RegisterManager

- Mandatory
- Number
  - One per concept collection document

The `grg:RE_RegisterManager` tag allows the inclusion of ISO19135 compatible information regarding the person or body appointed by a register owner to manage a register.

```
< grg:RE_RegisterManager>
      British Oceanographic Data Centre
</ grg:RE_RegisterManager>
```

dc:publisher

- Mandatory

- Number

    - One per concept collection document

The `dc:publisher` tag allows the inclusion of the publisher of the resource

```
<dc:publisher>
      Natural Environment Research Council
</dc:publisher>
```

skos:scheme

SKOS concept schemes represent an aggregation of concepts with interconnecting semantic relationships. A concept scheme is likely to contain a hierarchy, so the SKOS collections (or parts thereof) in NVS2.0 which form thesauri may be grouped together and formalised as concept schemes. The definition of a SKOS concept scheme gives the entry points to the broadest concept definitions within the hierarchy, which are referred to as the top concepts. Each top concept is also declared to be the top concept of a concept scheme, and each concept member of a concept scheme is declared to be a member of each scheme to which it belongs.

dc:title and skos:prefLabel

- Mandatory
- Number
    - One per concept scheme document

The Dublin Core metadata element set provides the `dc:title` tag to present the formal name given to a resource. In this instance, the `dc:title` and `skos:prefLabel` tags will carry the title of the concept scheme.

dc:alternative and skos:altLabel

- Optional
- Number
    - One per concept scheme document

Where a resource has more than one title by which it is known, the `dc:alternative` and `skos:altLabel` tags provides a method of encoding the alternative titles.

dc:description

- Optional
- Number
    - One per concept scheme document

Often, the formal name (or names) of a resource cannot carry enough information to make the resource both discoverable and usable. In this case, a plain text description of the resource can aid in the usage of the resource. In this case, the account of the content of the resource shall be contained within `dc:description` tags.

dc:date

- Mandatory
- Number
  - One per concept scheme document

The Dublin Core metadata element `dc:date` allows the inclusion of an important point in the lifecycle of the resource. In this case we use the time and date of creation of the current version of the concept scheme.

```
<dc:date>
       2011-05-31T08:00:20.136+0000
</dc:date>
```

owlxml:versionInfo

- Mandatory
- Number
  - One per concept scheme document

The `owlxml:versionInfo` tag gives the published version number of the concept scheme.

```
<owlxml:versionInfo>
       2
</owlxml:versionInfo>
```

dc:creator

- Mandatory
- Number
  - One per concept scheme document

The Dublin Core metadata elements provide the creator element defined as “the entity primarily responsible for making the resource”. The Dublin Core guidelines give examples of a creator as “include a person, an organization, or a service. Typically, the name of a Creator should be used to indicate the entity.” Therefore a `dc:creator` tag will be used to store the content governance body for a given concept scheme.

```
<dc:creator>
       SeaVox: SeaDataNet and MarineXML Vocabulary Content Governance Group
</dc:creator>
```

rdfs:comment

- Optional for inclusion in concept scheme documents
- Number
    - Zero or one per concept scheme document

An RDFS comment is added to the NVS2.0 payload in order to provide further information about the body in charge of the content governance for the concept collection or concept scheme.

```
<rdfs:comment>
      Group set up under the joint auspices of the SeaDataNet project and the
      Intergovernmental Oceanographic Commission MarineXML Steering Group for
      controlled vocabulary governance in the marine domain
</rdfs:comment>
```

dc:publisher

- Mandatory
- Number
    - One per concept scheme document

The `dc:publisher` tag allows the inclusion of the publisher of the resource

```
<dc:publisher>
      Natural Environment Research Council
</dc:publisher>
```

skos:concept

- Mandatory
- Number
  - One if the payload is in response to a request for a concept by identifier (URI)
  - Many if the payload is returned in response to a request for a list or thesaurus or a request for a concept using a query parameterized by anything other than a concept's URI

A `skos:Concept` is the base unit of currency within the NERC Vocabulary Server, on which other units such as lists and thesauri are built. Therefore each NVS response to a request shall return at least one concept. Each `skos:Concept` opening tag shall also contain the URL to the concept as an `rdf:about` subtag, e.g.:

```
<skos:concept rdf:about="http://vocab.nerc.ac.uk/collection/collid/ver/concept/">
</skos:concept>
```

Each `skos:Concept` may have associated annotations (including human-language translations), mappings, concept collections, concept schemes and provenance information.

skos:prefLabel

- Mandatory
- Number
  - A maximum of one per human readable language into which the concept has been translated

A `skos:Concept` returned from NVS2.0 shall have a preferred label in at least one human readable language. The `skos:prefLabel` is to contain the preferred human readable representation of the concept.

```
<skos:prefLabel xml:lang="en">
        Adriatic Sea
</skos:preLabel>
```

skos:altLabel

- Optional
- Number
    - Zero or one per concept

The `skos:altLabel` element can be used to provide alternative spellings or synonyms to a given concept, or to provide a lexical label for use in alternative circumstances, e.g. axes labels in plotting software.

```
<skos:altLabel xml:lang=”en”>
       Haloc_WC
</skos:altLabel>
```

skos:definition

- Mandatory
- Number
    - One per human readable language into which the concept has been translated

The `skos:definition` tag is used to carry supporting information which describes a concept in greater detail than is carried in the human readable title of the concept enclosed in `skos:prefLabel`. For concepts which require structured information to be carried with them, the contents of the `skos:definition` tag may be encoded as a JavaScript Object Notation (JSON) string.

```
<skos:definition xml:lang="en">
       {"country": "Italy",
       "platformclass": "self-propelled boat",
       "callsign": "IMNQ",
       "length": "19.05",
       "built": "1977",
       "notes": "Leased tugboat"}
</skos:definition>
```

dc:identifier and skos:notation

- Mandatory
- Number
    - One per concept

Version 1.X of the NVS uses the `skos:externalID` property to define the SeaDataNet Uniform Resource Name (URN) of a given concept. However, this property was deprecated in 2004, and the recommended replacement is the `dc:identifier` property from the Dublin Core metadata element set. `dc:identifier` is defined as "an unambiguous reference to the resource within a given context" which fits the usage of the property to declare the SeaDataNet URN or any other external identifiers given to a concept. The `skos:notation` tag is defined as a character string, not normally recognizable as a word or sequence of words in any human readable language, used to uniquely identify a concept within the scope of a concept scheme. This formal scope restriction is the reason that both `dc:identifier` and `skos:notation` tags are used for the same content.

```
<dc:identifier>
        SDN:C191::3_1_2_4
</dc:identifier>

<skos:notation>
        SDN:C191::3_1_2_4
</skos:notation>
```

dc:date

- Mandatory
- Number
    - One per concept

The Dublin Core metadata element `dc:date` allows the inclusion of an important point in the lifecycle of the concept. In this case we use the time and date of creation of this version of the concept.

```
<dc:date>
        2011-05-31T08:00:20.136+0000
</dc:date>
```

owlxml:versionInfo

- Mandatory
- Number
    - One per concept scheme document

The `owlxml:versionInfo` tag gives the version number of the concept included in the document.

```
<owlxml:versionInfo>
      2
</owlxml:versionInfo>
```

skos:note

- Mandatory
- Number
  - One per concept

The `skos:note` tag is designed to allow ancillary information about a SKOS concept. In the context of the payload documents under discussion here it is used to define a concept's publication status. A concept may be "accepted", "proposed" or "deprecated". This value is set respectively according to whether it has been accepted by the content governance body, it is being considered by the content governance body or the concept has been deprecated by the governance body.

```
<skos:note xml:lang=”en”>
      accepted
</skos:note>
```

owlxml:deprecated

- Mandatory
- Number
  - One per concept

The `owlxml:deprecated` tag encloses a Boolean value indicating if the concept has been deprecated ("false") or not ("true").

```
<owlxml:deprecated>
      true
</owlxml:deprecated>

<owlxml:deprecated>
      false
</owlxml:deprecated>
```

Multi-lingual provisioning

The encoding of which human language a SKOS annotation tag is written in should follow the World Wide Web Consortium guidelines of language encoding in XML. These recommendations state that the language tags from the Internet Assigned Numbers Authority (IANA) repository should be used with a hierarchy of

Primary language – extended language – script – region – variant – extension – private use

For the purposes of the NERC Vocabulary Server, a primary language encoding is the deepest the hierarchy need go.

```
<skos:prefLabel xml:lang=”en”>
       colour
</skos:preLabel>

<skos:prefLabel xml:lang=”fr”>
       couleur
</skos:preLabel>
```

Mappings

- Optional
- Number
    - Zero or many per concept or concept collection

Mappings, or semantic relations, indicate the links that a given concept has to another concept. Mappings in the original version of the SKOS specification could be broader, narrower, exact or close. However, in the latest version of the specification the concepts of exact and close matches have been superseded by the related tag.

Broader relations indicate that the current concept has a narrower definition than the concept to which it is related, narrower relations imply the inverse, and close matches imply that the two concepts are more loosely coupled.

Broader and narrower matches may also have the transitive property associated with them, which allows the use of a semantic inference engine. When a thesaurus is delivered through concept scheme in NVS2.0, the mappings internal to that thesaurus are supplied as transitive and those external to the thesaurus are supplied as non-transitive.

```
<skos:narrower rdf:resource=”http://a/Term/Url” />

<skos:narrowerTransitive rdf:resource=”http://a/Term/Url” />

<skos:broader rdf:resource=”http://a/Term/Url” />

<skos:broaderTransitive rdf:resource=”http://a/Term/Url” />
```

Loosely coupled concepts are tagged using the `skos:related` tag thus:

```
<skos:related rdf:resource=”http://a/RelatedTerm/Url” />
```

Finally, synonyms (which were identified using the `skos:exactMatch` tag in version 1 of the NVS) are now specified using the Web Onotology Language’s `sameAs` tag, thus:

```
<owlxml:sameAs rdf:resource=”http://a/synonym/Url” />
```

It should be noted that `skos:exactMatch` has not been deprecated from the latest version of the SKOS specification, but has been specified only within the scope of concept schemes. As the concepts in NVS2.0 are registered to concept collections, `skos:exactMatch` cannot be used to signify synonymous relationships between concepts in NVS2.0.

SOAP Complex Data Types

The service returns results as XML documents. The major data types of which are discussed below.

ConceptCollection

XML conforming to the following XML Schema fragment is returned for each matching concept collection.

```
<xsd:complexType name="ConceptCollection">
  <xsd:sequence>
    <xsd:element name="error" type="xsd:string" minOccurs="0"/>
    <xsd:element name="collectionURI" type="xsd:string" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="collectionTitle" type="xsd:string"/>
    <xsd:element name="collectionAltTitle" type="xsd:string" minOccurs="0"/>
    <xsd:element name="collectionDescription" type="xsd:string" minOccurs="0"/>
    <xsd:element name="collectionCreator" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="collectionComment" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="collectionPublisher" type="xsd:string" nillable="false"/>
    <xsd:element name="collectionVersion" type="xsd:int" nillable="false"/>
    <xsd:element name="modified" type="xsd:dateTime" nillable="false"/>
    <xsd:element name="related" type="voctype:related" nillable="true" minOccurs="0"
       maxOccurs="unbounded"/>
    <xsd:element name="broadMatch" type="voctype:broadMatch" nillable="true" minOccurs="0"
       maxOccurs="unbounded"/>
    <xsd:element name="sameAs" type="voctype:sameAs" nillable="true" minOccurs="0"
       maxOccurs="unbounded"/>
    <xsd:element name="narrowMatch" type="voctype:narrowMatch" nillable="true" minOccurs="0"
       maxOccurs="unbounded"/>
    <xsd:element name="members" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="uri" type="xsd:string"/>
</xsd:complexType>
```

Where

- error – A response reporting an error in the SOAP call
- collectionURI – URI, in this case the URL, for a particular concept collection
- collectionTitle – The preferred human-readable label for a concept collection
- collectionAltTitle – An alternative human-readable label for a concept collection
- collectionDescription – A description of the commonality which links the members of the collection
- collectionCreator – The person or body responsible for the content governance of a concept collection
- collectionComment – A description of the body described by collectionCreator
- collectionPublisher – The body responsible for publishing the concept collection

- collectionVersion – The version of the concept collection accessed by the call to the SOAP method
- modified – The date on which the concept collection version was modified
- related – Links to collections containing loosely related concepts
- broadMatch – links to collections containing concepts at a broader semantic granularity
- sameAs – links to collections containing synonymous concepts
- narrowMatch - links to collections containing concepts at a narrower semantic granularity
- members – Zero, one or many concepts reported as the Concept complex data type representing the concepts registered to the collection

ConceptScheme

XML conforming to the following XML Schema fragment is returned for each matching concept collection.

```
<xsd:complexType name="ConceptScheme">
  <xsd:sequence>
    <xsd:element name="error" type="xsd:string" minOccurs="0"/>
    <xsd:element name="schemeTitle" type="xsd:string" nillable="false"/>
    <xsd:element name="schemeAltTitle" type="xsd:string" nillable="false"/>
    <xsd:element name="schemeURI" type="xsd:string" nillable="false"/>
    <xsd:element name="schemeDescription" type="xsd:string" nillable="false"/>
    <xsd:element name="schemeCreator" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="schemePublisher" type="xsd:string" nillable="false"/>
    <xsd:element name="schemeVersion" type="xsd:int" nillable="false"/>
    <xsd:element name="modified" type="xsd:dateTime" nillable="false"/>
    <xsd:element name="topConcept" type="xsd:string" nillable="false" maxOccurs="unbounded"/>
    <xsd:element name="members" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"
        nillable="false"/>
  </xsd:sequence>
  <xsd:attribute name="uri" type="xsd:string"/>
</xsd:complexType>
```

Where

- error - A response reporting an error in the SOAP call
- schemeTitle – The preferred human-readable label for the concept scheme
- schemeAltTitle – An alternative human-readable label for the concept scheme
- schemeURI – The URI, in this case actually the URL, to the concept scheme
- schemeDescription – A plain text description of the content of the concept scheme
- schemeCreator – The body responsible for creating the concept scheme
- schemePublisher – The body responsible for publishing the concept scheme
- schemeVersion – The version of the concept scheme returned
- modified – The date and time of publication of the version of the concept scheme returned
- topConcept – URL of a concept which is an entry point into the concept scheme
- members – Zero, one or many concepts reported as the Concept complex data type representing the concepts in the concept scheme

Concept

XML conforming to the following XML Schema fragment is returned for each matching concept.

```
<xsd:complexType name="Concept">
  <xsd:sequence>
    <xsd:element name="error" type="xsd:string" minOccurs="0"/>
    <xsd:element name="conceptID" type="xsd:string" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="prefLabel" type="voctype:prefLabel" minOccurs="0" maxOccurs="unbounded"
        nillable="false"/>
    <xsd:element name="altLabel" type="voctype:altLabel" minOccurs="0" maxOccurs="unbounded"
        nillable="true"/>
    <xsd:element name="definition" type="voctype:definition" minOccurs="0" maxOccurs="unbounded"
        nillable="true"/>
    <xsd:element name="modified" type="xsd:dateTime" nillable="false" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="memberScheme" type="voctype:memberList" nillable="false" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="version" type="xsd:int" nillable="true"/>
    <xsd:element name="isTopConcept" type="voctype:isTopConcept" nillable="false" minOccurs="0"
        maxOccurs="1"/>
    <xsd:element name="isDeprecated" type="xsd:boolean" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="memberCollection" type="voctype:memberList" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="identifier" type="xsd:string" nillable="true" minOccurs="1" maxOccurs="1"/>
    <xsd:element name="related" type="voctype:related" nillable="true" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="broadMatch" type="voctype:broadMatch" nillable="true" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="sameAs" type="voctype:sameAs" nillable="true" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="narrowMatch" type="voctype:narrowMatch" nillable="true" minOccurs="0"
        maxOccurs="unbounded"/>
    <xsd:element name="transitiveNarrowerMatch" type="voctype:transitiveNarrowerMatch" nillable="true"
        minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="transitiveBroaderMatch" type="voctype:transitiveBroaderMatch" nillable="true"
        minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="uri" type="xsd:string"/>
</xsd:complexType>
```

Where

- error - A response reporting an error in the SOAP call
- conceptID – Opaque label uniquely identifying a concept within a collection
- prefLabel – The preferred human-readable label of the concept
- altLabel – An alternative human-readable label for the concept, often an abbreviation
- definition – Supporting information which describes the concept in greater detail
- modified – The time and date on which the concept was last updated
- version – The current version of the concept
- isTopConcept – True if the concept is a top concept in the response to a request for a concept scheme
- isDeprecated – True if the concept is deprecated
- memberCollection – The concept collection to which the concept is registered
- identifier – An external identifier for the concept (the concept's URN)
- related – Links to loosely related concepts
- broadMatch – Links to concepts at a broader semantic granularity

- sameAs – Links to synonymous concepts
- narrowMatch - Links to concepts at a narrower semantic granularity
- transitiveNarrowMatch - Links to concepts at a narrower semantic granularity within the same response to a request for a concept scheme
- transitiveBroadMatch - Links to concepts at a broader semantic granularity within the same response to a request for a concept scheme

RelatedConcepts

XML conforming to the following XML Schema fragment is returned for all matching related concepts.

```
<xsd:complexType name="RelatedConcepts">
  <xsd:sequence>
    <xsd:element name="error" type="xsd:string" minOccurs="0"/>
    <xsd:element name="concept" type="voctype:Concept"/>
    <xsd:element name="narrowMatches" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"
        nillable="true"/>
    <xsd:element name="broadMatches" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"
        nillable="true"/>
    <xsd:element name="related" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"
        nillable="true"/>
    <xsd:element name="sameAs" type="voctype:collectionMembers" minOccurs="0" maxOccurs="1"
        nillable="true"/>
  </xsd:sequence>
</xsd:complexType>
```

Where

- error - A response reporting an error in the SOAP call
- concept – An object of the Concept complex data type which represents the subject of the query
- narrowMatches – A number of objects of the Concept complex data type at a narrower semantic granularity than the subject concept
- broadMatches - A number of objects of the Concept complex data type at a broader semantic granularity than the subject concept
- related - A number of objects of the Concept complex data type loosely related to the subject concept
- sameAs - A number of objects of the Concept complex data type synonymous with the subject concept

SearchResults

```
<xsd:complexType name="SearchResponse">
  <xsd:sequence>
    <xsd:element name="error" type="xsd:string" minOccurs="0"/>
    <xsd:element name="query" type="xsd:string" minOccurs="0"/>
    <xsd:element name="noOfResults" type="xsd:int" nillable="false"/>
    <xsd:element name="noOfMemberCollections" type="xsd:int" nillable="false"/>
    <xsd:element name="results" type="voctype:collectionMembers" minOccurs="0"
        maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

Where

- error - A response reporting an error in the SOAP call
- query – A verbatim duplication of the query sent in the SOAP method call
- noOfResults – The number of concepts matching the query
- noOfMemberCollections – Returns zero
- results - Zero, one or many concepts reported as the Concept complex data type representing the concepts matching the query sent in the SOAP method call