Project No. 249024

NETMAR

Open service network for marine environmental data

Instrument: *Please tick*	CA	✓**STREP**	IP	NOE

ICT - Information and Communication Technologies Theme

D4.4.2 – Implementation of the Semantic Framework – Version 2

Reference: D4.4.2_Implementation_of_the_Semantic_Framework_V2_r2_20120824

Due date of deliverable (as in Annex 1): M0 + 30
Actual submission date: 24 August 2012

Start date of project: 1 February 2010 Duration: 3 years

Coastal and Marine Resources Centre (CMRC), University College Cork, National University of Ireland

Revision 2

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

European Commission
Information Society and Media

	**NETMAR** Open service network for marine environmental data Project Reference: 249024 Contract Type: Collaborative Project Start/End Date: 01/03/2010 - 31/01/2013 Duration: 36 months
	Coordinator: Prof. Stein Sandven Nansen Environmental and Remote Sensing Center Thormøhlensgate 47, Bergen, Norway Tel.: +47-55205800 Fax. +47 55 20 58 01 E-mail: stein.sandven@nersc.no

Acknowledgements

The work described in this report has been partially funded by the European Commission under the Seventh Framework Programme, Theme ICT 2009.6.4 ICT for environmental services and climate change adaptation.

Consortium

The NETMAR Consortium is comprised of:

- Nansen Environmental and Remote Sensing Center (NERSC), Norway (coordinator).
  Project Coordinator: Prof. Stein Sandven (stein.sandven@nersc.no)
  Deputy Coordinator: Dr. Torill Hamre (torill.hamre@nersc.no)
  Quality Control Manager: Mr. Lasse H. Pettersson (lasse.pettersson@nersc.no)
- British Oceanographic Data Centre (BODC), National Environment Research Council, United Kingdom
  Contact: Dr. Roy Lowry (rkl@bodc.ac.uk)
- Centre de documentation de recherche et d'expérimentations sur les pollutions accidentelles des eaux (Cedre), France.
  Contact: Mr. François Parthiot (Francois.Parthiot@cedre.fr)
- Coastal and Marine Resources Centre (CMRC), University College Cork, National University of Ireland, Cork, Ireland.
  Contact: Mr. Declan Dunne (d.dunne@ucc.ie)
- Plymouth Marine Laboratory (PML), United Kingdom.
  Contact: Mr. Steve Groom (sbg@pml.ac.uk)
- Institut français de recherche pour l'exploitation de la mer (Ifremer), France.
  Contact: Mr. Mickael Treguer (mickael.treguer@ifremer.fr)
- Norwegian Meteorological Institute (METNO), Norway.
  Contact: Mr. Øystein Torget (oysteint@met.no)

Author(s)

- Yassine Lassoued, UCCNUIC (CMRC), y.lassoued@ucc.ie

Document approval

- Document status: Revision 2
- WP leader approval: 2012-08-24
- Quality Manager approval: 2012-08-24
- Coordinator approval: 2012-08-24

Revision History

Issue	Date	Change records	Author(s)
Draft	2011-06-24	First draft of the report.	Y. Lassoued
Draft	2011-07-11	Revised draft of the report incorporating comments from quality control.	Y. Lassoued
1	2011-07-18	First version approved by the coordinator.	T. Hamre
Draft	2012-07-09	First draft of the revised report.	Y. Lassoued
Draft	2012-08-24	Revised draft of the revised report incorporating comments from quality control.	Y. Lassoued
2	2012-08-24	Second version approved by the coordinator.	T. Hamre

Executive Summary

NETMAR aims to develop a pilot European Marine Information System (EUMIS) for searching, downloading and integrating satellite, in situ and model data from ocean and coastal areas. EUMIS will use a semantic framework coupled with ontologies for identifying and accessing distributed data, such as near-real time, forecast and historical data.

The NETMAR semantic framework aims to provide the tools (APIs and services) required to handle semantics in various EUMIS components, and external applications. It is responsible for reasoning over the NETMAR semantic knowledge and for providing high-level and easy-to-use semantic operations through a standardised interface, called the semantic web service (SWS). The semantic framework is used by several EUMIS components for several purposes:

1. To facilitate the browsing of the NETMAR vocabularies by providing the EUMIS ontology browser with easy-to-use and high-level semantic methods (e.g., get related concepts, get concept hierarchy, etc.) required to browse the ontologies.
2. Facilitate the semantic interoperability of catalogue services through the use a catalogue mediator (CSWM). The CSWM uses the SF to translate terms from one given vocabulary/language into terms supported by the NETMAR catalogue services. For instance, you can search datasets using a keyword from one vocabulary or language and get results from a NETMAR catalogue service that uses a different vocabulary or language.
3. Improve the search results of the NETMAR discovery client, by exploiting the semantic relationships between terms (narrower, related, same as, etc.), and/or trying to interpret the meaning of a user free text keyword according to a given thesaurus. For instance, if you search for datasets matching the term “seabed”, you would be able to get those tagged with the keyword “seafloor” (synonym), or if you search for “CTD” (i.e., Conductivity, Temperature, Depth), you would be able to get “Sea Surface Salinity” datasets.
4. Improve service chaining by ensuring that the inputs and outputs of each component of a service chain are “semantically compatible.” In this way, correct connections between components can be enforced (e.g., do not send chlorophyll data to a component that only knows how to process sea surface temperature).

The semantic framework contains two main services, which are the semantic web service (SWS) and the catalogue service mediator, or CSW mediator (CSWM). The SWS version 2.0 was implemented in Java and using the Jena ontology framework. It is deployed on the CMRC server and can be accessed at http://netmar.ucc.ie/srv/SWS. The SWS uses Jena TDB as a backend for storing and managing the ontologies, which are automatically loaded from the NERC Vocabulary Server using and extract, transform and load (ETL) tool. Version 2.0 of the SWS has brought several improvements both internally (backend and code wise, e.g., code structure, Jena TDB triple store technology, Lucene index, etc.) and externally (from the user perspective, e.g. improved performance, support for multilingual ontologies, support for collections, etc.).

The CSWM 1.0, which is implemented in Java, allows access to the NETMAR distributed catalogue services, called catalogue nodes, e.g., NERSC, met.no, and IFREMER catalogue nodes. The CSWM is semantically enabled, i.e., it searches datasets and services in the catalogues by “meaning” rather than by mere keywords. To do so, it uses the SWS to interpret the user’s search keywords, and rewrites the user’s query to include more specific and related terms. For instance, if you search for datasets in the CSWM using the term “ice motion”, the CSWM will include “sea ice speed”, “sea ice displacement”, “sea ice velocity”, etc.

The CSWM does not harvest or index the metadata records of the catalogue nodes. Instead, it rewrites the user's query into queries supported by the catalogue nodes and executes them on the fly, then collects the answers from the different nodes and sends them back to the user.

The uncertainty propagation model that has been defined as part of the NETMAR semantic framework specification is currently being implemented in the NETMAR web processing services (WPS). The implementation of the uncertainty propagation model, which is based on UncertWeb, will help users understand how data uncertainty propagates through a service chain, which will give them an estimate of the quality (uncertainty) of the final outputs of a service chain. As this is still under development, the implementation of uncertainty propagation will be documented in a future version of this report.

Contents

1 INTRODUCTION 5

1.1 BACKGROUND 5
1.2 OBJECTIVE OF THIS REPORT 5
1.3 ORGANISATION OF THIS REPORT 6

2 SEMANTIC WEB SERVICE 2.0 7

2.1 IMPLEMENTATION 7
2.2 REQUEST PROTOCOLS 8
2.3 CHANGES SINCE VERSION 1.0 8
2.4 STATUS OF IMPLEMENTATION OF THE SWS 2.0 OPERATIONS 8
2.5 EXCEPTION MANAGEMENT 10
2.6 IMPLEMENTED SWS 2.0 OPERATIONS 10
2.6.1 *GetCapabilities Operation* *11*
2.6.2 *GetConceptSchemes Operation* *12*
2.6.3 *GetConceptScheme Operation* *12*
2.6.4 *GetConcepts Operation* *13*
2.6.5 *GetConcept Operation* *14*
2.6.6 *SearchConcept Operation* *15*
2.6.7 *GetRelatedConcepts Operation* *17*
2.6.8 *GetExplicitTopConcepts Operation* *18*
2.6.9 *GetImplicitTopConcepts Operation* *19*
2.6.10 *GetConceptHierarchy Operation* *19*
2.6.11 *InterpretKeyword Operation* *20*

3 CSW MEDIATOR 23

3.1 IMPLEMENTATION 23
3.2 CSWM CHAINING 24
3.2.1 *Metadata Record Identifiers* *24*
3.2.2 *Hop Count Control* *25*
3.3 REQUEST PROTOCOLS 25
3.4 EXCEPTION MANAGEMENT 25
3.5 CSWM OPERATIONS 26
3.5.1 *CSW 2.0.2 Operations* *26*
3.5.2 *CSWM 1.0 Operations* *29*

4 REFERENCES 33

5 ACRONYMS 34

1 Introduction

1.1 Background

NETMAR[1] aims to develop a pilot European Marine Information System (EUMIS) for searching, downloading and integrating satellite, in situ and model data from ocean and coastal areas. It will be a user-configurable system offering flexible service discovery, access and chaining facilities using OGC, OPeNDAP and W3C standards. It will use a semantic framework (SF) coupled with ontologies for identifying and accessing distributed data, such as near-real time, forecast and historical data. EUMIS will also enable further processing of such data to generate composite products and statistics suitable for decision-making in diverse marine application domains. Figure 1.1 illustrates how observations, derived parameters and predictions are retrieved from a distributed service network through standard protocols, and delivered through the EUMIS portal using ontologies and semantic frameworks to select suitable products and where new products can be generated dynamically using chained processing services.

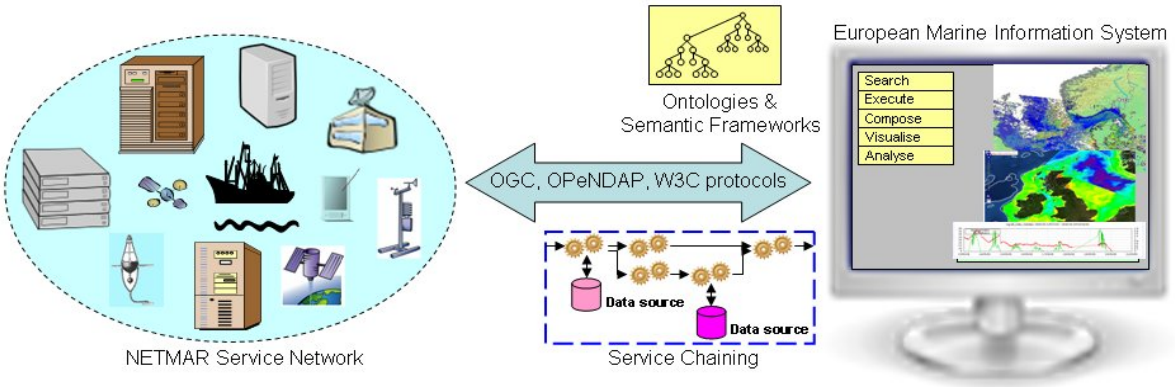

Figure 1.1. The NETMAR Service Network

1.2 Objective of this Report

This report describes the implementation of **version 2.0** of the NETMAR semantic framework (SF) for discovering, accessing and chaining marine environmental web services. It summarises the features supported by this version, and provides examples of how to use it. The NETMAR semantic framework and its features were specified and implemented according to the user requirements identified as part of work packages 1 (report D1.1, “Use case specification and schedule for testing”) and 2 (report D2.2.2, “System architecture”).

1.3 The NETMAR Semantic Framework

The NETMAR semantic framework is the system and application programming interfaces (APIs) responsible for handling (managing, reasoning and delivering) semantic knowledge in EUMIS. It is specified in the NETMAR report D4.3.2, entitled “NETMAR Semantic Framework Specification - Version 2” [D4.3.2].

The semantic framework is used in NETMAR to:

5. Facilitate the browsing of the NETMAR vocabularies by providing the EUMIS ontology browser with easy-to-use and high-level semantic methods (e.g., get related concepts, get concept hierarchy, etc.) required to browse the ontologies.

[1] http://www.netmar-project.eu/

6. Facilitate the semantic interoperability of catalogue services through the use a catalogue mediator (CSWM). The CSWM uses the SF to translate terms from one given vocabulary/language into terms supported by the NETMAR catalogue services. For instance, you can search datasets using a keyword from one vocabulary or language and get results from a NETMAR catalogue service that uses a different vocabulary or language.
7. Improve the search results of the NETMAR discovery client, by exploiting the semantic relationships between terms (narrower, related, same as, etc.), and/or trying to interpret the meaning of a user free text keyword according to a given thesaurus. For instance, if you search for datasets matching the term “seabed”, you would be able to get those tagged with the keyword “seafloor” (synonym), or if you search for “CTD” (i.e., Conductivity, Temperature, Depth), you would be able to get “Sea Surface Salinity” datasets.
8. Improve service chaining by ensuring that the inputs and outputs of each component of a service chain are “semantically compatible.” In this way, correct connections between components can be enforced (e.g., do not send chlorophyll data to a component that only knows how to process sea surface temperature).

Figure 1.2 shows the interaction between the EUMIS components and the NETMAR semantic framework as explained above.

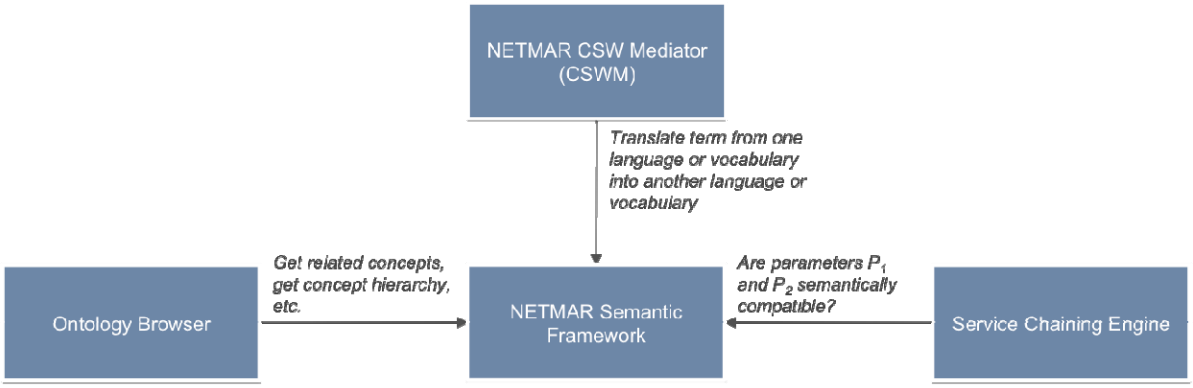

Figure 1.2. The NETMAR Semantic Framework Usage

1.4 Organisation of this Report

This report is organised as follows. Chapter 2 describes the status of implementation of the semantic web service. Next Chapter 3 describes the implementation of the catalogue service mediator, CSW mediator (CSWM).

2 Semantic Web Service 2.0

This chapter describes the implementation of the semantic web service, version 2.0, and provides examples of SWS requests and responses to illustrate how this version the service works and how it can be used.

2.1 Implementation

The semantic web service was implemented in Java, using the Jena ontology framework. It is deployed on the CMRC server and can be accessed at http://netmar.ucc.ie/srv/SWS. As shown in Figure 2.1, The SWS uses Jena TDB[2] as a backend for storing and managing the ontologies, which are automatically loaded from the NERC Vocabulary Server using and extract, transform and load (ETL) tool.

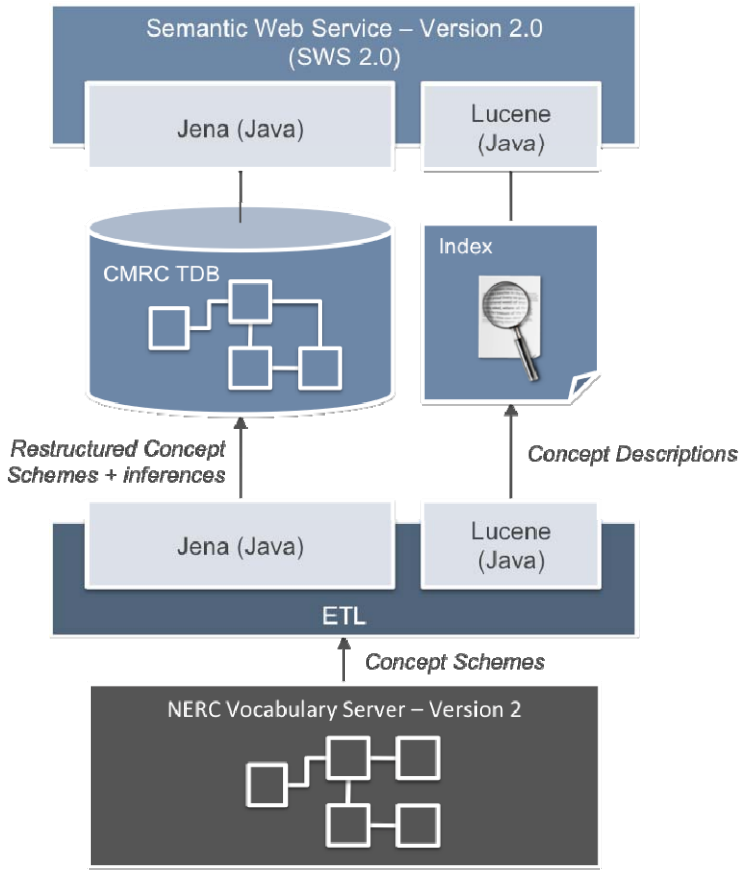

Figure 2.1. The NETMAR Semantic Framework Implementation Architecture

The ETL tool pulls the NEMTAR concept schemes from the NERC Vocabulary Server (NVS, version 2). Then, it computes inferences, and restructures the concept schemes according to the SWS requirements (c.f., report D4.3.2, "NETMAR Semantic Framework Specification - Version 2" [D4.3.2]). The ETL tool also indexes the concept descriptions (labels and definitions) for fast search.

[2] http://jena.apache.org/documentation/tdb/index.html

The SWS performs concept search using the concept indexes, whereas it extracts semantic knowledge from the CMRC TDB.

Both the SWS and the ETL tool use the Jena 2.7.0 framework to access and manage the TDB, and the Lucene[3] 3.5.0 API for accessing and managing the concept indexes.

2.2 Request Protocols

The SWS supports key-value pair (KVP) requests over HTTP. The XML interface specified in D4.3.2 is functionally equivalent to the KVP interface. Therefore, in view of the limited development time, we prioritised the KVP implementation over the XML one, as the former is easier to test using a standard web browser. The XML protocol is well specified in [D4.3.2] and can be implemented later if needed.

2.3 Changes since Version 1.0

The second version of the SWS has brought several improvements both internally (backend and code wise) and externally (from the user perspective). Some of these improvements have influenced by user requirements (e.g., multilingual search). Other improvements have been made for performance and scalability reasons (e.g., backend changes).

- From the user perspective, the implemented SWS 2.0 added the following features:
  - Support for the SWS 2.0 specification
  - Support for multilingual ontologies: users can now specify the response language, and search concepts using one language and request results in the same, or in a different, language (e.g., search for an English term using a French term, etc.).
  - Support for collections, e.g., get collections, get concepts belonging to a collection and/or concept scheme, etc.
  - Improved performance, that is 7 to 10 times faster with regard to concept hierarchy building and concept search compared to the previous version.
  - Support for a new convenience method (InterpretKeyword) for use by the CSW mediator. This method is intended to replace the series of interactions with the SWS (3 SWS calls) used in the previous version of the semantic framework.
- Internally, the following changes have been made in order to improve the system extensibility, maintenance, and performance:
  - The code structure and modularity have been improved since version 1.0.
  - The Jena TDB is now being used for storing and querying the NETMAR ontologies that are pulled from the NVS. Jena TDB replaces the older Jena PostgreSQL RDB used in SWS 1.0 and which is no more supported by the Jena community. This would allow easier software update and maintenance by using a community-supported backend.
  - SWS 2.0 uses a Lucene index for concept search. An index is created by the ETL tools for each supported language, and is used by the SWS to perform quick keyword-based concept search. The Lucene index improves the performance of concept search that was implemented using a manual ad-hoc operation in version 1.0.

2.4 Status of Implementation of the SWS 2.0 Operations

Table 2.1 shows the implementation status of the specified SWS 2.0 operations and the rational for the decision as to whether or not an operation was implemented. More details on these operations are to be found in subsections 2.6.1 to 2.6.13 as specified in Table 2.1.

[3] http://lucene.apache.org/core/

Table 2.1. Status and Rational for the Implementation of the SWS 2.0 Operations:

1. *The operation was not implemented because it is not used by the EUMIS components*
2. *The type of resources returned or handled by this operation is not used in NETMAR (e.g., collections)*
3. *For convenience, this operation is implemented elsewhere (e.g., NVS)*
4. *This is a new operation that has not been identified in the SWS 2.0 specification*

Section	Operation	Description	Implemented?
2.6.1	*GetCapabilities*	Retrieves service metadata, including supported operations, response formats, available concept schemes, and their supported languages.	Ongoing
2.6.2	*GetConceptSchemes*	Lists available concept schemes with their annotations (labels, definitions, etc.).	Yes
2.6.3	*GetConceptScheme*	Returns a concept scheme definition given its URI.	Yes
	SearchConceptScheme	Returns the definition(s) of one or more concept scheme(s) matching a specified free-text keyword.	No[1]
	GetConceptSchemeContent	Returns the content of a specified concept scheme identified by its URI, including its collections and concepts.	No[1]
	GetCollections	Lists available concept collections with their annotations. Collections may be filtered by one or more concept schemes.	Yes[2]
	GetCollection	Returns a collection definition identified by its URI.	Yes[2]
	SearchCollection	Returns the definition(s) of one or more collection(s) matching a specified free-text keyword.	No[1,2]
	GetCollectionContent	Returns the content of a given collection identified by its URI, including member collections and concepts.	No[1,2]
2.6.4	*GetConcepts*	Returns the definitions of the concepts belonging to a specified concept scheme and/or collection.	Yes
2.6.5	*GetConcept*	Returns a concept definition given its URI. The response includes the concept's annotations.	Yes
2.6.6	*SearchConcept*	A smart search operation that returns the concepts matching a given keyword.	Yes
2.6.7	*GetRelatedConcepts*	Returns the concepts related to one or many given concept(s) using one or many given SKOS relationship(s) both from direct assertions and by entailment.	Yes
2.6.8	*GetExplicitTopConcepts*	Returns the concepts that have explicitly been asserted as top concepts of a specified concept scheme.	Yes
2.6.9	*GetImplicitTopConcepts*	Returns the concepts of one concept scheme that have no broader concepts within the latter.	Yes
2.6.10	*GetConceptHierarchy*	Returns the hierarchy of the concepts within a given concept scheme and/or collection.	Yes
	CheckRelation	Checks whether two specified concepts are related via a specified SKOS relationship.	No[3]
2.6.11	*InterpretConcept*		Yes[4]

2.5 Exception Management

Table 2.2, below, lists the various types of exceptions that might be raised by the semantic web service.

Table 2.2. SWS Exception Codes

Exception code	Description
InternalError	Internal error to the SWS
InvalidParameterValue	The parameter value is not valid
InvalidRequest	The request message is either invalid or is not well-formed
MissingParameter	A parameter is missing
NoApplicableCode	There is no applicable code to this exception
NotImplemented	The (abstract) operation has not been implemented
NotSupported	A service option or capability is not supported
NullResourceValue	A requested resource has a null value
NullValue	Null value exception: a required parameter or variable is null
ResourceNotFound	The requested resource does not exist or could not be found
ResourceTypeMismatch	The requested resource does not have the required type
UnknownError	The error type is unknown

Particular attention was paid to providing human-readable messages as well as machine processable codes and locators in exceptions for identifying the types of exceptions and their causes.

The XML fragment below illustrates an exception report generated by the SWS due to an unknown concept specified in a *GetConcept* operation.

```
<sws:ExceptionReport xmlns:sws="http://cmrc.ucc.ie/sws/2.0"
 xmlns:xml=http://www.w3.org/XML/1998/namespace xml:lang="en" version="1.0">
  <sws:Exception exceptionCode="ResourceNotFound"
   locator="http://netmar.ucc.ie/ont/example#BadConceptURI">
    <sws:ExceptionText>
      Resource "http://netmar.ucc.ie/ont/example#BadConceptURI": No such
      concept or resource.
    </sws:ExceptionText>
  </sws:Exception>
</sws:ExceptionReport>
```

2.6 Implemented SWS 2.0 Operations

All the SWS requests below, except for GetCapabilities, shall specify the service type, the accepted service versions and the request type. In addition, they may specify the level of resource information to be returned by the SWS and the response language. Table 2.3 defines the list of the common operation parameters as implemented in version 2.0 of the SWS.

Table 2.3. Common SWS Request Parameters

Parameter	Cardinality	Definition	Example
service	1	Type of service requested ***Possible values:*** "SWS"	service=SWS
version	1	Service version accepted by the client ***Possible values:*** "2.0"	version=2.0
request	1	Operation requested by the client ***Possible values:*** see Table 4.1	request=GetConcept
acceptFormat	0..1	Response format expected by the client ***Specified Possible values:*** "text/xml",	AcceptFormat=text/xml

		"application/json" ***Implemented values:*** "text/xml" ***Default value:*** "text/xml"	
responseLanguage	0..1	Response language ***Type:*** ISO 639-1 two-letter language code ***Default value:*** empty string, meaning that appropriate annotations in all languages should ne returned ***Note:*** This parameter was initially named "language" in the SWS 2.0 specification. It has been changed to "responseLanguage" for disambiguation, in order to avoid confusion with the *keywordLanguage* parameter (c.f., *SearchConcept* operation, subsection 2.6.8).	language=en
elementSet	0..1	Level of resource details returned by the SWS ***Possible values and meanings:*** • "abstract": *only URI shall be returned* • "brief": *abstract information and preferred labels shall be returned* • "summary": *brief information, associated concept schemes (if applicable), and definitions shall be returned* • "full": *summary information and alternate and hidden labels shall be returned* • "extended": *full information and any external mappings shall be returned* ***Default value:*** "abstract"	elementSet=full

2.6.1 *GetCapabilities Operation*

The *GetCapabilities* operation only requires the *service* parameter, defined in table 2.3, and the *section* parameter that specifies the capabilities document sections requested by the client (c.f., Table 2.4).

Table 2.4. *GetCapabilities* Request Parameters

Parameter	Cardinality	Definition	Example
section	0..4	Capabilities document section requested ***Possible values:*** • "ServerIdentification" • "ServiceProvider" • "OperationsMetadata" • "SupportedConceptSchemeList" ***Default value:*** empty string, meaning that all sections should ne returned	section=ServiceProvider &section=OperationsMetadata

An example of a *GetCapabilities* request is given below.

```
http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&request=GetCapabilities
&section=ServerIdentification
&section= OperationsMetadata
```

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&request=GetCapabilities&section=ServerIdentification&section=OperationsMetadata

2.6.2 *GetConceptSchemes Operation*

The GetConceptSchemes operation is used to retrieve information about the available concept schemes. The GetConceptSchemes operation does not require more parameters than the ones defined in Table 2.3.

An example of a GetConceptSchemes request is given below.

http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetConceptSchemes
&elementSet=full
&responseLanguage=en

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetConceptSchemes&elementSet=full&responseLanguage=en

The GetConceptSchemes response is an RDF document containing the list of concepts schemes delivered by the SWS.

The XML below shows a fragment of the GetConceptSchemes response associated with the request example above. For sake of readability, namespace declarations have been omitted.

```
<rdf:RDF>
  <skos:ConceptScheme
   rdf:about="http://vocab.nerc.ac.uk/scheme/NETOC_INSTRUMENT/current/">
    <skos:prefLabel>
      NETMAR Oceanography Thesaurus Instrument Facet
    </skos:prefLabel>
    <skos:altLabel>NETMAR Oceans Instrument</skos:altLabel>
  </skos:ConceptScheme>
  <skos:ConceptScheme
   rdf:about="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/">
    <skos:prefLabel>
      NETMAR Oceanography Thesaurus Parameter Facet
    </skos:prefLabel>
    <skos:altLabel>NETMAR Oceans Parameter</skos:altLabel>
  </skos:ConceptScheme>
  <skos:ConceptScheme
   rdf:about="http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/">
    <skos:prefLabel>
      NETMAR Oceanography Thesaurus Platform Class Facet
    </skos:prefLabel>
    <skos:altLabel>NETMAR Oceans Platform Class</skos:altLabel>
  </skos:ConceptScheme>
  <!--Other concept schemes-->
</rdf:RDF>
```

2.6.3 *GetConceptScheme Operation*

The GetConceptScheme operation allows a client to retrieve information about a specified concept scheme identified by its URI.

In addition to the common SWS request parameters (defined in Table 2.3), the *GetConceptScheme* request uses the *conceptScheme* parameter that specifies the URI of the concept scheme requested by the client (c.f., Table 2.5).

Table 2.5 GetConceptScheme Request Parameters

Parameter	Cardinality	Definition	Example
conceptScheme	1	URI of the concept scheme requested ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/

An example of a GetConceptScheme request is provided below.

```
http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=1.0
&request=GetConceptScheme
&elementSet=full
&responseLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/
```

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetConceptScheme&elementSet=full&responseLanguage=en&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PLATCLASS%2Fcurrent%2F

The GetConceptScheme response is an RDF document describing the requested concept scheme.

The XML below shows a fragment of the GetConceptScheme response associated with the request example above.

```
<rdf:RDF>
  <skos:ConceptScheme
   rdf:about="http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/">
    <skos:prefLabel>
      NETMAR Oceanography Thesaurus Platform Class Facet
    </skos:prefLabel>
    <skos:altLabel>NETMAR Oceans Platform Class</skos:altLabel>
  </skos:ConceptScheme>
</rdf:RDF>
```

2.6.4 *GetConcepts Operation*

The *GetConcepts* operation retrieves all the concepts of one concept scheme with possible filtering by collections.

In addition to the common SWS semantic request parameters (defined in Table 2.3), the *GetConcepts* request uses the *conceptScheme* parameter that specifies the URI of the concept scheme of interest, and the collection parameter that specifies the URI of the collection to which the returned concepts must belong (c.f., Table 2.6).

Table 2.6. *GetConcepts* Request Parameters

Parameter	Cardinality	Definition	Example
conceptScheme	1	URI of the concept scheme requested ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/
collection	*	URI of a collection requested ***Type:*** URI	<COLLECTIONS ARE NOT USED IN NETMAR>

An example of a *GetConcepts* request is provided below.

http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetConcepts
&elementSet=full
&responseLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetConcepts&elementSet=full&responseLanguage=en&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PLATCLASS%2Fcurrent%2F

The *GetConcepts* response is an RDF document listing the concepts belonging to the requested concept scheme. The XML below shows a fragment of the *GetConcepts* response associated with the request example above.

```
<rdf:RDF>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/31">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/"/>
    <skos:prefLabel xml:lang="en">research vessel</skos:prefLabel>
    <skos:definition xml:lang="en">
      A platform of any size operating on the surface of the water column...
    </skos:definition>
  </skos:Concept>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/64">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/"/>
    <skos:prefLabel xml:lang="en">
      geostationary orbiting satellite</skos:prefLabel>
    <skos:definition xml:lang="en">
      A vehicle operating beyond the Earth's atmosphere without...
    </skos:definition>
  </skos:Concept>
  <!--Other concepts-->
</rdf:RDF>
```

2.6.5 *GetConcept Operation*

The *GetConcept* operation allows a client to retrieve information about a specified concept identified by its URI.

In addition to the common SWS request parameters (defined in Table 2.3), *GetConcept* requires the *concept* parameter that specifies the URI of the concept requested by the client (c.f., Table 2.7).

Table 2.7 *GetConcept* Request Parameters

Parameter	Cardinality	Definition	Example
concept	1	URI of the concept requested ***Type:*** URI	concept=concept=http://vocab.nerc.ac.uk/collection/L06/current/64

An example of a *GetConcept* request is provided below.

http://netmar.ucc.ie/thesaurer/Thesurer?
service=SWS
&version=2.0
&request=GetConcept
&elementSet=full
&responseLanguage=en
&concept=http://vocab.nerc.ac.uk/collection/L06/current/64

Encoded URL:
http://netmar.ucc.ie/thesaurer?Thesaurer=SWS&version=2.0&request=GetConcept&elementSet=full&responseLanguage=en&concept=http%3A%2F%2Fvocab.nerc.ac.uk%2Fcollection%2FL06%2Fcurrent%2F64

The *GetConcept* response is an RDF document describing the requested concept. The XML below shows a fragment of the *GetConcept* response associated with the request example above.

```
<rdf:RDF>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/64">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/"/>
    <skos:prefLabel xml:lang="en">
      geostationary orbiting satellite
    </skos:prefLabel>
    <skos:definition xml:lang="en">
      A vehicle operating beyond the Earth's atmosphere without human occupants
      that orbits the Earth at the same rate as the Earth's rotation keeping it
      over a fixed location on the Earth's surface.
    </skos:definition>
  </skos:Concept>
</rdf:RDF>
```

2.6.6 *SearchConcept Operation*

The *SearchConcept* operation allows a client to retrieve information about one or more concept(s) matching a free-text keyword.

In addition to the common SWS request parameters (defined in Table 2.3), the *SearchConcept* request uses the parameters defined in Table 2.8.

Table 2.8 *SearchConcept* Request Parameters

Parameter	Cardinality	Definition	Example
keyword	1	Search keyword ***Type:*** free text	keyword=ice coverage
keywordLanguage	0..1	keyword language ***Type:*** ISO 639-1 two-letter language code	keywordLanguage=en
conceptScheme	0..1	URI of a concept scheme of interest ***Type:*** URI	conceptScheme= http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

collection	*	URI of a collection of interest ***Type:*** URI	<COLLECTIONS ARE NOT USED IN NETMAR>

An example of a *SearchConcept* request is provided below.

http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=SearchConcept
&elementSet=full
&responseLanguage=en
&keyword=ice coverage
&keywordLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=SearchConcept&elementSet=full&responseLanguage=en&keyword=ice%20coverage&keywordLanguage=en&conceptScheme=http%3A%2F%2Fvocab%2Enerc%2Eac%2Euk%2Fscheme%2FNETOC%5FPARAM%2Fcurrent%2F

The *SearchConcept* response is an RDF document listing the concepts matching the specified keyword and belonging to the specified concept scheme. Concepts are ranked according to their relevance to the user's keyword. The XML below shows a fragment of the *SearchConcept* response associated with the request example above.

```
<rdf:RDF>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P25/current/ICECOV">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/"/>
    <skos:prefLabel xml:lang="en">Ice coverage</skos:prefLabel>
    <skos:definition xml:lang="en">
      The extent to which the Earth's surface is covered by frozen water
    </skos:definition>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/SICECMOD">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/"/>
    <skos:prefLabel xml:lang="en">
      Coverage (by area) of ice on the water body by model prediction
    </skos:prefLabel>
    <skos:altLabel xml:lang="en">IceCov_Model</skos:altLabel>
    <skos:definition xml:lang="en">Unavailable</skos:definition>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/SICECSAT">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/"/>
    <skos:prefLabel xml:lang="en">
      Coverage (by area) of ice on the water body by image analysis
    </skos:prefLabel>
    <skos:altLabel xml:lang="en">IceCov_Sat</skos:altLabel>
    <skos:definition xml:lang="en">Unavailable</skos:definition>
  </skos:Concept>
  <!--Other concepts-->
</rdf:RDF>
```


2.6.7 *GetRelatedConcepts Operation*

The *GetRelatedConcepts* operation retrieves all the concepts related to one or more specified concepts, identified by their URIs, by one or more SKOS relationships. Concepts may be filtered by concept scheme and/or collections.

In addition to the common SWS request parameters (defined in Table 2.3), the *GetRelatedConcepts* request uses the parameters defined in Table 2.9.

Table 2.9. *GetRelatedConcepts* Request Parameters

Parameter	Cardinality	Definition	Example
concept	1..*	URI of the concept of interest	concept=http://vocab.nerc.ac.uk/collection/L06/current/31
relation	*	URI or local name of a SKOS relationship	relationship=http://www.w3.org/2004/02/skos/core#related
conceptScheme	*	URI of a concept scheme ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

An example of a *GetRelatedConcepts* request is provided below.

```
http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetRelatedConcepts
&elementSet=full
&responseLanguage=en
&concept=http://vocab.nerc.ac.uk/collection/L06/current/31
&relation=http://www.w3.org/2004/02/skos/core#related
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/
```

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetRelatedConcepts&elementSet=summary&responseLanguage=en&concept=http%3A%2F%2Fvocab.nerc.ac.uk%2Fcollection%2FL06%2Fcurrent%2F31&relation=http%3A%2F%2Fwww%2Ew3%2Eorg%2F2004%2F02%2Fskos%2Fcore%23related&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PARAM%2Fcurrent%2F

The *GetRelatedConcepts* response is an RDF document listing the concepts related to the requested concept using the specified relationships and belonging to the specified concept schemes. The XML below shows a fragment of the *GetRelatedConcepts* response associated with the request example above.


```
<rdf:RDF>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/P25/current/CHLA">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/"/>
    <skos:prefLabel xml:lang="en">Chlorophyll-a concentration</skos:prefLabel>
    <skos:definition xml:lang="en">
      The amount (mass or quantity of matter) of the pigment chlorophyll-a per
      amount (volume or mass) of a body of water
    </skos:definition>
  </skos:Concept>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/P25/current/NO3">
    <skos:inScheme
     rdf:resource="http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/"/>
    <skos:prefLabel xml:lang="en">Nitrate concentration</skos:prefLabel>
    <skos:definition xml:lang="en">
      The amount (mass or quantity of matter) of nitrate per amount (volume or
      mass) of a body of water
    </skos:definition>
  </skos:Concept>
  <!--Other related concepts-->
</rdf:RDF>
```

2.6.8 *GetExplicitTopConcepts Operation*

The *GetExplicitTopConcepts* operation retrieves all the concepts asserted as being the top concepts of a specified concept scheme identified by its URI.

In addition to the common SWS request parameters (defined in Table 2.3), the *GetExplicitTopConcepts* request uses the *conceptScheme* parameter that specifies the URI of the concept scheme requested by the client (c.f., Table 2.10).

Table 2.10. *GetExplicitTopConcepts* Request Parameters

Parameter	Cardinality	Definition	Example
conceptScheme	1	URI of the concept scheme requested ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

An example of a *GetExplicitTopConcepts* request is provided below.

http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetExplicitTopConcepts
&elementSet=full
&responseLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetExplicitTopConcepts&elementSet=full&responseLanguage=en&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PARAM%2Fcurrent%2F

The *GetExplicitTopConcepts* response is an RDF document listing the top concepts of the requested concept scheme. The response is similar to that of a *GetConcepts* response (see example in section 2.6.4).

2.6.9 *GetImplicitTopConcepts Operation*

The *GetImplicitTopConcepts* operation retrieves all the concepts belonging to a concept scheme and having no broader concepts within that same concept scheme. This includes the explicit top concepts.

Like the *GetExplicitTopConcepts* operation, the *GetImplicitTopConcepts* request uses the *conceptScheme* parameter that specifies the URI of the concept scheme requested by the client (c.f., Table 2.11).

Table 2.11. *GetImplicitTopConcepts* Request Parameters

Parameter	Cardinality	Definition	Example
conceptScheme	1	URI of the concept scheme requested ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

An example of a *GetImplicitTopConcepts* request is provided below.

```
http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetImplicitTopConcepts
&elementSet=full
&responseLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/
```

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetImplicitTopConcepts&elementSet=full&responseLanguage=en&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PARAM%2Fcurrent%2F

The *GetImplicitTopConcepts* response is an RDF document listing and describing the implicit top concepts of the specified concept scheme. The response is similar to that of a *GetConcepts* response (see example in section 2.6.4).

2.6.10 *GetConceptHierarchy Operation*

The *GetConceptHierarchy* operation retrieves all the concepts of a specified concept scheme, identified by its URI, organised in a hierarchy (nested structure) according to the SKOS *broaderTransitive* and *narrowerTransitive* relationships. This operation is suitable for small concept schemes (hundreds of concepts) and is useful for ontology browsers.

In addition to the common SWS request parameters (defined in Table 2.3), the *GetConceptHierarchy* request uses the *conceptScheme* parameter that specifies the URI of a concept scheme requested by the client (c.f., Table 2.12).

Table 2.12. *GetConceptHierarchy* Request Parameters

Parameter	Cardinality	Definition	Example
conceptScheme	1	URI of a requested concept scheme ***Type:*** URI	conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/

An example of a *GetConceptHierarchy* request is provided below.

http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=GetConceptHierarchy
&elementSet=brief
&responseLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PLATCLASS/current/

Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=GetConceptHierarchy&elementSet=brief&responseLanguage=en&conceptScheme=http%3A%2F%2Fvocab.nerc.ac.uk%2Fscheme%2FNETOC_PLATCLASS%2Fcurrent%2F

The *GetConceptHierarchy* response is an RDF document containing the hierarchy of the concepts belonging to the requested concept scheme. The XML below shows a fragment of the *GetConceptHierarchy* response associated with the request example above.

```
<rdf:RDF>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/L19/current/SDNKG05">
    <skos:prefLabel xml:lang="en">platform_class</skos:prefLabel>
    <skos:narrower>
      <skos:Concept
       rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/64">
        <skos:prefLabel xml:lang="en">
          geostationary orbiting satellite</skos:prefLabel>
      </skos:Concept>
    </skos:narrower>
    <skos:narrower>
      <skos:Concept
       rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/65">
        <skos:prefLabel xml:lang="en">orbiting satellite</skos:prefLabel>
      </skos:Concept>
    </skos:narrower>
    <skos:narrower>
      <skos:Concept
       rdf:about="http://vocab.nerc.ac.uk/collection/L06/current/31">
        <skos:prefLabel xml:lang="en">research vessel</skos:prefLabel>
      </skos:Concept>
    </skos:narrower>
  </skos:Concept>
</rdf:RDF>
```

2.6.11 *InterpretKeyword Operation*

The *InterpretKeyword* operation allows a client to retrieve the concept(s) related to a free-text keyword. This operation was added in order to reduce the number of interactions between the CSW mediator and the SWS.

In addition to the common SWS request parameters (defined in Table 2.3), the *InterpretKeyword* request uses the parameters defined in Table 2.13.

Table 2.13. *InterpretKeyword* Request Parameters

Parameter	Cardinality	Definition	Example
keyword	1	Search keyword ***Type:*** free text	keyword=ice coverage
keywordLanguage	0..1	keyword language ***Type:*** ISO 639-1 two-letter language code	keywordLanguage=en

conceptScheme	0..1	URI of a concept scheme of interest ***Type:*** URI	conceptScheme= http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/

An example of an *InterpretKeywod* request is provided below.

```
http://netmar.ucc.ie/thesaurer/Thesaurer?
service=SWS
&version=2.0
&request=InterpretKeyword
&elementSet=brief
&responseLanguage=en
&keyword=ice coverage
&keywordLanguage=en
&conceptScheme=http://vocab.nerc.ac.uk/scheme/NETOC_PARAM/current/
```

```
Encoded URL:
http://netmar.ucc.ie/thesaurer/Thesaurer?service=SWS&version=2.0&request=InterpretKeyword&elementSet=brief&responseLanguage=en&keyword=ice%20coverage&keywordLanguage=en&conceptScheme=http%3A%2F%2Fvocab%2Enerc%2Eac%2Euk%2Fscheme%2FNETOC%5FPARAM%2Fcurrent%2F
```

The *InterpretKeyword* response is an RDF document listing the concepts related to the specified keyword and belonging to the specified concept scheme. The XML below shows a fragment of the *InterpretKeyword* response associated with the request example above.


```
<rdf:RDF>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P25/current/ICECOV">
    <skos:prefLabel xml:lang="en">Ice coverage</skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/SICEAMSR">
    <skos:prefLabel xml:lang="en">
      Coverage (by area) of ice on the water body by Advanced Microwave Scanning
      Radiometer for EOS (AMSRE-E)
    </skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/SICECSAT">
    <skos:prefLabel xml:lang="en">
      Coverage (by area) of ice on the water body by image analysis
    </skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/SICECMOD">
    <skos:prefLabel xml:lang="en">
      Coverage (by area) of ice on the water body by model prediction
    </skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P07/current/CFSN0422">
    <skos:prefLabel xml:lang="en">sea_ice_amount</skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P07/current/CFSN0423">
    <skos:prefLabel xml:lang="en">sea_ice_area</skos:prefLabel>
  </skos:Concept>
  <skos:Concept
   rdf:about="http://vocab.nerc.ac.uk/collection/P07/current/CFSN0424">
    <skos:prefLabel xml:lang="en">sea_ice_area_fraction</skos:prefLabel>
  </skos:Concept>
  <!--Other concepts-->
</rdf:RDF>
```


3 CSW Mediator

The CSW mediator (CSWM) is the EUMIS component that allows access to the NETMAR distributed catalogue services, called catalogue nodes, e.g., NERSC, met.no, and IFREMER catalogue nodes. The CSWM is semantically enabled, i.e., it searches datasets and services in the catalogues by "meaning" rather than by mere keywords. To do so, it uses the SWS (c.f., Chapter 2) to interpret the user's search keywords, and rewrites the user's query to include more specific and related terms. For instance, if you search for datasets in the CSWM using the term "ice motion", the CSWM will include "sea ice speed", "sea ice displacement", "sea ice velocity", etc.

The CSWM does not harvest or index the metadata records of the catalogue nodes. Instead, it rewrites the user's query into queries supported by the catalogue nodes and executes them on the fly, then collects the answers from the different nodes and sends them back to the user.

3.1 Implementation

The CSW mediator was implemented in Java. It is deployed on the CMRC server and can be accessed at http://netmar.ucc.ie/explorer/Explorer.

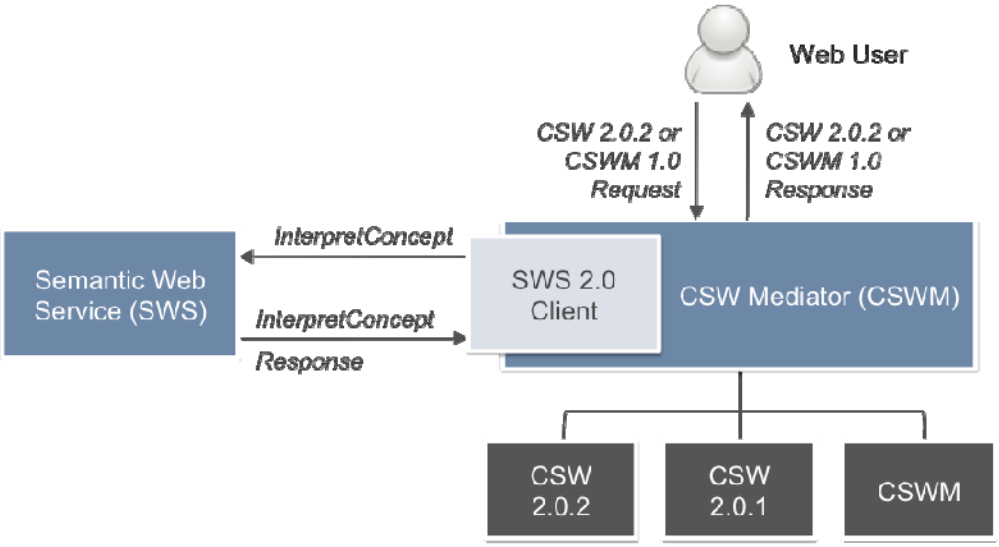

Figure 3.1. The NETMAR CSWM Implementation Architecture

As shown in Figure 3.1, the CSWM uses an SWS client to interact with the semantic web service in order to interpret the user search keywords contained in the user request. The CSWM supports three types of catalogue nodes:

1. OGC CSW 2.0.2 – This is the most common use case, as most nodes are catalogue services compliant with the CSW 2.0.2 specification, which is supported by the recent versions of GeoNetwork[4] (2.4+).
2. OGC CSW 2.0.1 – Although supported by the CSWM, we do not recommend the use of this older version of CSW as it does not allow you to specify the metadata standard in a *GetRecordById* operation (records are returned in their original standards and formats, which are not necessarily supported by the client application).

[4] http://geonetwork-opensource.org

3. CSWM 1.0 – The CSWM may access another CSWM, which in turn has catalogue nodes. This makes it possible to cascade CSWMs (c.f., section 3.2).

3.2 CSWM Chaining

A CSWM retrieves metadata records from other CSWs or CSWMs, and thus is typically part of a simple (1 CSW, 1 CSWM) or a complex (1 CSW, many cascading CSWMs) chain. When building such catalogue chains, two main problems need to be addressed:

1. Metadata record identifiers are unique within a single catalogue service node, but not necessarily unique throughout a set of distributed catalogue nodes. In addition, when processing a *GetRecordById*, the CSWM needs to know where (in which catalogue node) to fetch the requested record identified by its file ID.
2. As CSWM may be chained, it is important to avoid cascading problems related to possible catalogue chain cycles or long catalogue chains.

The following two subsections explain how these problems have been addressed in the NETMAR CSWM implementation.

3.2.1 *Metadata Record Identifiers*

When a CSWM retrieves a metadata record using a *GetRecords* or a *GetRecordById* request, it appends the identifier (URL) of the source catalogue node to the metadata record identifier. For instance, consider the metadata record identified by 67e3dcfa-5c14-4226-bba2-5741c779c896 in the met.no catalogue service, http://netmar.met.no/geonetwork/. In order to be able to resolve the source catalogue when processing a *GetRecordById* request, the CSWM mediator appends "http://netmar.met.no/geonetwork/" to the metadata record identifier, which then becomes:

67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/.

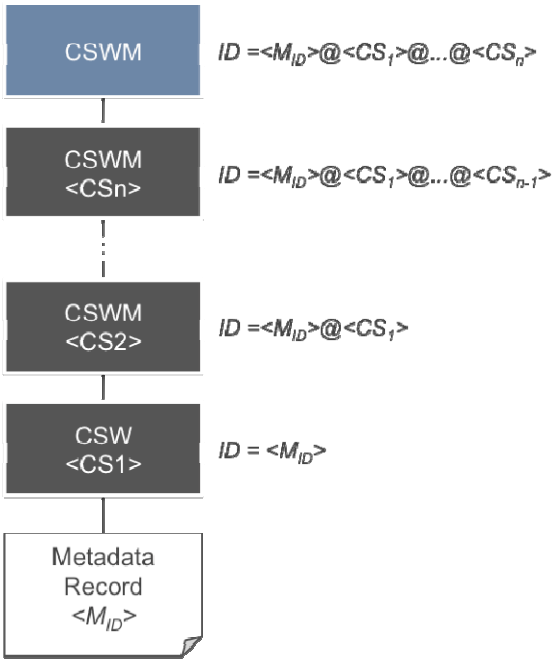

Figure 3.2. Metadata Records Identification Strategy in Catalogue Chains

In a CSWM chain, each CSWM appends the URL of the interrogated catalogue service (CSW or CSWM). Therefore, as illustrated in Figure 3.2, if a metadata record *M* is stored at a catalogue service $<CS_1>$ with the ID $<M_{ID}>$, then transmitted to a CSWM through a chain of

CSWMs, <CS_2>, …, <CS_n>, as part of a *GetRecords* Response, then its ID according to the last CSWM in the chain is:

<M_{ID}>@<CS_1>@<CS_2>@...@<CS_n>.

3.2.2 *Hop Count Control*

A *hopCount* parameter is used in the CSWM requests to limit the number of hops in catalogue chains in order to avoid cascading problems related to possible catalogue chain cycles or long catalogue chains. Like in a CSW, the *hopCount* request parameter governs the behaviour of a distributed search. It is the maximum number of message hops before the search is terminated. Each catalogue node in a catalogue chain decrements this value when the request is received, and must not forward the request if *hopCount*=0.

3.3 Request Protocols

The CSW mediator supports XML requests over HTTP.

3.4 Exception Management

Upon encountering an exception, the CSWM generates an OGC Web Service (OWS) exception report[5] stating that an exception has occurred and providing an intelligible traceback report of the error for debugging purposes.

The table below lists the various types of exceptions that might be raised by the CSWM.

Table 5.1. CSWM Exception Codes

Exception code	Description
ParsingError	The request message is not a valid XML document
InvalidParameterValue	The parameter value is not valid
InvalidRequest	The request message is either invalid or is not well-formed
NotImplemented	The (abstract) operation has not been implemented
NotFound	The requested resource does not exist or could not be found
TypeMismatch	Type mismatch error
NotSupported	A service option, feature, or capability is not supported
NoApplicableCode	There is no applicable code to this exception
NullValue	A required parameter or variable is null
InternalError	Internal error to the CSWM
ConnectionProblem	A connection problem

Particular attention was paid to providing human-readable messages as well as machine processable codes and locators in exceptions for identifying the types of exceptions and their causes.

The XML fragment below illustrates an exception report generated by the CSWM due to an empty request.

[5] http://schemas.opengis.net/ows/1.1.0/owsExceptionReport.xsd

```
<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows"
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:schemaLocation="http://www.opengis.net/ows../../../ows/1.0.0/
 owsExceptionReport.xsd"
 xml:lang="en" version="1.0.0">
  <ows:Exception exceptionCode="ParsingError">
    <ows:ExceptionText>Empty request</ows:ExceptionText>
  </ows:Exception>
</ows:ExceptionReport>
```

3.5 CSWM Operations

The CSWM has two web interfaces: CSW 2.0.2 interface and CSWM 1.0 interface. Each interface supports two operations: *GetRecords* and *GetRecordsById*.

The CSWM 1.0 interface extends CSW 2.0 by including support for catalogue nodes control (ability to specify the nodes to be included in the responses).

The subsequent subsections provide examples of CSW and CSWM *GetRecords* and *GetRecordById* requests and responses. To execute a request from your computer all you need to do is copy the request to a file; let us call it myRequest.xml for example. Then, depending on your operating system type one of the following command lines on your terminal, supposing that you are in the same folder as the request file (myRequest.xml).

For Mac OS and Linux (1 command line):

```
curl -H "Content-Type: text/xml" --data "@myRequest.xml"
http://netmar.ucc.ie/explorer/Explorer
```

For Linux and MS Windows (1 command line):

```
wget -O- --quiet --header='Content-Type: text/xml' --post-file
myRequest.xml http://netmar.ucc.ie/explorer/Explorer
```

3.5.1 *CSW 2.0.2 Operations*

As specified above, the CSW interface only supports the *GetRecords* and *GetRecordsById* operations. In this section, examples of CSW *GetRecords* and *GetRecordsById* requests and responses are provided.

3.5.1.1 CSW 2.0.2 *GetRecords* Operation

Below is an example of a CSW 2.0.2 *GetRecords* operation that aims to retrieve metadata records matching the keyword “ice coverage” with the Norwegian Sea as an area of interest (encoded as a bounding box).


```
<?xml version="1.0" encoding="UTF-8"?>
<csw:GetRecords service="CSW" version="2.0.2" maxRecords="50" startPosition="1"
 resultType="results" outputFormat="application/xml" outputSchema="csw:Record"
 xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
 xmlns:ogc="http://www.opengis.net/ogc" xmlns:ows="http://www.opengis.net/ows"
 xmlns:dc=http://purl.org/dc/elements/1.1/
 xmlns:dct="http://purl.org/dc/terms/"
 xmlns:gml="http://www.opengis.net/gml">
  <csw:Query typeNames="csw:Record">
    <csw:ElementSetName typeNames="csw:Record">summary</csw:ElementSetName>
    <csw:Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:Intersects>
            <ogc:PropertyName>ows:BoundingBox</ogc:PropertyName>
            <gml:Envelope>
              <gml:lowerCorner>
                -9.02068831025837 57.9767651784406
              </gml:lowerCorner>
              <gml:upperCorner>
                27.0790684243223 78.1219500577069
              </gml:upperCorner>
            </gml:Envelope>
          </ogc:Intersects>
          <ogc:PropertyIsLike escapeChar="\" singleChar="?" wildCard="*">
            <ogc:PropertyName>Subject</ogc:PropertyName>
            <ogc:Literal>ice coverage</ogc:Literal>
          </ogc:PropertyIsLike>
        </ogc:And>
      </ogc:Filter>
    </csw:Constraint>
  </csw:Query>
</csw:GetRecords>
```

The XML below shows a fragment of the response for the above CSW *GetRecords* request, returned by the CSWM.


```
<?xml version="1.0" encoding="UTF-8"?>
<csw:GetRecordsResponse xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2
 http://schemas.opengis.net/csw/2.0.2/CSW-discovery.xsd">
  <csw:SearchStatus timestamp="2012-07-09T10:53:11+0100"/>
  <csw:SearchResults totalNumberOfRecordsMatched="4"
   totalNumberOfRecordsReturned="4" elementSet="summary">
    <csw:SummaryRecord xmlns:geonet="http://www.fao.org/geonetwork"
     xmlns:dc=http://purl.org/dc/elements/1.1/
     xmlns:dct="http://purl.org/dc/terms/">
      <dc:identifier>
        67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
      </dc:identifier>
      <dc:title>METNO Manual Ice Chart</dc:title>
      <dc:type>dataset</dc:type>
      <dc:subject>Oceanographic geographical features</dc:subject>
      <dc:subject>
        Coverage (by area) of ice on the water body by image analysis
      </dc:subject>
      <dc:subject>Percent</dc:subject>
      <dc:subject>
        http://vocab.nerc.ac.uk/collection/P01/current/SICECSAT
      </dc:subject>
      <dc:subject>
        http://vocab.nerc.ac.uk/collection/P06/current/UPCT
      </dc:subject>
      <dc:subject>environment</dc:subject>
      <dct:abstract>
        Ice chart based on a manual interpretation of satellite data from earth
        observing satellites.
      </dct:abstract>
    </csw:SummaryRecord>
    <csw:SummaryRecord xmlns:geonet="http://www.fao.org/geonetwork"
     xmlns:dc="http://purl.org/dc/elements/1.1/"
     xmlns:dct="http://purl.org/dc/terms/">
      <dc:identifier>
        13c3c2ff-2204-4b57-8161-e42f2cb2be1f@http://netmar.met.no/geonetwork/
      </dc:identifier>
      <dc:title>OSI SAF Ice edge for the Northern Hemisphere</dc:title>
      <dc:type>dataset</dc:type>
      <dc:subject>Oceanographic geographical features</dc:subject>
      <dc:subject>
        Edge (EUMETSAT OSI-SAF) of ice on the water body by classification of
        brightness temperature and ASCAT backscatter
      </dc:subject>
      <dc:subject>Dimensionless</dc:subject>
      <dc:subject>http://vocab.nerc.ac.uk/collection/P22/current/28</dc:subject>
      <dc:subject>
        http://vocab.nerc.ac.uk/collection/P01/current/ICEEDGEX
      </dc:subject>
      <dc:subject>
        http://vocab.nerc.ac.uk/collection/P06/current/UUUU
      </dc:subject>
      <dc:subject>environment</dc:subject>
      <dct:abstract>
        The sea ice edge product from the EUMETSAT OSI SAF. Ice classes are
        assigned from atmospherically corrected SSM/I brightness temperatures
        and ASCAT backscatter values, using a Bayesian approach. It is
        operational since 2005.</dct:abstract>
    </csw:SummaryRecord>
    <!--More CSW Summary Records (csw:SummaryRecord)-->
  </csw:SearchResults>
</csw:GetRecordsResponse>
```


3.5.1.2 CSW 2.0.2 *GetRecordById* Operation

You can retrieve a metadata record by its ID from the CSWM using a CSW request. For instance, consider the first record of the *GetRecords* response example above. The ID of this record according to the CSWM is

67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/.

Note that this ID consists of two parts: the original record ID as stored in the source catalogue (i.e., 67e3dcfa-5c14-4226-bba2-5741c779c896) and the ID (URL) of the source catalogue (the met.no catalogue service, http://netmar.met.no/geonetwork/), separated by the 'at' sign: '@'. As explained in subsection 3.2.1, The CSWM appends the latter part to each record ID in the *GetRecords* response in order to be able to resolve the source catalogue when processing a *GetRecordById* request.

Below is an example of a CSW *GetRecordById* request that retrieves the above metadata record.

```
<?xml version="1.0" encoding="UTF-8"?>
<csw:GetRecordById service="CSW" version="2.0.2" outputSchema="csw:IsoRecord"
 xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
  <csw:ElementSetName typeNames="csw:Record">full</csw:ElementSetName>
  <csw:Id>
    67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
  </csw:Id>
</csw:GetRecordById>
```

A fragment of the response to this request is shown in the XML below.

```
<?xml version="1.0" encoding="UTF-8"?>
<csw:GetRecordByIdResponse xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
  <gmd:MD_Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd"
   xmlns:gco="http://www.isotc211.org/2005/gco"
   xmlns:gmx="http://www.isotc211.org/2005/gmx"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:geonet="http://www.fao.org/geonetwork">
    <gmd:fileIdentifier>
      <gco:CharacterString>
        67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
      </gco:CharacterString>
    </gmd:fileIdentifier>
    <!--more metadata elements-->
  </gmd:MD_Metadata>
</csw:GetRecordByIdResponse>
```

3.5.2 *CSWM 1.0 Operations*

As specified above, the CSWM interface supports the *GetRecords* and *GetRecordsById* operations. In this section, examples of CSWM *GetRecords* and *GetRecordsById* requests and responses are provided.

3.5.2.1 CSWM 1.0 *GetRecords* Operation

Below is an example of a CSWM 1.0 *GetRecords* operation that aims to retrieve metadata records matching the English keyword "ice coverage" with the Norwegian Sea as an area of interest (encoded as a bounding box). Note that the CSWM request includes a filter on the

catalogue nodes (federated catalogues) to be interrogated: met.no and NERSC catalogues only.

```
<?xml version="1.0" encoding="UTF-8"?>
<cswm:GetRecords service="CSWM" version="1.0" maxRecords="50" startPosition="1"
 resultType="results" outputFormat="application/xml" outputSchema="csw:Record"
 xmlns:cswm="http://cmrc.ucc.ie/cswm/1.0"
 xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
 xmlns:ogc="http://www.opengis.net/ogc" xmlns:ows="http://www.opengis.net/ows"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:dct="http://purl.org/dc/terms/"
 xmlns:gml="http://www.opengis.net/gml">
  <cswm:DistributedSearch hopCount="2">
    <cswm:FederatedCatalogues>
      <cswm:FederatedCatalogue id="http://netmar.met.no/geonetwork/"/>
      <cswm:FederatedCatalogue id="http://nport.nersc.no:8081/geonetwork/"/>
    </cswm:FederatedCatalogues>
  </cswm:DistributedSearch>
  <csw:Query typeNames="csw:Record">
    <csw:ElementSetName typeNames="csw:Record">summary</csw:ElementSetName>
    <csw:Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:Intersects>
            <ogc:PropertyName>ows:BoundingBox</ogc:PropertyName>
            <gml:Envelope>
              <gml:lowerCorner>
                -9.02068831025837 57.9767651784406
              </gml:lowerCorner>
              <gml:upperCorner>
                27.0790684243223 78.1219500577069
              </gml:upperCorner>
            </gml:Envelope>
          </ogc:Intersects>
          <ogc:PropertyIsLike escapeChar="\" singleChar="?" wildCard="*">
            <ogc:PropertyName>Subject</ogc:PropertyName>
            <ogc:Literal>"ice coverage"@en</ogc:Literal>
          </ogc:PropertyIsLike>
        </ogc:And>
      </ogc:Filter>
    </csw:Constraint>
  </csw:Query>
</cswm:GetRecords>
```

The XML below shows a fragment of the response for the above CSWM *GetRecords* request, returned by the CSWM. Note that metadata records are grouped according to their provenance.


```
<?xml version="1.0" encoding="UTF-8"?>
<cswm:GetRecordsResponse xmlns:cswm="http://cmrc.ucc.ie/cswm/1.0">
  <cswm:SearchStatus timestamp="2012-07-09T15:36:10+0100"/>
  <cswm:FederatedSearchResults numberOfResponses="5" numberOfFailures="0"
   totalNumberOfRecordsMatched="4" totalNumberOfRecordsReturned="4">
    <cswm:SearchResults numberOfRecordsMatched="0" numberOfRecordsReturned="0"
     elementSet="summary" catalogueId="http://netmar.ucc.ie/geonetwork"
     catalogueName="MIDA Catalogue Service" catalogueAcronym="MIDA"
     catalogueURL="http://netmar.ucc.ie/geonetwork/srv/en/csw"
     catalogueLogoURL="http://netmar.ucc.ie/assets/explorer/img/mida.png"/>
    <!--NERSC results (empty):-->
    <cswm:SearchResults numberOfRecordsMatched="0"
     numberOfRecordsReturned="0" elementSet="summary"
     catalogueId="http://nport.nersc.no:8081/geonetwork/"
     catalogueName="NERSC Catalogue Service" catalogueAcronym="NERSC"
     catalogueURL="http://nport.nersc.no:8081/geonetwork/srv/en/csw"
     catalogueLogoURL="http://netmar.ucc.ie/assets/explorer/img/nersc.png"/>
    <!--met.no results:-->
    <cswm:SearchResults numberOfRecordsMatched="4"
     numberOfRecordsReturned="4" elementSet="summary"
     catalogueId="http://netmar.met.no/geonetwork/" catalogueName="met.no
     Catalogue Service" catalogueAcronym="met.no"
     catalogueURL="http://netmar.met.no/geonetwork/srv/en/csw"
     catalogueLogoURL="http://netmar.ucc.ie/assets/explorer/img/metno.png">
      <csw:SummaryRecord xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
       xmlns:geonet="http://www.fao.org/geonetwork"
       xmlns:dc="http://purl.org/dc/elements/1.1/"
       xmlns:dct="http://purl.org/dc/terms/">
        <dc:identifier>
          67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
        </dc:identifier>
        <dc:title>METNO Manual Ice Chart</dc:title>
        <dc:type>dataset</dc:type>
        <dc:subject>Oceanographic geographical features</dc:subject>
        <dc:subject>
          Coverage (by area) of ice on the water body by image analysis
        </dc:subject>
        <dc:subject>Percent</dc:subject>
        <dc:subject>
          http://vocab.nerc.ac.uk/collection/P01/current/SICECSAT
        </dc:subject>
        <dc:subject>
          http://vocab.nerc.ac.uk/collection/P06/current/UPCT
        </dc:subject>
        <dc:subject>environment</dc:subject>
        <dct:abstract>
          Ice chart based on a manual interpretation of satellite data from
          earth observing satellites.
        </dct:abstract>
      </csw:SummaryRecord>
      <!--More met.no CSW Summary Records (csw:SummaryRecord)-->
    </cswm:SearchResults>
  </cswm:FederatedSearchResults>
</cswm:GetRecordsResponse>
```

3.5.2.2 CSWM 1.0 *GetRecordById* Operation

You can retrieve a metadata record by its ID from the CSWM using a CSWM request.

Below is an example of a CSW *GetRecordById* request equivalent to the CSW GetRecordById request example of subsection 3.5.1.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<cswm:GetRecordById service="CSWM" version="1.0" outputSchema="csw:IsoRecord"
 xmlns:cswm="http://cmrc.ucc.ie/cswm/1.0">
  <cswm:ElementSetName typeNames="csw:Record">full</cswm:ElementSetName>
  <cswm:Id>
    67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
  </cswm:Id>
</cswm:GetRecordById>
```

A fragment of the response to this request is shown in the XML below.

```
<?xml version="1.0" encoding="UTF-8"?>
<cswm:GetRecordByIdResponse xmlns:cswm="http://cmrc.ucc.ie/cswm/1.0">
  <gmd:MD_Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd"
   xmlns:gco="http://www.isotc211.org/2005/gco"
   xmlns:gmx="http://www.isotc211.org/2005/gmx"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:geonet="http://www.fao.org/geonetwork">
    <gmd:fileIdentifier>
      <gco:CharacterString>
        67e3dcfa-5c14-4226-bba2-5741c779c896@http://netmar.met.no/geonetwork/
      </gco:CharacterString>
    </gmd:fileIdentifier>
    <!--More metadata elements-->
  </gmd:MD_Metadata>
</cswm:GetRecordByIdResponse>
```


4 References

[D4.3.1] Lassoued, Y., Leadbetter, A., Clements, O., Grant, M., de Jesus, J., and Walker, P.: D4.3.1 – NETMAR Semantic Framework Specification – Version 1. 15 June 2011

[D4.3.2] Lassoued, Y., Leadbetter, A., Clements, O., Grant, M., de Jesus, J., and Walker, P.: D4.3.2 – NETMAR Semantic Framework Specification – Version 1. 31 October 2011

5 Acronyms

CMRC	Coastal and Marine Resources Centre
EUMIS	European Marine Information System
KVP	Key-Value Pair
NERC	Natural Environmental Research Council
NETMAR	Open service Network for Marine Environmental Data
NVS	NERC Vocabulary Server
OGC	Open Geospatial Consortium
RDF	Resource Description Framework
RDFS	(Also RDF-S) RDF Schema
SKOS	Simple Knowledge Organisation System
SWS	Semantic Web Service
URI	Uniform Resource Identifier
WPS	Web Processing Service
XML	eXtensible Markup Language

