International Association of Oil & Gas Producers

Geomatics
Guidance Note Number 7, part 3

EPSG Geodetic Parameter Registry - Developer Guide

Revision history:

Version	Date	Amendments
0.*	To April 2009	Draft versions of document.
1.0	May 2009	First release of this document, GN7 part 3.
1.1	March 2010	Correction to example code in section 4.3.1 Find Entity by BBOX
2.0	October 2011	Updated API service methods and examples (sections 4 and 5) for updated client software. Added Annex D, "Summary of August 2011 changes to registry service message format".
3.0	August 2012	Updated API service methods and examples (sections 4 and 5), as well as other references to GML, for move to GML 3.2.1. Added examples for polygons and Annex E, "Summary of August 2012 Changes to Registry".

CONTENTS

PREFACE

The EPSG Geodetic Parameter Dataset, abbreviated to the **EPSG Dataset**, is a repository of parameters required to:

- define a *coordinate reference system* (CRS) which ensures that coordinates describe position unambiguously.
- define transformations and conversions that allow coordinates to be changed from one CRS to another CRS. Transformations and conversions are collectively called *coordinate operations*.

The EPSG Dataset is maintained by the OGP Surveying and Positioning Committee's Geodetic Subcommittee. It conforms to ISO 19111 – *Spatial referencing by coordinates*. It is distributed in three ways:

- the **EPSG Registry**, in full the *EPSG Geodetic Parameter Registry*, a web-based delivery platform in which the data is held in GML using the CRS entities described in ISO 19136.
- the **EPSG Database**, in full *the EPSG Geodetic Parameter Database*, a relational database structure where the entities which form the components of CRSs and coordinate operations are in separate tables, distributed as an MS Access database;
- in a relational data model as **SQL scripts** which enable a user to create an Oracle, MySQL, PostgreSQL or other relational database and populate that database with the EPSG Dataset;

OGP Surveying and Positioning Guidance Note 7 is a multi-part document for users of the EPSG Dataset.

- *Part 0, Quick Start Guide*, gives a basic overview of the Dataset and its use.

- *Part 1, Using the Dataset*, sets out detailed information about the Dataset and its content, maintenance and terms of use.

- *Part 2, Formulas*, provides a detailed explanation of formulas necessary for executing coordinate conversions and transformations using the coordinate operation methods supported in the EPSG dataset. Geodetic parameters in the Dataset are consistent with these formulas.

- *Part 3, Registry Developer Guide*, (this document), is primarily intended to assist computer application developers who wish to use the API of the Registry to query and retrieve entities and attributes from the dataset.

- *Part 4, Database Developer Guide*, is primarily intended to assist computer application developers who wish to use the Database or its relational data model to query and retrieve entities and attributes from the dataset.

The complete text may be found at http://www.epsg.org/guides/index.html. The terms of use of the dataset are also available at http://www.epsg.org/CurrentDB.html.

In addition to these documents, the Registry user interface contains online help and the Database user interface includes context-sensitive help accessed by left-clicking on any label.

This Part 3 of the multipart Guidance Note is primarily intended to assist computer application developers in using the EPSG geodetic parameter registry. It may also be useful to other users of the data. Readers are recommended to have read Part 1 of the guidance note before this part. Part 4 section 2 may also serve to provide tips for searching and its annex E examples that, with modification, may be applied to the registry.

1 INTRODUCTION

1.1 The Registry Geodetic Model, ebRIM and GML

The EPSG Registry geodetic model is in accordance with OGC Abstract Specification Topic 2, which, in turn, is equivalent to *"ISO 19111:2007 Geographic information – Spatial referencing by coordinates"* and has been implemented in GML through *"ISO 19136 Geographic information – Geographic mark-up language (GML)"*. The EPSG Registry model implementation contains a number of additional object types intended for data management or life cycle management purposes. These are detailed in Annexes A and B.

The EPSG Registry is based on the ebRIM 3.0 specification, a copy of which is available at http://docs.oasis-open.org/regrep/v3.0/specs/regrep-rim-3.0-os.pdf.

All EPSG Registry entities have been encoded in GML 3.2.1. See Annex C for a reference to the defining documentation.

The registry uses a propriatory product, INdicio, to manage web service interfacing. The INdicio software was updated in August 2011. As a consequence of this upgrade the registry service interface message content has changed significantly. The API message content changes are summarised in Annex D.

In August 2012 the EPSG GML schema were changed. See Annex E for a summary of changes. This version of this Guidance Note has been updated to reflect these changes.

1.2 Conventions Used in This Guidance Note

Instructions in this manual were written for a Windows environment; some things may be different in a UNIX/Linux environment. Please make appropriate changes as necessary, such as replacing **%STR%** with "**$STR**" in commands, replacing "\" with "/" in file names, and replacing "*.bat*" with "*.sh*" in file extensions.

The following text formatting conventions are used throughout this Guidance Note:

- Folder names are indicated in bold **...\<Tomcat>\bin** and file names are indicated in italic, for example: *server.xml*.
- Menu names and application names use a ***sans serif bold italic*** font.
- Interface control names use a *sans serif italic* font.
- Interface [Button] names and keyboard [Key] names are enclosed in square brackets.
- `Code samples`, and XML `element` and `attribute` names, are shown in a fixed width font.
- Code elements that are `[optional]` are indicated with square brackets.
- Code elements to be replaced with `<user_values>` are indicated in angle brackets.

Note Notes are used to highlight important information.

1.3 Glossary of terms

Term	Definition
valid entities	A valid entity is one that, at this moment, contains numerically and contextually correct information. Some valid entities may be superseded or retired in practical usage, but they remain valid in terms of dataset content. A valid entity has a deprecation status of "false".
invalid entities	An invalid entity is one that contains a significant error and has been withdrawn from the EPSG dataset content. It should not be used for the production of new data. It is not physically withdrawn from the registry and may be referenced for historical purposes. Invalid entities have a deprecation status of "true". See Guidance Note 7 part 1 for the deprecation policy applied to the EPSG Dataset.
guest user	A guest user is any user who does not have an account as a registered user or a registered user who is not logged in.
registered user	A registered user is any user who has registered an account with a registry and who is logged in. A registered user may export the registry as a GML dictionary.
GML schema	The GML schema describes at a relatively high level of abstraction, and provides a way to validate, the geodetic data in terms of the structure of the information, constraints on the structure, and the content of a GML document.
application schema	An application schema extends the GML schema with domain-specific extensions held in metadata.
GML document	A GML document is an XML string following the provisions of ISO 19136 and the Geography Markup Language (GML) Encoding Specification, and with a defined GML schema.
RepositoryItem	A RepositoryItem is a GML document describing a dataset entity. It has metadata contained in an associated ExtrinsicObject. See also: Section 2: "RepositoryItems"
RegistryObject	A RegistryObject is an entity that contains metadata about a RepositoryItem and can be used to retrieve the RepositoryItem from the registry. See also: Section 2: "RegistryObjects"
ExtrinsicObject	An ExtrinsicObject is a specific type of RegistryObject that is associated with a RepositoryItem. See also: Section 2: "ExtrinsicObject"
Association	An Association describes a many-to-many relationship among RegistryObjects. See also: Chapter 2: "Association"
Classification	A Classification is a specialized form of Association that is used to classify RegistryObjects in some way. See also: Chapter 2: "Classification"
Slot	A Slot is a means of adding attributes to RegistryObjects. See also: Chapter 2: "Slots"
area of use	Area entities describe the region(s) to which geodetic entities, for example a Coordinate Reference System, may be applicable. See Guidance Note 7 part 1 for further information.

2 REGISTRY ELEMENTS

2.1 Overview

The EPSG Registry stores GML documents as RepositoryItems in a *repository*, and standardized metadata describing the content as RegistryObjects in a *registry*. Both the *registry* and the *repository* are components of the EPSG Registry itself.

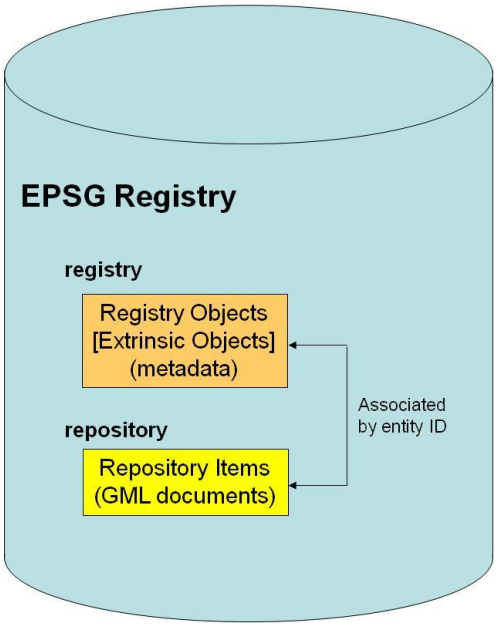

Figure 1: EPSG Registry overview

2.2 ebRIM

The registry component is based on the ebRIM 3.0 specification, a copy of which is available at http://docs.oasis-open.org/regrep/v3.0/specs/regrep-rim-3.0-os.pdf. Some key concepts and information from the specification are described here in order to provide context and define terminology that will be used throughout this guide.

The ebRIM specification describes an information model for the classes, objects, associations, services, etc. that are needed to support an information system that securely organizes and manages different types of content and the metadata describing it. An ebXML registry is capable of storing any type of electronic content such as XML documents, text documents, images, sound, and video

The following UML diagram shows the relationship between the relevant generic ebRIM objects implemented in the EPSG Registry and described in the following sections:

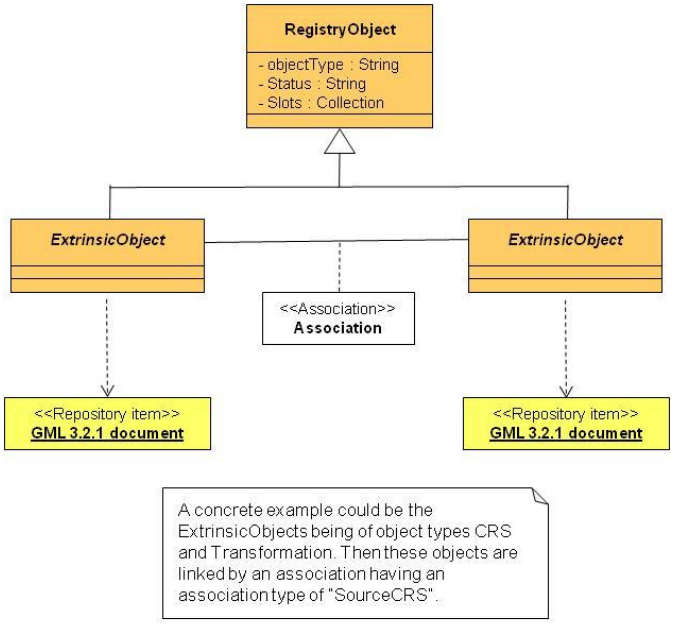

Figure 2: ebRIM objects implemented in the EPSG Registry

2.3 Registry Components

2.3.1 RegistryObjects

A RegistryObject is an abstract class. For the EPSG Registry, the most significant RegistryObject are the ExtrinsicObjects which are derived from it.

2.3.2 ExtrinsicObject

An ExtrinsicObject is a specialization of the RegistryObject class. An ExtrinsicObject contains the metadata for a RepositoryItem. Each ExtrinsicObject may be associated to a related RepositoryItem which is held in the repository. No other ExtrinsicObject may be associated with that RepositoryItem. Each ExtrinsicObject has an identifier that is used to refer to its associated RepositoryItem, if one exists.

ExtrinsicObjects may be thought of as the index cards stored in the card catalogue of a library, where each index card describes one book that is stored in the library. In the same way, each ExtrinsicObject stored in the registry contains the metadata for one GML document stored in the repository as a RepositoryItem. Every RepositoryItem must have an ExtrinsicObject that describes it in the same way that every book in a library must have an index card in the card catalogue.

2.3.3 RepositoryItems

A geodetic entity in the dataset is defined through data in a GML document, that document being stored in the repository as a RepositoryItem. RepositoryItems may be thought of as the books in a library; each book is described on one index card in the card catalogue. In the EPSG Registry, RepositoryItems are GML version 3.2.1 documents.

.

2.3.4 Association

A RegistryObject may be connected to other RegistryObjects through an Association. Associations define a many-to-many relationship between RegistryObjects. An Association represents a relationship between a sourceObject and a targetObject. It is important to know which object is the source and which is the target, as an Association has the concept of direction.

Associations may be thought of as "see also" references which may be noted on the index card of one book in order to refer to another book that is related to it in some way.

Each Association must have an attribute which identifies what type of association it is. The value of the associationType attribute is specified by the identifier (ID) of one of the ClassificationNodes in the Association type ClassificationScheme.

2.3.5 Classification

A Classification is a specialized form of Association. RegistryObjects may be classified in many ways such as by industry, product, geographical location, and so on. Classification may be thought of as the taxonomy used to organize the RegistryObjects in the same way that the Dewey Decimal Classification System is used to organize the books in a library.

The information model defines classes and relationships so that both single level and multi-level Classifications can be supported. For example, in the model that underlies the EPSG Registry, Coordinate Reference Systems are classified as GeodeticCRS, ProjectedCRS, EngineeringCRS and VerticalCRS.

A general ClassificationScheme can be viewed as a tree structure of ClassificationNodes that describes a hierarchical taxonomy for organizing and classifying RegistryObjects by referencing a node within the scheme.

2.3.6 Slots

Slots provide a means to add attributes to RegistryObject instances, thus providing extensibility of the information model. Slots may be thought of as spaces on an index card that a librarian or a reader may fill in with notes that provide additional information about the book described on the card. See Annex B for a list of slots used in the EPSG Registry.

3 ENTITY LIFE CYCLE STATUS AND ACCESS CONTROL

3.1 Entity life cycle status

The life-cycle of an entity is defined by the status attribute of its RegistryObject. Each RegistryObject instance has a life cycle status attribute. The status of a RegistryObject will change throughout its life cycle. The following table lists the possible states that a RegistryObject may be in as defined by the value of its status attribute and relates them to typical EPSG terminology.

Entity Validity	EPSG term	Registry life-cycle status	Description
Valid	Published	Approved	An entity that has been publicly released and which has not been deprecated.
	Superseded	Superseded with supersessionType property = "Supersession"	A released entity that is no longer preferred for new spatial datasets, but it may still be used in some business sectors for various reasons, e.g. historical continuity.
	Retired	Superseded with supersessionType property = "Retirement"	A released entity describing information that has been withdrawn by its information source; it may, or may not, have been replaced. The entity may be referenced by old datasets and is still valid in the sense that it is free from errors. It should no longer be used for new datasets.
Invalid	Deprecated	Deprecated	Entities that have been identifed as invalid. Deprecated entities contain significant error and should only be used when documenting or reversing the use of the entity that was made before it was declared invalid. Deprecated entities remain in the dataset.
Not released	Pending	Submitted	Entities that have not yet been published. They may be either entirely new entities or versions of valid entities with minor amendments applied.

From the perspective of the user, the EPSG Dataset comprises of valid and deprecated records, but excludes pending data.

3.2 Access control

The access control levels for the EPSG Registry are dependent upon user status and entity status.

3.2.1 API access to records

Through the API it is possible to access all Valid records as well as Deprecated records. Records with a status of 'Submitted' cannot be accessed through the API.

3.2.2 Graphic User Interface access to records

The GUI is accessible through an internet browser using the URL http://www-epsg-registry.org. It offers a facility to browse contents, generate reports, and export a GML Dictionary of individual entities or of the complete EPSG Dataset. For members of the OGP Geodesy Subcommittee only, it additionally allows facilities for data maintenance. Only OGP Geodesy Subcommittee members have access to pending records.

The following user statuses are available:

- Guest User
- Registered User

3.2.2.1 GUI Guest User

Guest access is the default access level to the GUI. The Guest users are users that do not have a registered account in the EPSG Registry or are Registered Users who are accessing the registry without logging in. Guest Users have access only to Valid entities; they do not have access to records with status of 'Deprecated' or 'Submitted'. They can download GML dictionaries of individual Valid entities but not of the whole dataset.

3.2.2.2 GUI Registered User

Registered Users are users that have registered a user account with the EPSG Registry and have logged in through the GUI. The EPSG Registry maintains a list of user accounts. Registered Users are able to access both Valid and (at their option) 'Deprecated' records, but are not able to access records with status of 'Submitted'. Registered Users are able download GML dictionaries of individual Valid or 'Deprecated' entities and to use the capability to export the full dataset as a GML dictionary or as a SQL script

To apply for a Registered User account with the EPSG Registry:

1) Open a browser window and enter the address for the EPSG Geodetic Parameter Registry:

   http://www.epsg-registry.org/

2) Click on the link to *(login or register)*;

3) On the Registry Login page, click on the link to *Register as a new user*;

4) Read through and accept the Terms of Use for the EPSG Geodetic Parameter Registry;

5) Fill in all the required fields in the form and add any additional information as desired;

6) Submit the form to create an account.

4 API SERVICE METHODS

The EPSG Registry API is strictly read-only; it only supports finding and retrieving of entities from the database. This section lists the service methods provided by the API and describes how to use them. In the following examples, `<host>` is the IP address for the registry, and `<port>` is the INdicio port number. For the EPSG Registry the port number is not necessary.

GML 3.1.1 is used for queries because the ebRIM Profile dictates the use of GML 3.1.1. GML 3.2 is used to define entities in the EPSG Geodetic Parameter Dataset because GML 3.2 is the application schema that was chosen for the registry model. A copy of the application schema is contained within the Developer Guide Package.

By default, queries from registered users will return both valid and deprecated (invalid) entities. Deprecated entities include significantly erroneous information and, in general, should not be used. Therefore, in general, queries from registered users should filter out entities with the status of 'Deprecated'. There may, however, occasionally be a need to use the information in a deprecated entity, for example to rework computations which used the erroneous information before the entity was deprecated.

The Application Model for the EPSG Registry is described in Annex A.

4.1 Basic Query Mechanisms

Two methods of query are used to access ExtrinsicObjects and RepositoryItems respectively:

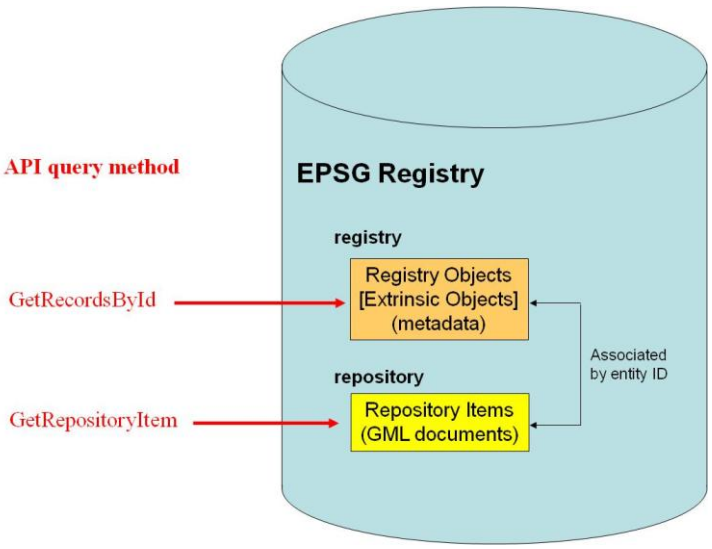

Figure 3: Registry query mechanisms

4.1.1 GetRecords

The query endpoint allows a user to execute a number of different requests against the registry service (INdicio). The query can be executed by performing an `HTTP GET` or `POST` request on the following URL:

```
HTTP://<host>:<port>/indicio/query
```

For the EPSG Registry `:<port>` is not required (as it uses the default port 80). The following URL is used:
http://www.epsg-registry.org/indicio/query

For `POST` the desired query is attached in the message body (XML), and for `GET` the correct KVP parameters are specified. In most cases a `GetRecords (POST)`, `GetRecordById (GET)`, or `GetRepositoryItem (GET)` request is used to query the registry and retrieve the desired ebRIM records or underlying GML data.

4.1.1.1 GetRecordById

The `GetRecordById` method retrieves a single ebRIM metadata record for an entity. To execute a `GetRecordById` query, perform an `HTTP GET` request on the following URL:

```
http://<host>:<port>/indicio/query?request=GetRecordById&id=[ID]&ElementSetName
=full
```

`[ID]` is the URN identifier of the item to be retrieved, and `ElementSetName` specifies the amount of information INdicio will return about the object (allowed values are `full`, `summary`, and `brief`).

Specifying ElementSetName=brief returns the following information:

Information	Description
`rim:RegistryObject/@id`	the identifier for the object
`rim:RegistryObject/@objectType`	a URN from the ObjectType ClassificationScheme
`rim:RegistryObject/@status`	a URN from the StatusType ClassificationScheme
`rim:RegistryObject/rim:VersionInfo`	the object specific version details

Specifying `ElementSetName=summary` returns the same amount of information as for the `brief` view, plus the following:

Information Level	Description
`rim:RegistryObject/rim:Slot`	additional entity properties
`rim:RegistryObject/rim:Name`	a short descriptive name for the object
`rim:RegistryObject/rim:Description`	a description of the object

Specifying an `ElementSetName=full` returns a complete representation of all entity properties.

4.1.2 GetRepositoryItem

The `GetRepositoryItem` method retrieves the definitive GML form of an entity through an `HTTP GET` request using the `id` of the related ExtrinsicObject.

To execute a `GetRepositoryItem` query, perform an `HTTP GET` on the following URL:

```
http://<host>:<port>/indicio/query?request=GetRepositoryItem&id=[ID]
```

This will return the RepositoryItem with the specified `[ID]`. The mime type of the HTTP response will be the mime type of the ExtrinsicObject. For example, the request:

```
http://www.epsg-registry.org/indicio/query?request=GetRepositoryItem&id=
urn:ogc:def:crs:EPSG::4610
```

will return the repository GML data for CRS code 4610.

4.2 Property Based Queries

The following is a case-insensitive query that will return a list of all entities that have the string "`paris`" in either their `Name` or `Description`:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Query on Entity by matching a search term to its Name or Description
    property. -->
  <Query typeNames="wrs:ExtrinsicObject rim:ClassificationNode_cnode3_cnode5">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- Search the entity's textual content using wildcards -->
          <ogc:Or>
            <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
              <?indicio-case-sensitive false?>
                 <ogc:PropertyName>/wrs:ExtrinsicObject/rim:Name/rim:LocalizedS
                 tring@value</ogc:PropertyName>
              <ogc:Literal>*paris*</ogc:Literal>
            </ogc:PropertyIsLike>
            <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
              <?indicio-case-sensitive false?>
                 <ogc:PropertyName>/wrs:ExtrinsicObject/rim:Description/rim:Loc
                 alizedString@value</ogc:PropertyName>
              <ogc:Literal>*paris*</ogc:Literal>
            </ogc:PropertyIsLike>
          </ogc:Or>
          <!-- Restrict ObjectType of Entity to an EPSG entity type; this
          excludes EPSG Metadata record types such as Change Request,
          Deprecation, etc.  -->
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$cnode3/@parent</ogc:PropertyName>
              <ogc:Literal>urn:ogc:def:ObjectType:GML:Geodetic-
Entity</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Or>
                <ogc:PropertyIsEqualTo>
                 <ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:Proper
          tyName>
                   <ogc:PropertyName>/$cnode3/@id</ogc:PropertyName>
                </ogc:PropertyIsEqualTo>
                <ogc:And>
                   <ogc:PropertyIsEqualTo>
                   <ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:Proper
                   tyName>
                     <ogc:PropertyName>/$cnode5/@id</ogc:PropertyName>
                   </ogc:PropertyIsEqualTo>
                   <ogc:PropertyIsEqualTo>
                     <ogc:PropertyName>/$cnode5/@parent</ogc:PropertyName>
                     <ogc:PropertyName>/$cnode3/@id</ogc:PropertyName>
                   </ogc:PropertyIsEqualTo>
                </ogc:And>
            </ogc:Or>
          </ogc:And>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.2.1 Find Entity by Type

Unless specified explicitly, queries will return both valid and deprecated (invalid) entities. Deprecated entities include significantly erroneous information and in general should not be used. (There may occasional be a need to use them to rework computations using the erroneous information). The following query will return a list of all valid (i.e. not deprecated) Projected CRS entities.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Query on Entity by entity-type. -->
  <Query typeNames="wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- Specify the entity type via a URN from the ObjectType
          ClassificationScheme -->
          <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:ObjectType:GML:ProjectedCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/wrs:ExtrinsicObject/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.2.2 Find Entity by Code

The following query will return a list of entities with code "1250":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <Query typeNames="wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:Literal>*:1250</ogc:Literal>
          </ogc:PropertyIsLike>
          <!-- Typically these metadata records (non-entities) would be
          excluded from the result set -->
          <ogc:Not>
            <ogc:Or>
              <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
                <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:deprecation</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <ogc:PropertyIsEqualTo>
```

```
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
                  <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:supersession</ogc:Literal>
               </ogc:PropertyIsEqualTo>
               <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
                  <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:ReleaseDelta</ogc:Literal>
               </ogc:PropertyIsEqualTo>
            </ogc:Or>
          </ogc:Not>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.2.3 Find Entity by Type + Code

Because codes are only unique within an entity type, a query for entities with a specific code may return more than one result. Qualifying a search by code tthrough adding criteria for the entity type or name will narrow the search results. The following query will return the specific Coordinate Transformation with code "1250":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
<!-- Query on Entity Code and by ObjectType. By also constraining the results
by ObjectType we eliminate entities which are not of interest. -->
  <Query typeNames="wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:Literal>*:1250</ogc:Literal>
          </ogc:PropertyIsLike>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
<ogc:Literal>urn:ogc:def:ObjectType:GML:CoordinateTransformation</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.2.4 Find Entity by Name + Type

The following query will return a list of any Projected CRS with the string "scotia" in its name that is also a valid entity:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Query on Entity by matching a search-term to its Name or Alias property.
  -->
  <Query typeNames="wrs:ExtrinsicObject">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
```

```
          <ogc:And>
            <!-- Specify the entity type via a URN from the ObjectType
            ClassificationScheme -->
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:ogc:def:ObjectType:GML:ProjectedCRS</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Or>
              <!-- Search the textual content of Name and Alias properties -->
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/wrs:ExtrinsicObject/rim:Name/rim:LocalizedString/@value</ogc
:PropertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
                <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                     <?indicio-case-sensitive false?>
                     <ogc:PropertyName>wrs:ExtrinsicObject/rim:Slot[@name =
'EntityAlias']/rim:ValueList/rim:Value</ogc:PropertyName>
                     <ogc:Literal>*[*scotia*]*</ogc:Literal>
                </ogc:PropertyIsLike>
            </ogc:Or>
            <ogc:Not>
              <!-- Constrain the Status of the entity to be Valid -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/wrs:ExtrinsicObject/@status</ogc:PropertyName>
                <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
              </ogc:PropertyIsEqualTo>
            </ogc:Not>
          </ogc:And>
        </ogc:Filter>
      </ogc:Constraint>
    </Query>
</GetRecords>
```

4.3 Spatial Queries

Certain geodetic entity types in the EPSG Registry have an association with an Area (of use). The geodetic entities associated with an Area are Coordinate Reference System, Datum and Coordinate Operation. All other entity types (for example ellipsoid) are not associated with and Area. Areas in the EPSG Registry include a textual description and bounding box coordinates; valid Areas will usually also have a polygon. See figure 4.

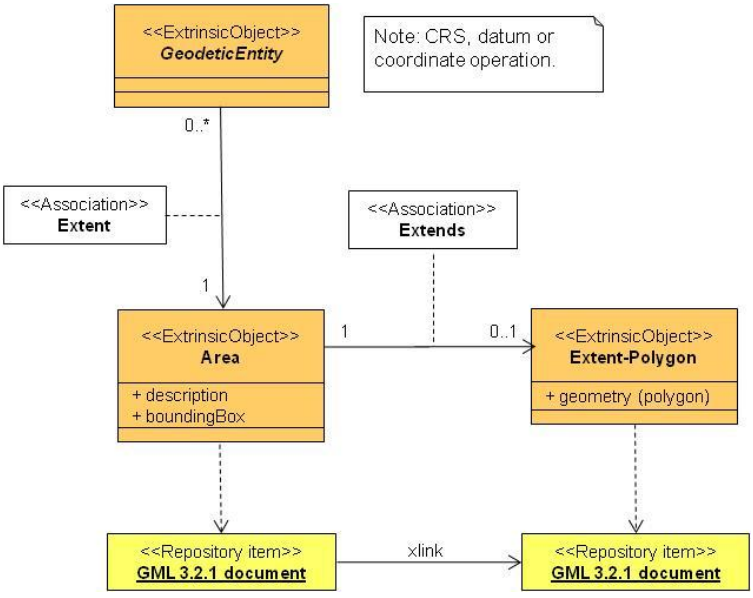

Figure 4: Object Relationships available for use in Spatial Queries

Spatial queries retrieve geodetic entities (CRS, datum, coordinate operation) from the dataset based on either the bounding box or the geometry of the Area associated with the geodetic entity. Queries of this type involve either entering the coordinates for a user-defined bounding box, or by specifying a user-defined geometry.

The supported GML 3.1.1 Envelope and Geometry types that may be used in queries are:

`gml:Envelope`	`gml:Point`	`gml:MultiPoint`
	`gml:Polygon`	`gml:MultiPolygon`
	`gml:LineString`	`gml:MultiLineString`

The supported OGC Filter 1.1.0 Spatial Operators are:

BBOX	Intersects	Within
Equals	Touches	Contains
Disjoint	Crosses	Overlaps

4.3.1 Querying Against an Area Envelope

4.3.1.1 Find Entity by BBOX

This query compares a search geometry, here a GML Envelope (or BBOX), against Area of Use Envelopes. Entities are returned if their related Area of Use Envelope matches according to the chosen spatial operator, here ogc:BBOX is used.
The following query will return all entities whose related Area of Use Envelope interacts with the defined bounding box area (which contains the region around the City of Vancouver):

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
   <!-- Use a Spatial query (BBOX) to match an Area record, then select the
   "Valid" entity records that are related to the selected Area via an "Extent"
   Association.  -->
  <Query typeNames="wrs:ExtrinsicObject_entity_extent
       rim:Association rim:ClassificationNode">
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@sourceObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The target of the association is the area specified
               as search criteria (BBOX or Envelope) -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@targetObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The association type must be correct -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@associationType
              </ogc:PropertyName>
              <ogc:Literal>
                urn:x-ogp:def:AssociationType:EPSG:Extent
```

```
                </ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <!-- Specify that the query is against EPSG Area records -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
                <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <!-- Specify the bounding box for the search -->
              <ogc:BBOX>
                <ogc:PropertyName>
                  /$extent/rim:Slot[@name =
                  'http://www.opengis.net/gml/Envelope']
                  /wrs:ValueList/wrs:AnyValue
                </ogc:PropertyName>
                <gml:Envelope srsName="urn:ogc:def:crs:EPSG::4326">
                  <gml:lowerCorner>49.1 -123.3</gml:lowerCorner>
                  <gml:upperCorner>49.3 -122.8</gml:upperCorner>
                </gml:Envelope>
              </ogc:BBOX>
            </ogc:And>
            <ogc:Not>
              <!-- Constrain the Status of the entity to be Valid -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>
                  /$entity/@status
                </ogc:PropertyName>
                <ogc:Literal>
                  urn:oasis:names:tc:ebxml-regrep:StatusType:Deprecated
                </ogc:Literal>
              </ogc:PropertyIsEqualTo>
            </ogc:Not>
            <ogc:And>
              <!-- Find the classification node for the object type -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/rim:ClassificationNode/@id</ogc:PropertyName>
                <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
              </ogc:PropertyIsEqualTo>
              <ogc:Or>
                <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                  <ogc:PropertyName>
                    /rim:ClassificationNode/@path
                  </ogc:PropertyName>
                  <ogc:Literal>
/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*
                  </ogc:Literal>
                </ogc:PropertyIsLike>
                <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                  <ogc:PropertyName>
                    /rim:ClassificationNode/@path
                  </ogc:PropertyName>
                  <ogc:Literal>
/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/EPSG-
Metadata*
                  </ogc:Literal>
                </ogc:PropertyIsLike>
              </ogc:Or>
            </ogc:And>
          </ogc:And>
        </ogc:Filter>
      </Constraint>
    </Query>
</GetRecords>
```

4.3.1.2 Find Entity by Geometry

This query compares a search geometry, here a Polygon, against Area of Use Envelopes. Entities are returned if their related Area of Use Envelope matches according to the chosen spatial operator, here ogc:Intersects is used. Other filter operators may be combined with geometries to create more sophisticated queries.

The following query returns valid entities whose Area of Use Envelope intersects the region defined by the coordinates of the specified polygon (in this example around Stonehenge in the UK):

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
   <!-- Use a Spatial operator to match an Area related to an entity of
   interest; constrain the entity to have a Status of "Approved" or
   "Superseded". -->
  <Query typeNames="rim:Association wrs:ExtrinsicObject_entity_extent">
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@sourceObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The target of the association is an area record -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@targetObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The association type must be 'Extent' -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /rim:Association/@associationType
              </ogc:PropertyName>
              <ogc:Literal>
                urn:x-ogp:def:AssociationType:EPSG:Extent
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- Specify that the spatial component of the query
                 is against EPSG Area of Use records -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-Specify the spatial operation, here a polygon -->
            <ogc:Intersects>
              <ogc:PropertyName>
                /$extent/rim:Slot[@name =
                'http://www.opengis.net/gml/Envelope']
                /wrs:ValueList/wrs:AnyValue
              </ogc:PropertyName>
              <gml:Polygon srsDimension="2"
                  srsName="urn:ogc:def:crs:EPSG::4326" gml:id="test1">
                <gml:exterior>
                  <gml:LinearRing>
                    <gml:posList>51.17 -1.83 51.17 -1.815 51.181 -1.815
                       51.181 -1.83 51.17 -1.83</gml:posList>
                  </gml:LinearRing>
                </gml:exterior>
              </gml:Polygon>
            </ogc:Intersects>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>
                urn:oasis:names:tc:ebxml-regrep:StatusType:Deprecated
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
        </ogc:And>
      </ogc:Filter>
```

```
      </Constraint>
    </Query>
  </GetRecords>
```

4.3.2 Querying Against an Area Polygon

4.3.2.1 Find Entity by BBOX

This query compares a search geometry, here a GML Envelope (or BBOX), against Area of Use **Polygon**. Entities are returned if their related Area of Use **Polygon** matches according to the chosen spatial operator, here ogc:BBOX is used.

The following query will return all entities whose related Area of Use **Polygon** interacts – as defined by the ogc:BBOX operator – with the defined bounding box area:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords
    xmlns="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
    xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
    xmlns:gml="http://www.opengis.net/gml/3.2"
    xmlns:ogc="http://www.opengis.net/ogc"
    outputFormat="application/xml; charset=UTF-8"
    service="WRS" version="1.0.1"
    startPosition="1" maxRecords="10"
    resultType="results">
    <!--
      Use a Spatial operator to match an Area related to an entity of interest;
      spatial query targets the AoU-related Polygon instead of AoU Envelope.
    -->
  <Query typeNames="Association_extentAssoc_extpolyAssoc
      ExtrinsicObject_entity_extent_polygon">
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extentAssoc/@sourceObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The target of the association is an area record -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extentAssoc/@targetObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The association type must be 'Extent' -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extentAssoc/@associationType
              </ogc:PropertyName>
              <ogc:Literal>
                urn:x-ogp:def:AssociationType:EPSG:Extent
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- specify that the entity is related to the Polygon metadata
              record through the EPSG Area of Use records -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- source of the association is the Polygon metadata record -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extpolyAssoc/@sourceObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$polygon/@id</ogc:PropertyName>
```

```
            </ogc:PropertyIsEqualTo>
            <!-- The target of the association is an area record -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extpolyAssoc/@targetObject
              </ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- The association type must be 'Extends' -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>
                /$extpolyAssoc/@associationType
              </ogc:PropertyName>
              <ogc:Literal>
                urn:oasis:names:tc:ebxml-regrep:AssociationType:Extends
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$polygon/@objectType</ogc:PropertyName>
              <ogc:Literal>
                urn:x-ogp:def:ObjectType:EPSG:extent-polygon
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- specify the spatial operation, here Intersects -->
            <ogc:Intersects>
              <!-- identify target geometry, here it's a Polygon
                in slot named '../spatial' -->
              <ogc:PropertyName>
                /$polygon/rim:Slot[@name =
                'http://purl.org/dc/terms/spatial']/wrs:ValueList/wrs:AnyValue
              </ogc:PropertyName>
              <gml:Polygon srsDimension="2"
                  srsName="urn:ogc:def:crs:EPSG::4326" gml:id="test2">
                <gml:exterior>
                  <gml:LinearRing>
                    <gml:posList>51.17 -1.83 51.17 -1.815 51.181 -1.815
                       51.181 -1.83 51.17 -1.83</gml:posList>
                  </gml:LinearRing>
                </gml:exterior>
              </gml:Polygon>
            </ogc:Intersects>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>
                urn:oasis:names:tc:ebxml-regrep:StatusType:Deprecated
              </ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.4 Association Based Queries

Association based queries retrieve entities from the dataset based on their association to an entity specified in the query. Queries of this type involve traversing one or more associations to retrieve entities associated with the original entity.

4.4.1 Find CRS by Datum

The following query will find and return all the CRS entities that are based upon (use) the specified datum code.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
```

```
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Return all CRS entities which used the specified Datum. -->
  <Query typeNames="rim:Association wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The source of the association is the object we return -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:Datum</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- The ID of a Datum must be specified -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:datum:EPSG::6003</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.4.2 Find Projected CRS by Base CRS

The following query will return all the Projected CRS entities that use the base CRS having code "4600":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Return all CRS entities which use the specified BaseCRS code. -->
  <Query typeNames="rim:Association wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The source of the association is the object we return -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:BaseCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- The ID of a BaseCRS must be specified -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:crs:EPSG::4600</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.4.3 Find Coordinate Operation by Method Used

The following query will return coordinate operations that use the method having code "9664":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Return all Coordinate Operation entities which use the specified Method
code.  -->
  <Query typeNames="rim:Association wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The source of the association is the object we return -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:Method</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- The ID of a Operation Method must be specified -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:method:EPSG::9664</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

4.4.4 Find Coordinate Operation by CRS

The following query will return all coordinate operations that use the CRS having code "4973":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
   <!-- Return all Coordinate Operation entities which use the specified Source
   CRS code. -->
  <Query typeNames="rim:Association wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The source of the association is the object we return -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
<ogc:Literal>urn:ogc:def:AssociationType:GML:SourceCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- The ID of a CRS must be specified -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:crs:EPSG::4973</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
```

```
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

> **Note:** Many transformation methods are reversible and allow the coordinate transformation to be used both from CRS 'A' to CRS 'B' and from CRS 'B' to CRS 'A'. A coordinate transformation between CRS 'A' and CRS 'B' which is reversible is stored in the EPSG Registry only once. If a user wishes to derive information about this coordinate transformation between CRS 'A' and CRS 'B', the described query should be run twice (or extended), once with CRS 'A' as SourceCRS and CRS 'B' as TargetCRS and once with CRS 'B' as SourceCRS and CRS 'A' as TargetCRS. This will find the transformation irrrespective of the direction (A to B or B to A) in which it is stored. In both cases multiple transformation entities may be returned. OGP Geomatics Guidance Note 13 provides advice on the strategy to adopt and select the most appropriate Coordinate Transformation version or variant.

4.5 Query by Slot

The following query will find all Geodetic CRS records that have a Slot ("EntitySubType") with value "geographic 3D":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Query Geographic 3D CRS records. -->
  <Query typeNames="wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The type of the entity we are searching for is GeodeticCRS -->
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:ObjectType:GML:GeodeticCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- The sub-type is specified in the EntitySubType slot -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/wrs:ExtrinsicObject/Slot[@name =
'EntitySubType']/ValueList/Value</ogc:PropertyName>
            <ogc:Literal>geographic 3D</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

5 API SERVICE EXAMPLES

This section is a collection of query examples organized around a series of real-world scenarios for using the EPSG Registry dataset. The queries use the methods for the basic query mechanisms described in section 4.

5.1 How To... Dereference URNs

Accessing the value that a reference points to instead of the reference itself is known as "dereferencing" it. A reference is an identifier that points to an object located somewhere else. Using references improves flexibility and allows objects to be stored in separate locations, allocated in different ways, and pointed to from within the code instead of being included in it. References may be followed from one object to another to lead from a known object to an unknown one.
Whenever an xlink is encountered, the URN may be resolved to its referenced object by:

- using a GetRecordById request;
- using the same id to retrieve the associated GML document using a GetRepositoryItem request.

5.2 How To... Check the Dataset Version

The following query will return a list of Version History records for the dataset in the registry.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Retrieve approved Version History records. Sort by the Version History
    Slot named "VersionDate"; use Ascending order so the last record in the
    result list will represent the current version of the dataset. -->
  <Query typeNames="wrs:ExtrinsicObject">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
            <ogc:PropertyName>wrs:ExtrinsicObject/@id</ogc:PropertyName>
            <ogc:Literal>*:EPSG::*</ogc:Literal>
          </ogc:PropertyIsLike>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
            <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:version-
history</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>wrs:ExtrinsicObject/@status</ogc:PropertyName>
            <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Approved</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>wrs:ExtrinsicObject/rim:Slot/@name</ogc:PropertyName>
            <ogc:Literal>VersionDate</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
    <ogc:SortBy>
      <ogc:SortProperty>
        <ogc:PropertyName>wrs:ExtrinsicObject/rim:Slot[@name =
'VersionDate']/rim:ValueList/rim:Value</ogc:PropertyName>
        <ogc:SortOrder>ASC</ogc:SortOrder>
      </ogc:SortProperty>
    </ogc:SortBy>
  </Query>
</GetRecords>
```

5.3 How To... Export a Release

The following query uses the current version history record to retrieve a GML dictionary of the dataset as a *.zip* file which can be saved to an appropriate location:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Retrieve the ReleaseObject related to the current Version History
    record. Then use the ReleaseObject identifier to obtain the actual
    compressed GML Dictionary file containing the EPSG Dataset via a
    getRepositoryItem request. -->
  <Query typeNames="rim:Association rim:RegistryObject">
    <ElementSetName typeNames="rim:RegistryObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- Find the correct association -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>rim:Association/@targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:version-history:EPSG::7.6.5</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>rim:Association/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:ReleaseFor</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- Now relate it to the ReleaseObject -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>rim:Association/@sourceObject</ogc:PropertyName>
            <ogc:PropertyName>rim:RegistryObject/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

5.4 How To... Retrieve An Entity in its Entirety

To retrieve an entity in its entirety starts with first obtaining the ebRIM record for the given CRS. This may be accomplished via any of the `GetRecords` requests provided in this guide. Secondly, the GML which represents the entity definition must be obtained. The GML document can be acquired by executing a `GetRepositoryItem` request using the identifier of the ebRIM record for the desired CRS. Thirdly, once the GML has been obtained, all of the references contained in it must be extracted and resolved.

The following is the ebRIM metadata for the (Projected) CRS having EPSG code "`2295`":

```
<?xml version="1.0" encoding="utf-8"?>
<wrs:ExtrinsicObject xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  id="urn:ogc:def:crs:EPSG::2295" lid="urn:ogc:def:crs:EPSG::2295"
  objectType="urn:ogc:def:ObjectType:GML:ProjectedCRS"
  status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved"
  mimeType="application/xml">
  <rim:Slot name="isDeprecated"
    slotType="urn:oasis:names:tc:ebxml-regrep:DataType:Boolean">
    <rim:ValueList>
      <rim:Value>false</rim:Value>
    </rim:ValueList>
  </rim:Slot>
  <rim:Slot name="AreaOfUse"
```

```
        slotType="urn:oasis:names:tc:ebxml-regrep:DataType:String">
        <rim:ValueList>
          <rim:Value>Canada - Nova Scotia - west of 63°W</rim:Value>
        </rim:ValueList>
      </rim:Slot>
      <rim:Slot name="EntityAlias"
        slotType="urn:oasis:names:tc:ebxml-regrep:DataType:String">
        <rim:ValueList>
          <rim:Value>[ATS77 / MTM NS zone 5]</rim:Value>
        </rim:ValueList>
      </rim:Slot>
      <rim:Slot name="DataSource"
        slotType="urn:oasis:names:tc:ebxml-regrep:DataType:String">
        <rim:ValueList>
          <rim:Value>OGP</rim:Value>
        </rim:ValueList>
      </rim:Slot>
      <rim:Slot name="SortKey"
        slotType="urn:oasis:names:tc:ebxml-regrep:DataType:String">
        <rim:ValueList>
          <rim:Value>1ats77 / mtm nova scotia zone 5</rim:Value>
        </rim:ValueList>
      </rim:Slot>
      <rim:Name>
        <rim:LocalizedString xml:lang="en" value="ATS77 / MTM Nova Scotia zone 5"/>
      </rim:Name>
      <rim:Description/>
      <rim:VersionInfo versionName="20110407T185230Z" comment=""/>
      <rim:ContentVersionInfo versionName="20110407T180930Z" comment=""/>
      <wrs:repositoryItemRef xlink:type="simple" xlink:href=
    "http://localhost/indicio/query?request=GetRepositoryItem&amp;service=CSW-
    ebRIM&amp;id=urn:ogc:def:crs:EPSG::2295"/>
    </wrs:ExtrinsicObject>
```

The identifier for this CRS is: `urn:ogc:def:crs:EPSG::2295`. Using this id in a `GetRepositoryItem` request yields the GML form of this entity:

```
<?xml version="1.0"?>
<ProjectedCRS xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  xmlns:epsg="urn:x-ogp:spec:schema-xsd:EPSG:1.0:dataset"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns="http://www.opengis.net/gml/3.2"
  gml:id="epsg-crs-2295">
  <metaDataProperty>
    <epsg:CommonMetaData>
      <epsg:type>projected</epsg:type>
      <epsg:alias alias="ATS77 / MTM NS zone 5" code="229"
        codeSpace="urn:ogc:def:naming-system:EPSG::7302"/>
      <epsg:informationSource>Geomatics Centre; Nova Scotia Ministry of Housing
      and Municipal Affairs.</epsg:informationSource>
      <epsg:revisionDate>1997-11-13</epsg:revisionDate>
      <epsg:show>true</epsg:show>
      <epsg:isDeprecated>false</epsg:isDeprecated>
    </epsg:CommonMetaData>
  </metaDataProperty>
  <identifier codeSpace="OGP">urn:ogc:def:crs:EPSG::2295</identifier>
  <name>ATS77 / MTM Nova Scotia zone 5</name>
  <remarks>In use from 1979. To be phased out in late 1990's.</remarks>
  <domainOfValidity xlink:href="urn:ogc:def:area:EPSG::1535"/>
  <scope>Large and medium scale topographic mapping and engineering
survey.</scope>
  <conversion xlink:href="urn:ogc:def:coordinateOperation:EPSG::17795"/>
  <baseGeodeticCRS xlink:href="urn:ogc:def:crs:EPSG::4122"/>
  <cartesianCS xlink:href="urn:ogc:def:cs:EPSG::4400"/>
  <style/>
</ProjectedCRS>
```

The following is a list of references contained in this entity:

an Area of Use:	urn:ogc:def:area:EPSG::1535

a Coordinate System:	urn:ogc:def:cs:EPSG::4400
a base CRS:	urn:ogc:def:crs:EPSG::4122
a Coordinate Conversion:	urn:ogc:def:coordinateOperation:EPSG::17795

Each of these related entities may be retrieved via the `GetRepositoryItem` request, as was done for the CRS itself. These entities may, in turn, contain further references which, again, will need to be retrieved.

The model for the EPSG Registry is described in Appendix B. It includes aggregations, for example a geodetic CRS has component datum which in turn has a component ellipsoid and this in turn has attributes which have units. When querying for the complete definition of a higher level entity it is necessary to iteratively mine down to the lower-level entities. The essential elements for the description of CRSs and coordinate transformations are described in Guidance Note 7-1.

5.5 How To... Find All CRS's with a Common Property

The following query filters on two criteria: an objectType whose "parent" is CRS (the base type CRS, thus effectively selecting all CRS sub-types), and whose Name contains the search term "Europe". This illustrates how one can query against an entire class of entity types. This technique is also applicable for other entity types which are grouped within a base-type: Coordinate Systems, Datums and Coordinate Operations. Refer to *Appendix A.3.1* "

Entity Types" for the base-type URNs for each group of entity types.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
  <!-- Return all CRS entities which contain the search term in their Name. -->
  <Query typeNames="wrs:ExtrinsicObject ClassificationNode_cnode_baseType">
    <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- The entity's objectType matches a ClassificationNode -->
          <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/wrs:ExtrinsicObject/@objectType</ogc:PropertyName>
            <ogc:PropertyName>/$cnode/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <!-- The objectType ClassificationNode has baseType as a parent -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$cnode/@parent</ogc:PropertyName>
            <ogc:PropertyName>/$baseType/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <!-- The baseType URN must be CRS -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$baseType/@id</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:ObjectType:GML:CRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <!-- Name property contains the search-term -->
          <ogc:PropertyIsLike wildCard="*" singleChar="?" escapeChar=".">
<ogc:PropertyName>/wrs:ExtrinsicObject/rim:Name/rim:LocalizedString/@value</ogc
:PropertyName>
            <ogc:Literal>*Europe*</ogc:Literal>
          </ogc:PropertyIsLike>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

5.6 How To... Find All Entities in a Named Region

The following query will return an unsorted list of all the entities that are related to an area with "scotia" in their area description:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Return all Entity objects related to an Area which matches the text
    search term specified. Note: SortBy clause included - see following example
    5.7. -->
  <Query typeNames="rim:Association rim:ClassificationNode
wrs:ExtrinsicObject_entity_extent">
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
<ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
```

```
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          <!-- A couple of predicates to ensure the Sort Key slot is
          represented fully -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/rim:Slot/@name</ogc:PropertyName>
              <ogc:Literal>SortKey</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsGreaterThan>
<ogc:PropertyName>/$entity/rim:Slot/rim:ValueList/rim:Value</ogc:PropertyName>
              <ogc:Literal>0</ogc:Literal>
            </ogc:PropertyIsGreaterThan>
            <!-- Search the Extent's textual description using wildcards -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
                 <ogc:PropertyName>/$extent/rim:Name/rim:LocalizedString/@value
                 </ogc:PropertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                 <?indicio-case-sensitive false?>
                 <ogc:PropertyName>/$extent/rim:Description/rim:LocalizedString
                 /@value</ogc:PropertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <!-- Find the classification node for the object type -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/rim:ClassificationNode/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
          <!-- Ensure that the ClassificationNode belongs to either the
          Geodetic object scheme, or the EPSG metadata scheme (this effectively
          screens out entity types which are not of interest) -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/EPSG-
Metadata*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
    <ogc:SortBy>
```

```
      <ogc:SortProperty>
        <ogc:PropertyName>/$entity/rim:Slot[@name=
'SortKey']/rim:ValueList/rim:Value[1]</ogc:PropertyName>
        <ogc:SortOrder>ASC</ogc:SortOrder>
      </ogc:SortProperty>
    </ogc:SortBy>
  </Query>
</GetRecords>
```

5.7 How To... Sort Result Sets (Using SortBy)

The `SortBy` command may be used to apply different criteria to sort the result set from a query. The basic `SortBy` clause looks like this:

```
      <ogc:SortBy xmlns:ogc="http://www.opengis.net/ogc">
        <ogc:SortProperty>
          <ogc:PropertyName>{property-name}</ogc:PropertyName>
          <ogc:SortOrder>{sort-order-keyword}</ogc:SortOrder>
        </ogc:SortProperty>
      </ogc:SortBy>
```

The `SortOrder` element may have the value '`ASC`', for ascending order, or '`DESC`', for descending order.
The `SortBy` clause may contain multiple `SortProperty` elements. A partial example with two sort properties looks like this:

```
<Query typeNames="wrs:ExtrinsicObject">
    ...
    <ogc:SortBy>
        <ogc:SortProperty>
            <ogc:PropertyName>/wrs:ExtrinsicObject/rim:Slot[@name =
'country']/rim:ValueList/rim:Value</ogc:PropertyName>
            <ogc:SortOrder>DESC</ogc:SortOrder>
        </ogc:SortProperty>
        <ogc:SortProperty>
            <ogc:PropertyName>/wrs:ExtrinsicObject
/rim:Name/rim:LocalizedString/@value</ogc:PropertyName>
            <ogc:SortOrder>ASC</ogc:SortOrder>
        </ogc:SortProperty>
    </ogc:SortBy>
    ...
</Query>
```

The following query will return a list of all the entities that are related to an area with the specified name and the result set will be sorted by the entity name:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Return all Entity objects related to an Area which matches the text
    search term specified. -->
  <Query typeNames="rim:Association rim:ClassificationNode
wrs:ExtrinsicObject_entity_extent">
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
<ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
```

```
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
                <!-- A couple of predicates to ensure the Sort Key slot is
                represented fully -->
            <ogc:PropertyIsGreaterThan>
              <ogc:PropertyName>/$entity/rim:Slot[@name =
'SortKey']/rim:ValueList/rim:Value</ogc:PropertyName>
              <ogc:Literal>0</ogc:Literal>
            </ogc:PropertyIsGreaterThan>
            <!-- Search the Extent's textual description using wildcards -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Name/rim:LocalizedString/@value</ogc:PropertyNam
e>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Description/rim:LocalizedString/@value</ogc:Prop
ertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <!-- Find the classification node for the object type -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/rim:ClassificationNode/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
          <!-- Ensure that the ClassificationNode belongs to either the
          Geodetic object scheme, or the EPSG metadata scheme (this effectively
          screens out entity types which are not of interest) -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/EPSG-
Metadata*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
    <ogc:SortBy>
      <ogc:SortProperty>
        <ogc:PropertyName>/$entity/rim:Slot[@name=
'SortKey']/rim:ValueList/rim:Value[1]</ogc:PropertyName>
        <ogc:SortOrder>ASC</ogc:SortOrder>
      </ogc:SortProperty>
    </ogc:SortBy>
```

```
  </Query>
</GetRecords>
```

5.8 How To... Filter Out Duplicate Results (Using DISTINCT)

The EPSG Registry uses the OGC Filter language as its primary query language. Queries expressed in OGC Filter syntax are ultimately converted into SQL for querying the underlying database. Some OGC Filter constructs can result in the selection of duplicate records which, in certain cases, may be undesirable. INdicio provides a processing instruction (PI) to reduce a result set containing duplicates to a set of distinct records based on the records identifier.

The `DISTINCT` processing instruction is expressed as follows:

```
<?indicio-distinct-values true?>
```

Note that the `DISTINCT` processing instruction **must** occur immediately following the start tag of the "`Query`" element. If it occurs anywhere else it is ignored. The following example shows the correct placement of the `DISTINCT` processing instruction:

```
<GetRecords ... >
    <Query typeNames="wrs:ExtrinsicObject">
        <?indicio-distinct-values true?>
        <ElementSetName typeNames="wrs:ExtrinsicObject">full</ElementSetName>
        <Constraint version="1.1.0">
            <ogc:Filter>

                ...lines omitted...

            </ogc:Filter>
        </Constraint>
    </Query>
</GetRecords>
```

The following query will return a list of all the entities that are related to an area with the specified name and will remove any duplicate entities from the result set:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
    <!-- Return all Entity objects related to an Area which matches the text
    search term specified. Note: No SortBy clause. Filter clause to also match
    search term in Alias property. Requires use of DISTINCT processing
    instruction to exclude duplicates. -->
  <Query typeNames="rim:Association rim:ClassificationNode
wrs:ExtrinsicObject_entity_extent">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
<ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
```

```
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- Search the Extent's textual description using wildcards -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Name/rim:LocalizedString@value</ogc:PropertyName
>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Description/rim:LocalizedString@value</ogc:Prope
rtyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
                <ogc:PropertyName>/$extent/rim:Slot[@name =
'SortKey']/rim:ValueList/rim:Value</ogc:PropertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <!-- Find the classification node for the object type -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/rim:ClassificationNode/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!-- Ensure that the ClassificationNode belongs to either the
            Geodetic object scheme, or the EPSG metadata scheme (this effectively
            screens out entity types which are not of interest) -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/EPSG-
Metadata*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

5.9 How To... Combine SortBy and DISTINCT Commands

The `SortBy` and `DISTINCT` commands may also be used together within a query.

> **WARNING:** for when using SortBy together with the DISTINCT processing instruction. The SortBy command supports multi-level sort criteria. However, when the DISTINCT processing instruction is used, SortBy, if also present, must only have a SINGLE sort criterion. This is due to the interaction between these two features.

The following query will return a list of all the entities that are related to an area with the specified name, sorted by name, and with duplicate entities removed from the result set:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
<!-- Return ANY Entity object which is related to an Area which matches the
text search term specified. Note: includes SortBy clause. Filter clause to also
match search term in Alias property. Requires use of DISTINCT processing
instruction to exclude duplicates. -->
  <Query typeNames="rim:Association rim:ClassificationNode
wrs:ExtrinsicObject_entity_extent">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <!-- The source of the association is the object we return -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
<ogc:PropertyName>/rim:Association/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/rim:Association/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
<ogc:PropertyName>/rim:Association/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          <!-- A couple of predicates to ensure the Sort Key slot is
          represented fully -->
            <ogc:PropertyIsGreaterThan>
              <ogc:PropertyName>/$entity/rim:Slot[@name =
'SortKey']/rim:ValueList/rim:Value</ogc:PropertyName>
              <ogc:Literal>0</ogc:Literal>
            </ogc:PropertyIsGreaterThan>
            <!-- Search the Extent's textual description using wildcards -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Name/rim:LocalizedString@value</ogc:PropertyName
>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
```

```
                <?indicio-case-sensitive false?>
<ogc:PropertyName>/$extent/rim:Description/rim:LocalizedString@value</ogc:Prope
rtyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
                <?indicio-case-sensitive false?>
                <ogc:PropertyName>/$extent/rim:Slot[@name =
'Alias']/rim:ValueList/rim:Value</ogc:PropertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- Constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <!-- Find the classification node for the object type -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/rim:ClassificationNode/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
          <!-- Ensure that the ClassificationNode belongs to either the
          Geodetic object scheme, or the EPSG metadata scheme (this effectively
          screens out entity types which are not of interest) -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="." singleChar="?">
<ogc:PropertyName>/rim:ClassificationNode/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/EPSG-
Metadata*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
    <ogc:SortBy>
      <ogc:SortProperty>
        <ogc:PropertyName>/$entity/rim:Slot[@name =
'SortKey']/rim:ValueList/rim:Value[1]</ogc:PropertyName>
        <ogc:SortOrder>ASC</ogc:SortOrder>
      </ogc:SortProperty>
    </ogc:SortBy>
  </Query>
</GetRecords>
```

5.10 How To... Find Valid Entities that replace a Deprecated Entity

Assuming that an entity which is not valid has been identified, the valid entity which replaced it may need to be located. This can be done by following what is called the "deprecation trail". In instances where no replacement entity was provided, the query would be expected to return an empty result set.

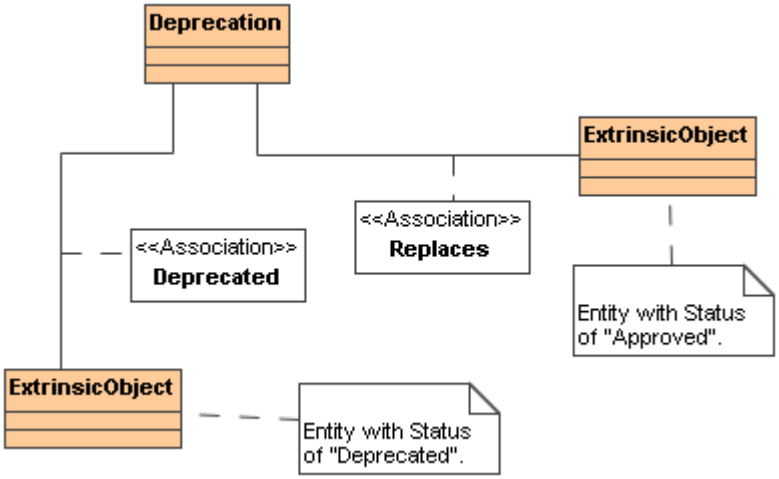

Figure 5: Associations Connecting a Deprecated Entity to its Valid Replacement Entity

The following is an example of an INVALID entity, a Geodetic CRS:

```
<?xml version="1.0"?>
<GeodeticCRS xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  xmlns:epsg="urn:x-ogp:spec:schema-xsd:EPSG:1.0:dataset"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns="http://www.opengis.net/gml/3.2"
  gml:id="epsg-crs-4902">
  <metaDataProperty>
    <epsg:CommonMetaData>
      <epsg:type>geographic 2D</epsg:type>
      <epsg:revisionDate>2004-01-06</epsg:revisionDate>
      <epsg:changes>
        <epsg:changeID xlink:href="urn:ogc:def:change-request:EPSG::2000.070"/>
      </epsg:changes>
      <epsg:show>true</epsg:show>
      <epsg:isDeprecated>true</epsg:isDeprecated>
    </epsg:CommonMetaData>
  </metaDataProperty>
  <identifier codeSpace="OGP">urn:ogc:def:crs:EPSG::4902</identifier>
  <name>NDG (Paris)</name>
  <domainOfValidity xlink:href="urn:ogc:def:area:EPSG::1369"/>
  <scope>Geodetic survey.</scope>
  <ellipsoidalCS xlink:href="urn:ogc:def:cs:EPSG::6403"/>
  <geodeticDatum xlink:href="urn:ogc:def:datum:EPSG::6902"/>
</GeodeticCRS>
```

Using the identifier from the invalid entity, create a query to find the replacement records which are related to it via its Deprecation trail:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
  xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10" resultType="results"
  outputFormat="application/xml; charset=UTF-8"
  service="WRS" version="1.0.1">
   <!-- Find any entities which have replaced an invalid (or deprecated)
   entity. -->
  <Query typeNames="wrs:ExtrinsicObject_entity_depn Association_repl_depd">
    <ElementSetName typeNames="entity">full</ElementSetName>
```

```
        <Constraint version="1.1.0">
          <ogc:Filter>
            <ogc:And>
              <!-- Constrain the objectType of the Deprecation record -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$depn/@objectType</ogc:PropertyName>
                <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:deprecation</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <!-- The Deprecated association links the Deprecation record to the
              invalid entity -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$depd/@sourceObject</ogc:PropertyName>
                <ogc:PropertyName>/$depn/@id</ogc:PropertyName>
              </ogc:PropertyIsEqualTo>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$depd/@associationType</ogc:PropertyName>
                <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Deprecated</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$depd/@targetObject</ogc:PropertyName>
                <ogc:Literal>urn:ogc:def:crs:EPSG::4902</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <!-- The Replaces association links the Deprecation record to the new
              entity -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$repl/@targetObject</ogc:PropertyName>
                <ogc:PropertyName>/$depn/@id</ogc:PropertyName>
              </ogc:PropertyIsEqualTo>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$repl/@associationType</ogc:PropertyName>
                <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:AssociationType:Replaces</ogc:Literal>
              </ogc:PropertyIsEqualTo>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$repl/@sourceObject</ogc:PropertyName>
                <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
              </ogc:PropertyIsEqualTo>
            </ogc:And>
          </ogc:Filter>
        </Constraint>
      </Query>
    </GetRecords>
```

If a replacement entity was specified when the initial entity was deprecated, the above query will return it. Verify that the `Status` property of the replacement entity is not "Deprecated". If the `Status` property of the replacement entity is, in fact, "Deprecated", this indicates that there is a multi-level Deprecation trail. In this situation, the query must be repeated, after substitution of the replacement entity's identifier for the "Deprecated" Association's targetObject property

5.11 How To... Retrieve the Area Polygons archive

The EPSG Area Polygons archive is stored as a repository item, under the registry object ID
"urn:x-ogp:def:archive:EPSG::extent-polygons".
This archive is a zip file that contains all of the current Area Polygon files for the registry. The archive can be downloaded from the client UI, or directly through the registry via the GetRepositoryItem request described below.

Retrieve the Area Polygons archive via an `HTTP GET request`:

```
http://www.epsg-registry.org/indicio/query
   ?request=GetRepositoryItem
   &id=urn:x-ogp:def:archive:EPSG::extent-polygons
```

Note: The above URL is spread across multiple lines to increase readability. When submitted through a browser, first remove all whitespace (including line feed characters).

5.12 How To… Retrieve a Polygon Metadata Record for a known Area of Use

As shown in Figure 4. "Object Relationships available for use in Spatial Queries", Area of Use records may have an associated Polygon record which will have a GML Polygon as its repository item. When you know the identifier of the Area record and want to retrieve its Polygon, use the query below, by first substituting the Area record identifier for the **${area-id}** property. An empty result set indicates that the given Area record does not have a related Polygon.

Note 1: The Polygon metadata record includes the Polygon geometry in a Slot named "http://purl.org/dc/terms/spatial" for indexing purposes. This may be sufficient for some uses. If, however, the Polygon geometry is required as a stand-alone GML document, it can be obtained once the Polygon metadata record has been retrieved. See clause 5.13 for further details.

Note 2: This request can also be viewed as the first step in obtaining the Polygon geometry for an Area of Use record. The two steps required, are: 1) Retrieve the Polygon metadata record (if one exists), in order to obtain its identifier, and 2) Retieve the Polygon geometry via the GetRepositorItem request (which requires the identifier obtained in step 1). See clause 5.13 for how to accomplish the second step.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords
    xmlns="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:wrs="http://www.opengis.net/cat/wrs/1.0"
    xmlns:ogc="http://www.opengis.net/ogc"
    outputFormat="application/xml; charset=UTF-8"
    service="WRS" version="1.0.1"
    startPosition="1" maxRecords="5"
    resultType="results">
  <Query typeNames="ExtrinsicObject_polygon Association_extentAssoc">
    <ElementSetName typeNames="polygon">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- substitute an Area identifier -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$extentAssoc/@targetObject</ogc:PropertyName>
            <ogc:Literal>${area-id}</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$extentAssoc/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:AssociationType:Extends</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$extentAssoc/@sourceObject</ogc:PropertyName>
            <ogc:PropertyName>$polygon/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$polygon/@objectType</ogc:PropertyName>
            <ogc:Literal>
              urn:x-ogp:def:ObjectType:EPSG:extent-polygon
            </ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
  </Query>
</GetRecords>
```

5.13 How To… Retrieve a Polygon Geometry given a Polygon Metadata Record Identifier

Note: This request represents the second step necessary to obtain Polygon geometry for a known Area of Use. See clause 5.12 for further details.

Assuming one has obtained a Polygon metadata record, its related repository item is a Polygon geometry. The GetRepositoryItem request is used to return the geometry. Substitute the Polygon metadata record identifier for the **${identifier}** string in this request:

```
http://www.epsg-registry.org/indicio/query
   ?request=GetRepositoryItem
   &id=${identifier}
```

Note: The above URL is spread across multiple lines to increase readability. When submitted through a browser, first remove all whitespace (including line feed characters).

ANNEX A – THE EPSG REGISTRY APPLICATION MODEL

This annex provides an overview of the information model of the EPSG Registry. The dataset is composed of geodetic entities which are maintained within the registry as GML 3.2.1 objects in XML format. The following sections provide brief descriptions and diagrams for:

- the GML 3.2.1 model;
- the Registry Model for the ebRIM 3.0 objects which are generated from the GML objects.

A.1 GML Model and Application Schema

The GML application schemas for the EPSG Registry conform GML 3.2.1. The scope of the data content of the EPSG dataset is limited to GML CRS entities. The schema files for the EPSG Registry are available from http://www.epsg.org/RegistrySchemas/main.asp. The schema files in the zip file archive include the following:

- Folder “epsg”: Contains the EPSG Registry Schema.
- Folder "gml": Contains the GML 3.2.1 Schema set; the root document of the GML Schema is file "gml.xsd".
- Folder "w3c": Contains the imported W3C schemas (XLink 1.1, etc).
- Folder "iso": Contains the imported GMD schema and related schemas (see ISO/TS 19139).

The namespace for the EPSG Registry GML application schema is:

```
urn:x-ogp:spec:schema-xsd:EPSG:1.0:dataset
```

Prior to August 2012 the EPSG dataset used the draft specification of GML, v ersion3.2.0. Note that there have been a number of changes between GML 3.2.0 and GML 3.2.1, that affect the EPSG Dataset definitions. These are summarised in Annex E.

A.2 Geodetic Model

The two diagrams which follow describe, in a simplified manner, the geodetic model for the EPSG Registry. They are taken from "*ISO 19111:2007 Geographic information – Spatial referencing by coordinates*" as implemented in GML through "*ISO 19136 Geographic information – Geographic mark-up language (GML)*".

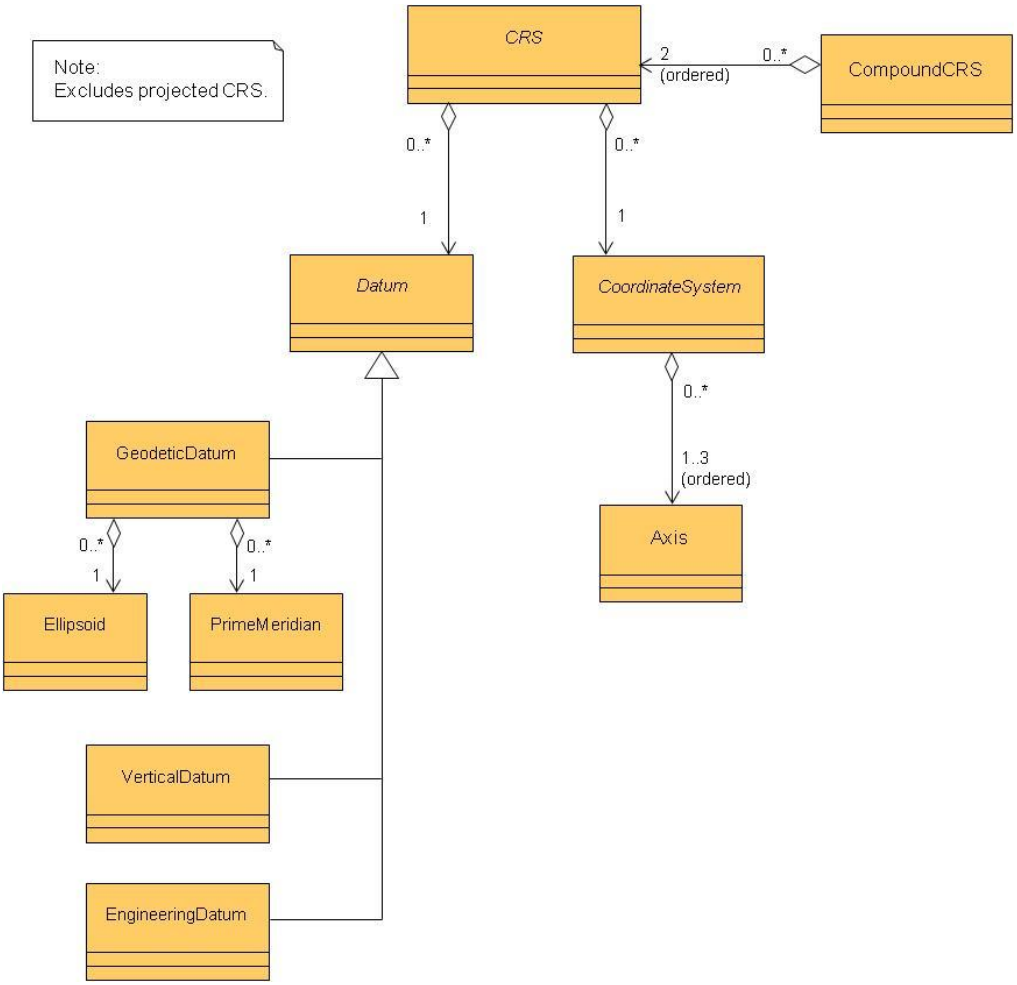

Figure A–1: Coordinate Reference System

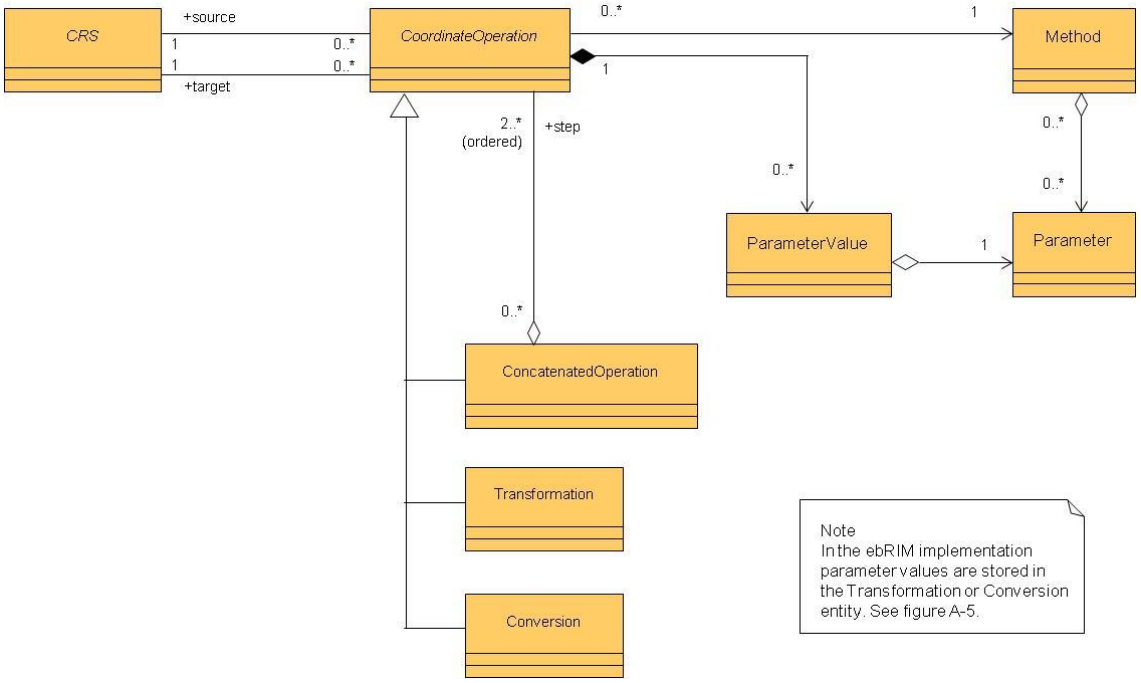

Figure A–2: Coordinate Operations

Note: Concatenated Operations cannot contain other Concatenated Operations, only Transformations and/or Conversions. For reasons of simplicity that constraint has not been expressed in the above diagram.

A.3 ebRIM Model

The EPSG Dataset, once converted to GML form, is used to generate metadata records (or registry objects), in ebRIM format, for insertion into the EPSG Registry. These metadata records serve as the target for queries used to find geodetic entity content held by the registry.

The object type for a given ebRIM ExtrinsicObject and the association types which may be used to relate two ExtrinsicObjects are shown from the diagrams presented in figures A-3 through A-6 below. The short name for the object or association is used in the diagram and matches a URN in the tables which follow the diagrams. The URN's in the tables should be used when constructing OGC Filter queries for the EPSG Registry.

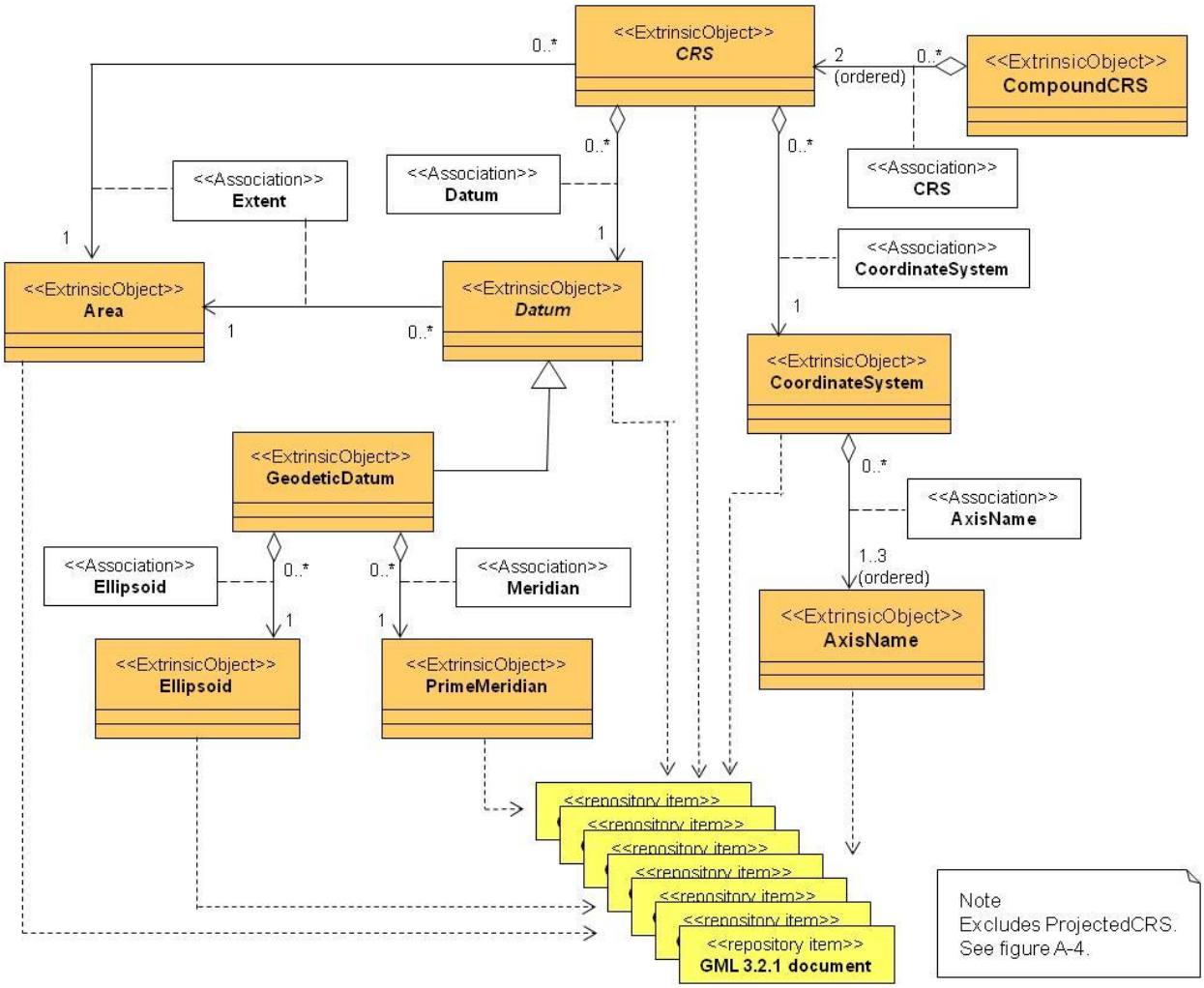

Figure A–3: EPSG Registry Model (ebRIM): CRS and Related Entities

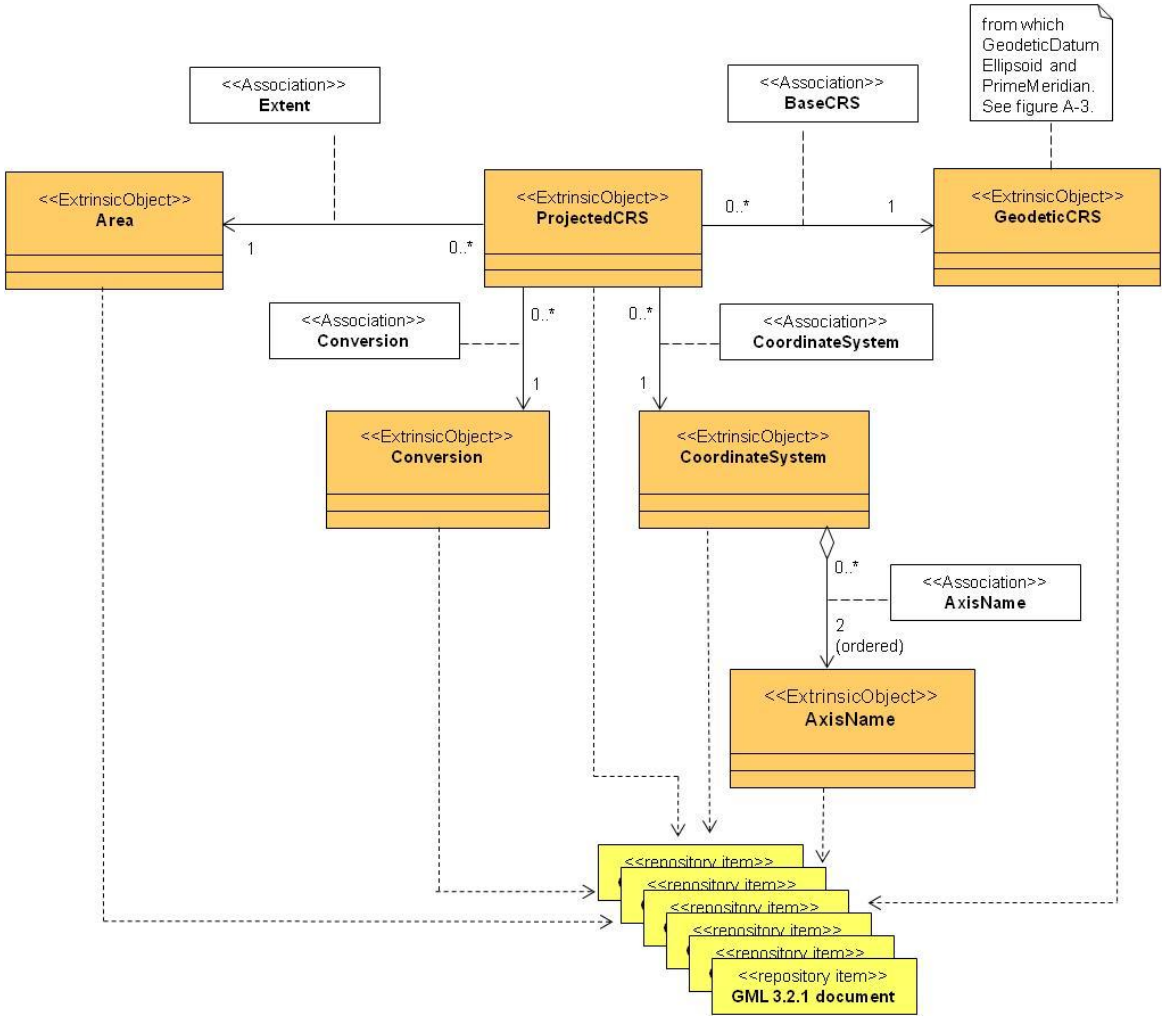

Figure A–4: EPSG Registry Model (ebRIM): ProjectedCRS

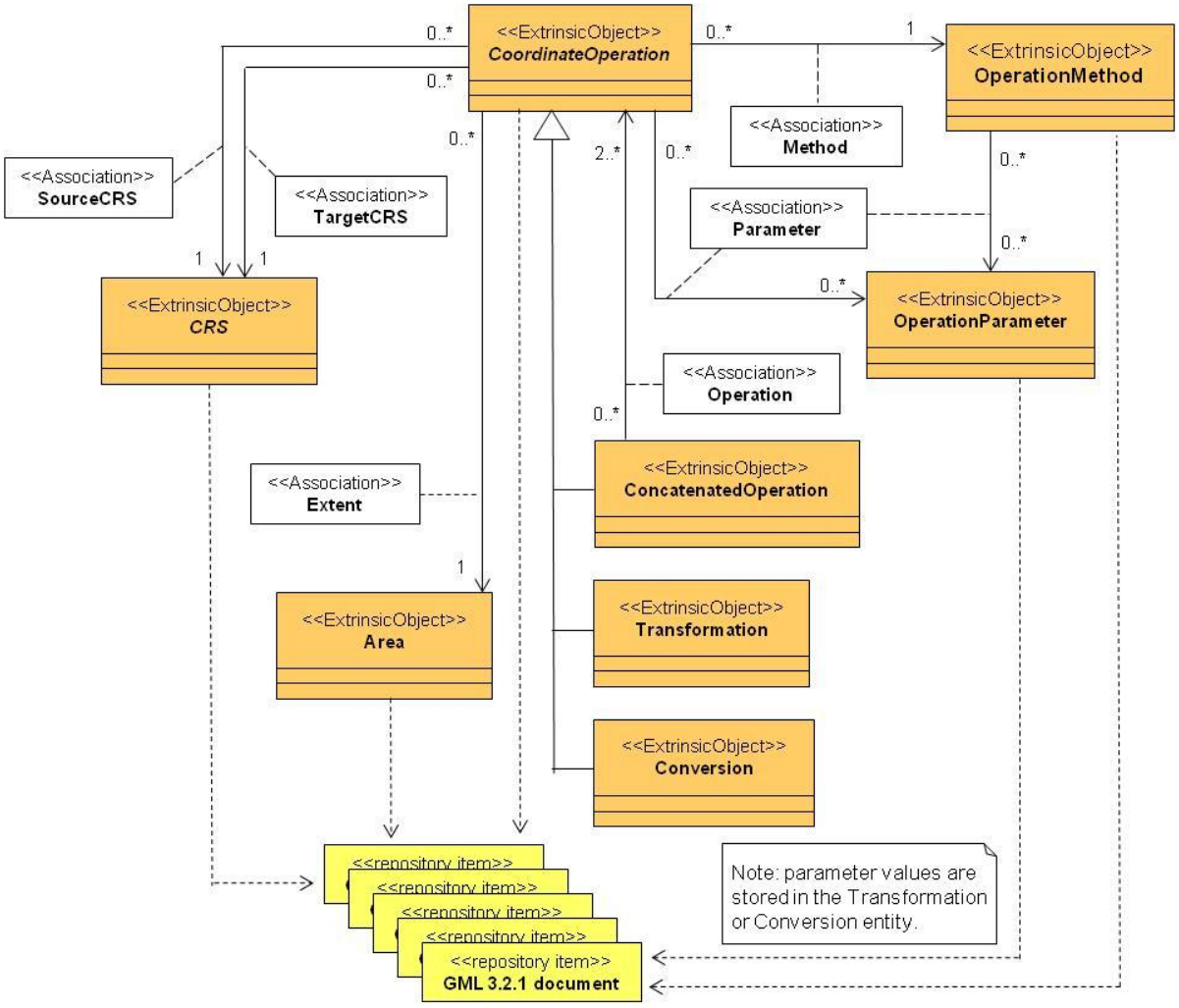

Figure A–5: EPSG Registry Model (ebRIM): Coordinate Operations

In the EPSG Registry, for performance reasons area description and bounding box data is stored separately from the polygon data describing the area boundary.

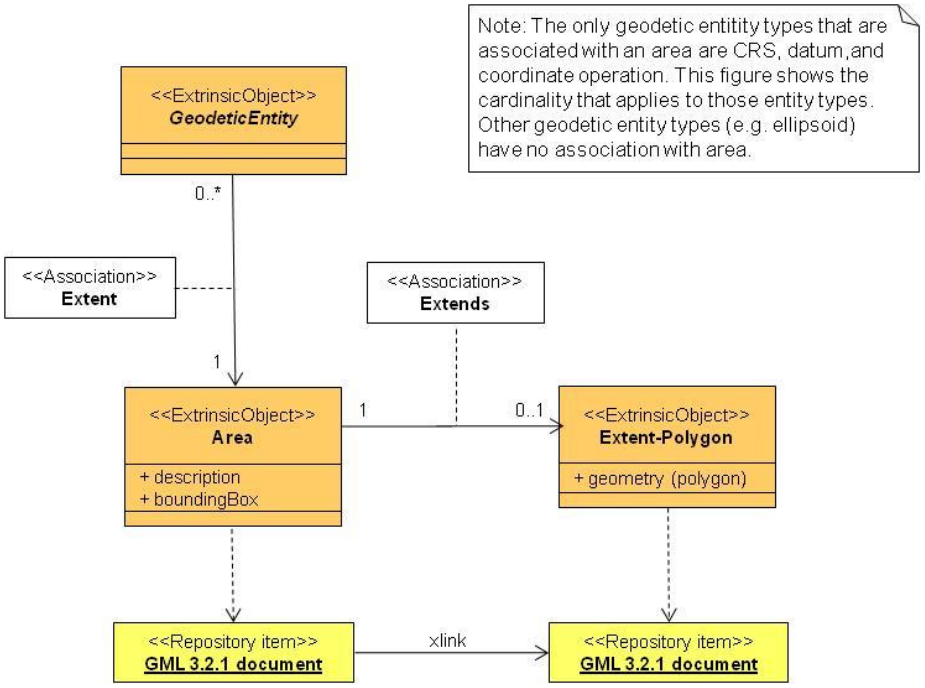

Figure A–6: Area

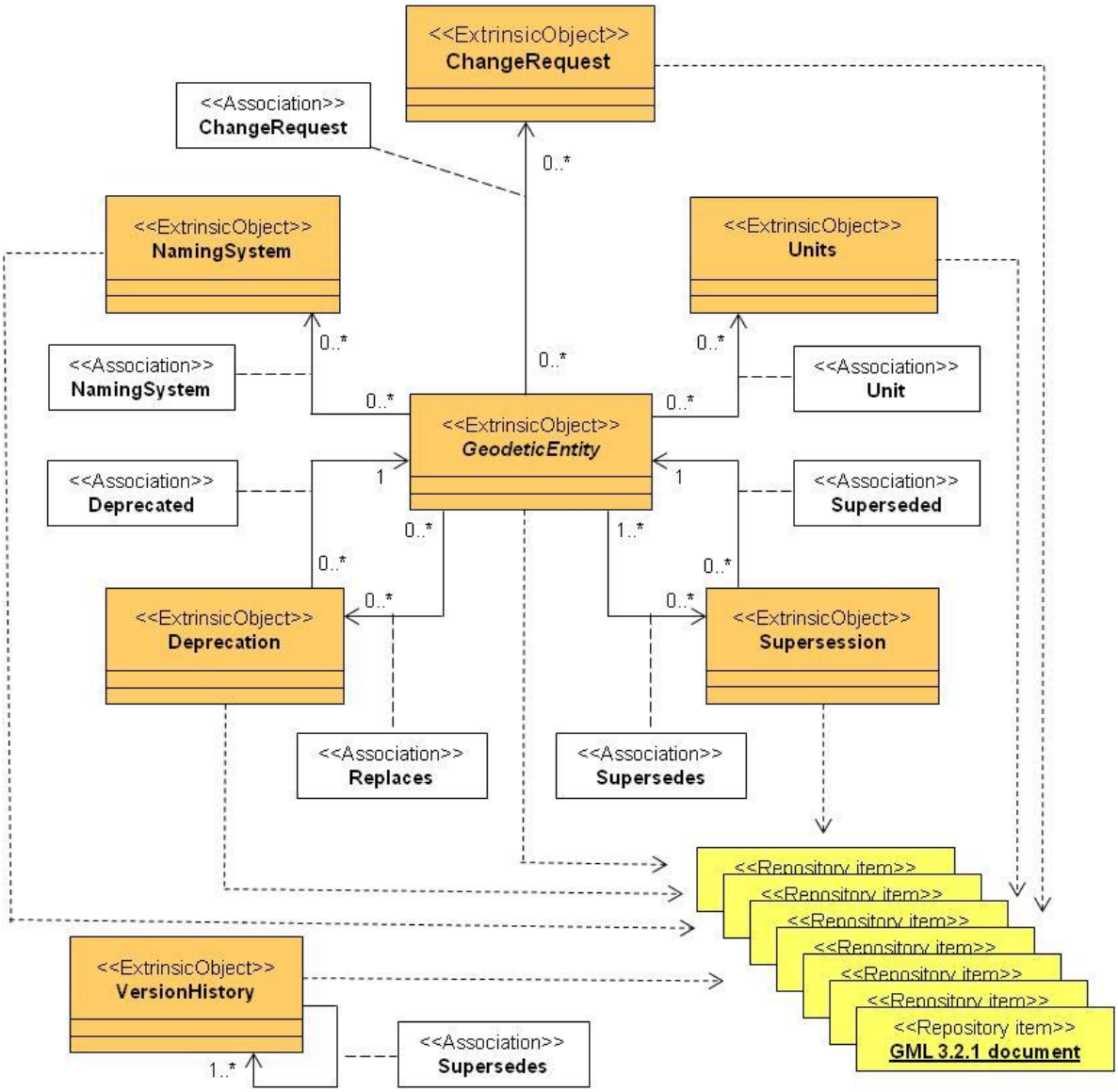

Figure A–7: EPSG Registry Model (ebRIM): Additional EPSG Metadata Objects

A.3.1 Entity Types

There are four entity types for which there exists an abstract base-type: Datum, CRS, Coordinate System and Coordinate Operation. Concrete entity instances in the dataset always reference a sub-type, but these base-type definitions can be used for querying at the type level.

A.3.1.1 CRS

The CRS base-type URN is: `urn:ogc:def:ObjectType:GML:CRS`

```
urn:ogc:def:ObjectType:GML:CompoundCRS
urn:ogc:def:ObjectType:GML:DerivedCRS
urn:ogc:def:ObjectType:GML:EngineeringCRS
urn:ogc:def:ObjectType:GML:GeodeticCRS
urn:ogc:def:ObjectType:GML:ImageCRS
urn:ogc:def:ObjectType:GML:ProjectedCRS
urn:ogc:def:ObjectType:GML:TemporalCRS
urn:ogc:def:ObjectType:GML:VerticalCRS
```

A.3.1.2 Coordinate System

The Coordinate System base-type URN is: `urn:ogc:def:ObjectType:GML:CoordinateSystem`

```
urn:ogc:def:ObjectType:GML:AffineCS
urn:ogc:def:ObjectType:GML:CartesianCS
urn:ogc:def:ObjectType:GML:CylindricalCS
urn:ogc:def:ObjectType:GML:EllipsoidalCS
urn:ogc:def:ObjectType:GML:LinearCS
urn:ogc:def:ObjectType:GML:PolarCS
urn:ogc:def:ObjectType:GML:SphericalCS
urn:ogc:def:ObjectType:GML:TimeCS
urn:ogc:def:ObjectType:GML:UserDefinedCS
urn:ogc:def:ObjectType:GML:VerticalCS
```

A.3.1.3 Datum

The Datum base-type URN is: `urn:ogc:def:ObjectType:GML:Datum`

```
urn:ogc:def:ObjectType:GML:GeodeticDatum
urn:ogc:def:ObjectType:GML:EngineeringDatum
urn:ogc:def:ObjectType:GML:ImageDatum
urn:ogc:def:ObjectType:GML:TemporalDatum
urn:ogc:def:ObjectType:GML:VerticalDatum
```

A.3.1.4 Coordinate Operation

The Coordinate Operation base-type URN is: `urn:ogc:def:ObjectType:GML:CoordinateOperation`

```
urn:ogc:def:ObjectType:GML:CoordinateConversion
urn:ogc:def:ObjectType:GML:CoordinateTransformation
urn:ogc:def:ObjectType:GML:ConcatenatedCoordinateOperation
```

A.3.1.5 Other Object Types

```
urn:ogc:def:ObjectType:GML:Ellipsoid
urn:ogc:def:ObjectType:GML:PrimeMeridian
urn:ogc:def:ObjectType:GML:OperationMethod
urn:ogc:def:ObjectType:GML:OperationParameter
urn:ogc:def:ObjectType:GML:Unit

urn:x-ogp:def:ObjectType:EPSG:area
urn:x-ogp:def:ObjectType:EPSG:axis-name
urn:x-ogp:def:ObjectType:EPSG:change-request
urn:x-ogp:def:ObjectType:EPSG:deprecation
urn:x-ogp:def:ObjectType:EPSG:supersession
urn:x-ogp:def:ObjectType:EPSG:naming-system
urn:x-ogp:def:ObjectType:EPSG:version-history
urn:x-ogp:def:ObjectType:EPSG:Release
urn:x-ogp:def:ObjectType:EPSG:extent-polygon
```

A.3.2 Association Types

The association types which may be used to relate two ExtrinsicObjects can be inferred from the diagrams in figures A-3 through A-6 above. The association's short name is used in the diagrams and matches a URN in the following tables. The URN's in these tables should be used when constructing OGC Filter queries for the EPSG Registry.

A.3.2.1 Association Types

```
urn:ogc:def:AssociationType:GML:BaseCRS
urn:ogc:def:AssociationType:GML:CRS
urn:ogc:def:AssociationType:GML:CoordinateSystem
urn:ogc:def:AssociationType:GML:Datum
urn:ogc:def:AssociationType:GML:Conversion
urn:ogc:def:AssociationType:GML:Transformation
urn:ogc:def:AssociationType:GML:Operation
urn:ogc:def:AssociationType:GML:SourceCRS
urn:ogc:def:AssociationType:GML:TargetCRS
urn:ogc:def:AssociationType:GML:Meridian
urn:ogc:def:AssociationType:GML:Ellipsoid
urn:ogc:def:AssociationType:GML:Method
urn:ogc:def:AssociationType:GML:Parameter
urn:ogc:def:AssociationType:GML:AxisName
urn:ogc:def:AssociationType:GML:Unit

urn:x-ogp:def:AssociationType:EPSG:Superseded
urn:x-ogp:def:AssociationType:EPSG:Deprecated
urn:x-ogp:def:AssociationType:EPSG:ChangeRequest
urn:x-ogp:def:AssociationType:EPSG:Extent
urn:x-ogp:def:AssociationType:EPSG:NamingSystem
urn:x-ogp:def:AssociationType:EPSG:ReleaseFor
urn:oasis:names:tc:ebxml-regrep:AssociationType:Extends
```

ANNEX B – SLOTS USED IN THE EPSG REGISTRY

Slot Names and Descriptions

The following is a description of each slot:

Slot Name	Description
AreaOfUse	This slot contains the code for the area of use associated with this entity
CodesAffected	This slot contains the "codes affected" field for the change request entity
DataSource	This slot contains the data source for the entity
DateClosed	This slot contains the closed date of the change request. It also distinguishes between an open change request and a closed change request.
EntityAlias	This slot contains the entity aliases for the entity
EntitySubType	This slot is used for geodetic CRSs to distinguish between geographic 2d, geographic 3d and geocentric
EntityTypeAffected	This slot contains the "entity types affected" field for the change request entity
Envelope	This slot contains a GML envelope to do geospatial queries against
isDeprecated	This slot contains the deprecation state for the entity
Request	This slot contains the "request" field for the change request entity
SortKey	This slot is used for sorting in the UI
Status	This slot is used for tracking the state of an open change request
VersionDate	This slot is used for tracking Dataset version release dates
Format	This slot is used to determine what type of release is referred to. It can be SQL or GML.
http://purl.org/dc/terms/spatial	This slot name indicates a GML geometry. Currently used on Extent Polygon metadata record for slot containing the Polygon geometry.

Entities Containing Slots

The following table shows which objects contain which slots:

	AreaOfUse	CodesAffected	DataSource	DateClosed	EntityAlias	EntitySubType	EntityTypesAffected	http://www.opengis.net/gml/Envelope	isDeprecated	Request	SortKey	Status	VersionDate	Format
urn:x-ogc:def:ObjectType:GML:AffineCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:CartesianCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:CompoundCRS	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:ConcatenatedCoordinateOperation	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:CoordinateConversion	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:CoordinateTransformation	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:CylindricalCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:Ellipsoid			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:EllipsoidalCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:EngineeringCRS	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:EngineeringDatum	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:GeodeticCRS	✓		✓		✓	✓			✓		✓			
urn:x-ogc:def:ObjectType:GML:GeodeticDatum	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:LinearCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:OperationMethod			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:OperationParameter			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:PolarCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:PrimeMeridian			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:ProjectedCRS	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:SphericalCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:Unit			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:VerticalCRS	✓		✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:VerticalCS			✓		✓				✓		✓			
urn:x-ogc:def:ObjectType:GML:VerticalDatum	✓		✓		✓				✓		✓			
urn:x-ogp:def:ObjectType:EPSG:area			✓		✓			✓	✓		✓			
urn:x-ogp:def:ObjectType:EPSG:axis-name			✓		✓						✓			
urn:x-ogp:def:ObjectType:EPSG:change-request		✓	✓	✓	✓		✓		✓	✓	✓	✓		
urn:x-ogp:def:ObjectType:EPSG:deprecation			✓		✓						✓			
urn:x-ogp:def:ObjectType:EPSG:naming-system			✓						✓		✓			
urn:x-ogp:def:ObjectType:EPSG:supersession			✓		✓						✓			
urn:x-ogp:def:ObjectType:EPSG:version-history			✓		✓				✓		✓		✓	
urn:x-ogp:def:ObjectType:EPSG:Release														✓

ANNEX C – STANDARDS AND SPECIFICATIONS

This appendix lists the various specifications, standards, and protocols that are implemented in whole or in part by the registry software, or are used in the definition of the dataset entities or the structure of queries.

ebRIM and Web Services

The following standards and specifications are implemented by the registry software:

Specification	Version	Document	Short Name
OpenGIS® Catalogue Services - ebRIM (ISO/TS 15000-3) profile of CSW	1.0.0	OGC document 05-025r3	WRS *
(CSW ebRIM)			
OGC Catalogue Services Specification	2.0.1	OGC document 04-017r3	CSW
OASIS ebXML Registry Information Model	3.0	OASIS regrep-rim-3.0-OS	ebRIM
OGC Web Services Common Specification	1.0.0	OGC document 05-008	OWS Common
OpenGIS® Filter Encoding Implementation Specification	1.1.0	OGC document 04-095	OGC Filter
OpenGIS® Geography Markup Language (GML) Encoding Specification	3.1.1	OGC document 03-105r1	GML
Definition identifier URNs in OGC namespace	1.2	OGC document 07-092r2	URNs

*The WRS application profile is based on Clause 10 of the CSW specification, which describes HTTP protocol binding for catalogue services.

For additional information about various standards:

- The OpenGis Consortium Website address is http://www.opengeospatial.org/
- The OASIS Website address is http://www.oasis-open.org/

XML and HTTP

The following standards and recommendations define the protocols and data structure for requests:

Specification	Version	Document	Short Name
Network Working Group Hypertext Transfer Protocol -- HTTP/1.1	N/A	RFC 2616	HTTP
World Wide Web Consortium (W3C) Extensible Markup Language	1.0 (Fourth Edition)	N/A	XML
World Wide Web Consortium (W3C) XML Linking Language (XLink)	1.1	http://www.w3.org/TR/xlink11/	Xlink

For more information about HTTP basic authentication:

- RFC 2616 (*Hypertext Transfer Protocol – HTTP/1.1*) http://tools.ietf.org/html/rfc2616
- RFC 2617 (*HTTP Authentication: Basic and Digest Access Authentication*) http://tools.ietf.org/html/rfc2617

Dataset Entities

The entities in the EPSG Geodetic Parameter dataset, held as RepositoryItems, conform to the following specifications:

Specification	Version	Document	Short Name
OpenGIS® Geography Markup Language (GML) Encoding Specification	3.2.1	OGC document 07-036	GML
which is equivalent to ISO Geographic information – Geography Markup Language (GML)	2007	ISO 19136:2007	19136
OGC Abstract Specification Topic 2, Spatial referencing by coordinates	r4	OGC document 08-015r2	Topic 2
which is equivalent to ISO Geographic information – Spatial referencing by coordinates	2007	ISO 19111:2007	19111

ANNEX D – SUMMARY OF AUGUST 2011 CHANGES TO REGISTRY SERVICE MESSAGE FORMAT

The EPSG web interface software (INdicio) was upgraded in August 2011. As a consequence of this upgrade the registry service interface methods have not changed, however their message content has. The following list of changes summarises the key differences in message content between the previous version (1.2.x) and the current version (2.2.x) of the application EPSG Registry.

- **Namespaces** –
  The following XML namespaces have changed:
  - **CSW** – the CSW namespace has been changed from "http://www.opengis.net/cat/csw" to "http://www.opengis.net/cat/csw/2.0.2".
  - **WRS** – the WRS namespace has been changed from "http://www.opengis.net/cat/wrs" to "http://www.opengis.net/cat/wrs/1.0".

- **ExtrinsicObject**
  The namespace of the ExtrinsicObject has changed. It is now in the **WRS** namespace, whereas previously it was in the **RIM** namespace. Therefore, instead of rim:ExtrinsicObject, one must use wrs:ExtrinsicObject.

- **OGC Filter syntax** – used in GetRecords requests.
  - **Slot names** – Instead of having to use an ogc:And operator to filter on a Slot name and value using two separate predicates, a single predicate can be used to select a Slot value by its name attribute, e.g. an ogc:PropertyName to filter on Slots with the name "EntitySubType" would appear as follows:
    ```
    /wrs:ExtrinsicObject/Slot[@name = 'EntitySubType']/ValueList/Value
    ```
  - **Aliases** – An alias is declared by appending the alias name to its objectType name in the typeNames property, using an underscore character. Additionally, an alias specified in an ElementSetName no longer uses a prefixed dollar sign, e.g. the following shows two Association aliases (note the aliases "assoc1" and "assoc2", referenced as "$assoc1" and "$assoc2" within any subsequent filter predicates):
    ```
    <csw:Query typeNames="rim:Association_assoc1_assoc2">
    ```

- `GET` requests case sensitivity
  The software is now rigorous in checking case sensitivity of GET requests. Therefore, instead of `getRepositoryItem`, one must use `GetRepositoryItem`.

ANNEX E – SUMMARY OF AUGUST 2012 CHANGES TO REGISTRY

Several factors caused changes to be made to the EPSG Registry schema in August 2012:

1. A change in OGC policy to adopt W3C Xlink v1.1 schema (see http://www.opengeospatial.org/blog/1597).
2. An update of the registry schema replacing GML v3.2.0 with v3.2.1. This causes change to the GML namespace URI.
3. The EPSG v0.1 application schema had an Area of Use entity definition which did not conform fully with GML. The primary issue was that the Area of Use definition **contained** the spatial extent (gmd:EX_Extent) definition instead of being **derived** from it – and therefore was not a valid substitute for the gmd:EX_Extent within GML documents. This prevented the EPSG GML representation from being processed fully by schema-aware XML processors.
4. Changes to accommodate the addition of area polygons.
5. Addition of wrapping elements – epsg:ExtentDefinition and epsg:extent – for Area of Use definitions within an exported EPSG Dictionary file. These elements conform to GML's Object-Property rule and enable Area of Use definitions to reside within a gml:dictionaryEntry element in a schema valid manner. The duplicated epsg:ExtentDefinition/gml:identifier element is a mandatory GML property.

Collectively these changes impact API service message syntax. In summary the changes required are :

Prior to August 2012
urn:x-ogp:spec:schema-xsd:EPSG:0.1:dataset

http://schemas.opengis.net/xlink/1.0.0/xlinks.xsd
xlink:simpleLink
http://www.opengis.net/gml

<gml:CoordinateSystemAxis gml:uom="...">
<isSphere>Sphere</isSphere>

```
<gml:dictionaryEntry>
  <epsg:AreaOfUse>
      …
  </epsg:AreaOfUse>
</gml:dictionaryEntry>
```

From August 2012
urn:x-ogp:spec:schema-xsd:EPSG:1.0:dataset

http://www.w3.org/1999/xlink.xsd
xlink: simpleAttrs
http://www.opengis.net/gml/3.2

<gml:CoordinateSystemAxis uom="...">
<isSphere>true</isSphere>

```
<gml:dictionaryEntry>
  <epsg:ExtentDefinition
      gml:id="epsg-area-1024">
    <gml:identifier codeSpace="OGP">
      urn:ogc:def:area:EPSG::1024
    </gml:identifier>
    <epsg:extent>
      <epsg:AreaOfUse>
          …
      </epsg:AreaOfUse>
    </epsg:extent>
  </epsg:ExtentDefinition>
</gml:dictionaryEntry>
```