Surveying and Positioning
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.

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 3.2.0. See Annex C for a reference to the defining documentation.

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

The following are definitions for some key terms that are used in this documentation:

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".

Term	Definition
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: "Figure 2: Registry components 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 ebRIM

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. 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:

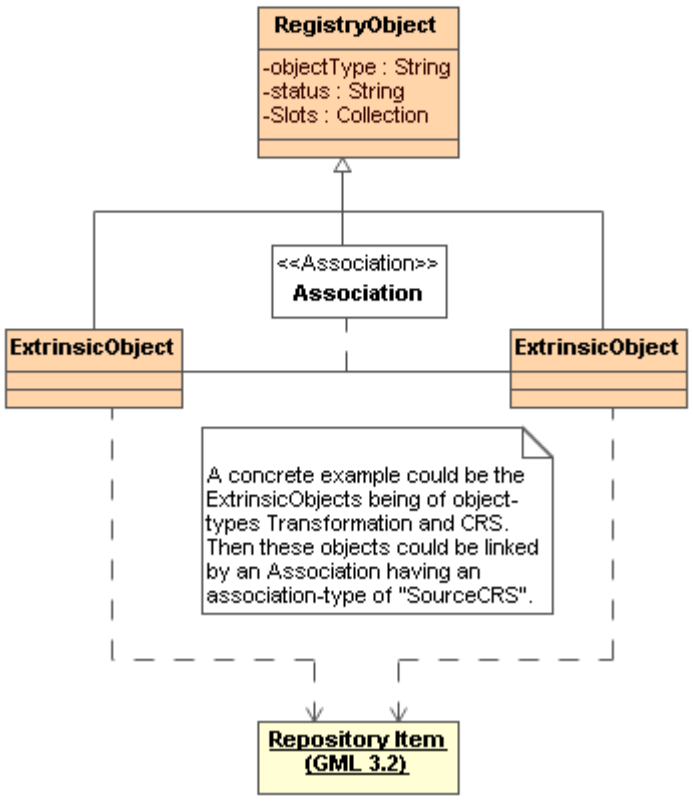

RegistryObject
-objectType : String
-status : String
-Slots : Collection
<<Association>>
Association
ExtrinsicObject
ExtrinsicObject
A concrete example could be the ExtrinsicObjects being of object-types Transformation and CRS. Then these objects could be linked by an Association having an association-type of "SourceCRS".
Repository Item
(GML 3.2)

Figure 1: ebRIM objects implemented in the EPSG Registry

2.2 Registry Components

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.

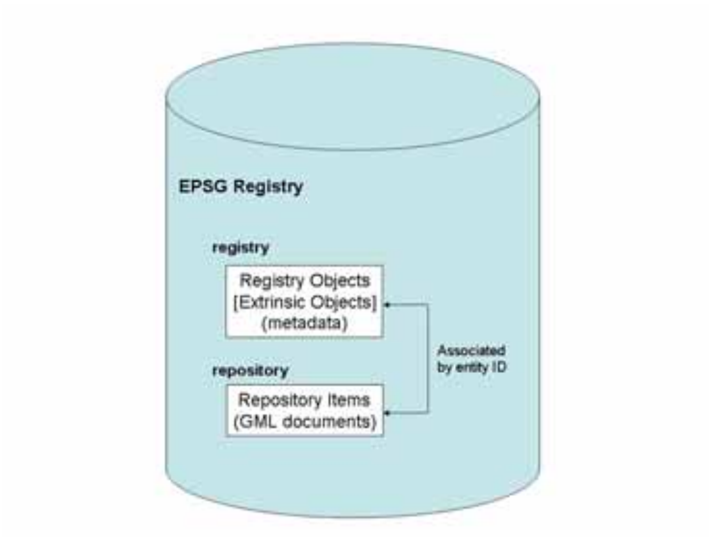


Figure 2: Registry components

2.2.1 RegistryObjects

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

2.2.2 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.

2.2.3 RepositoryItems

A geodetic entity in the dataset is defined as data in a GML document that is 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 always GML 3.2 documents.

.

2.2.4 Association

A RegistryObject may be connected to other RegistryObjects through an Association, which defines a many-to-many relationship among 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 id of one of the ClassificationNodes in the Association type ClassificationScheme.

2.2.5 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.

2.2.6 Slots

Slots provide a means to add arbitrary 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:

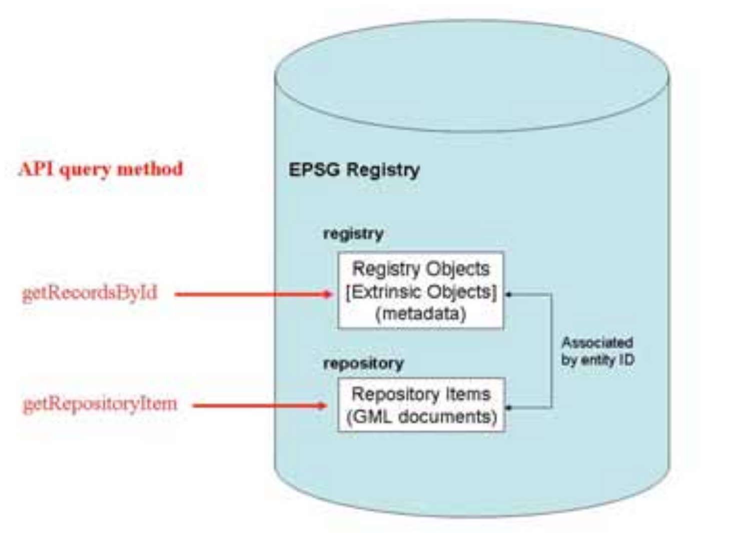


Figure 3: Registry query mechanisms

4.1.1 GetRecordById

The getRecords method is a general query mechanism. To execute any getRecords query, perform an `HTTP POST` request on the following URL:

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

with the desired query attached in the message body. For the EPSG Registry `:<port>` is not required.

4.1.1.1 GetRecordById

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

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

where `[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"
    xmlns:wrs="http://www.opengis.net/cat/wrs"
    xmlns:ogc="http://www.opengis.net/ogc"
    startPosition="1" maxRecords="10"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
  <!--
    Query on Entity by matching a search-term to its Name or Description
property.
  -->
  <Query typeNames="ExtrinsicObject=entity ClassificationNode=cnode_3,cnode_5">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="$entity">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>/$entity/rim:Name/rim:LocalizedString@value</ogc:PropertyName
>
              <ogc:Literal>*paris*</ogc:Literal>
            </ogc:PropertyIsLike>
            <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
              <?indicio-case-sensitive false?>

<ogc:PropertyName>/$entity/rim:Description/rim:LocalizedString@value</ogc:Prope
rtyName>
              <ogc:Literal>*paris*</ogc:Literal>
            </ogc:PropertyIsLike>
          </ogc:Or>
          <!--
            restrict ObjectType of Entity to an EPSG entity type;
            this excludes EPSG Metadata record types such as:  Change Request,
Deprecation etc.
          -->
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$cnode_3/@parent</ogc:PropertyName>
              <ogc:Literal>urn:ogc:def:ObjectType:GML:Geodetic-
Entity</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Or>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
                <ogc:PropertyName>/$cnode_3/@id</ogc:PropertyName>
              </ogc:PropertyIsEqualTo>
              <ogc:And>
                <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
                  <ogc:PropertyName>/$cnode_5/@id</ogc:PropertyName>
                </ogc:PropertyIsEqualTo>
                <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>/$cnode_5/@parent</ogc:PropertyName>
                  <ogc:PropertyName>/$cnode_3/@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 deprecated (invalid) and valid 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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc" startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8" resultType="results">
  <!--
    Query on Entity by entity-type.
  -->
  <Query typeNames="ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- specify the entity type via a URN from the ObjectType
ClassificationScheme -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$entity/@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>/$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>
```

4.2.2 Find Entity by 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 to add criteria for the entity type or name will help to narrow the search results.
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"
    xmlns:wrs="http://www.opengis.net/cat/wrs"
    xmlns:ogc="http://www.opengis.net/ogc"
    startPosition="1" maxRecords="10"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
  <Query typeNames="ExtrinsicObject=r">
    <ElementSetName typeNames="$r">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="/" singleChar="?">
            <ogc:PropertyName>/$r/@id</ogc:PropertyName>
            <ogc:Literal>*:1250</ogc:Literal>
          </ogc:PropertyIsLike>
          <!-- typically these metadata records (non-entities) would be
excluded from the reslt set -->
          <ogc:Not>
            <ogc:Or>
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$r/@objectType</ogc:PropertyName>
                <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:deprecation</ogc:Literal>
```

```
                </ogc:PropertyIsEqualTo>
                <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>/$r/@objectType</ogc:PropertyName>
                  <ogc:Literal>urn:x-
ogp:def:ObjectType:EPSG:supersession</ogc:Literal>
                </ogc:PropertyIsEqualTo>
                <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>/$r/@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

This type of query will find and return the single entity that matches the criteria specified because the entity code is unique within a RegistryObject type.
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"
    xmlns:wrs="http://www.opengis.net/cat/wrs"
    xmlns:ogc="http://www.opengis.net/ogc"
    startPosition="1" maxRecords="10"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
  <!--
    Query on Entity Code and by ObjectType.
    By also constraining the results by ObjectType we eliminate entities which
are not of interest.
  -->
  <Query typeNames="ExtrinsicObject=r">
    <ElementSetName typeNames="$r">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="/" singleChar="?">
            <ogc:PropertyName>/$r/@id</ogc:PropertyName>
            <ogc:Literal>*:1250</ogc:Literal>
          </ogc:PropertyIsLike>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$r/@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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc" startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8" resultType="results">
  <!--
    Query on Entity by matching a search-term to its Name or Alias property.
  -->
  <Query typeNames="ExtrinsicObject=entity entity/rim:Slot=aliasSlot_1">
    <?indicio-distinct-values true?>
```

```
        <ElementSetName typeNames="$entity">full</ElementSetName>
        <Constraint version="1.1.0">
          <ogc:Filter>
            <ogc:And>
              <!-- specify the entity type via a URN from the ObjectType
    ClassificationScheme -->
              <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>/$entity/@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>/$entity/Name/LocalizedString/@value</ogc:PropertyName>
                  <ogc:Literal>*scotia*</ogc:Literal>
                </ogc:PropertyIsLike>
                <ogc:And>
                  <ogc:PropertyIsEqualTo>
                    <ogc:PropertyName>$entity/$aliasSlot_1/name</ogc:PropertyName>
                    <ogc:Literal>EntityAlias</ogc:Literal>
                  </ogc:PropertyIsEqualTo>
                  <ogc:PropertyIsLike wildCard="*" escapeChar="/" singleChar="?">
                    <?indicio-case-sensitive false?>

    <ogc:PropertyName>$entity/$aliasSlot_1/ValueList/Value</ogc:PropertyName>
                    <ogc:Literal>*[*scotia*]*</ogc:Literal>
                  </ogc:PropertyIsLike>
                </ogc:And>
              </ogc:Or>
              <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 Spatial Queries

Spatial queries retrieve entities from the dataset based on either the bounding box or the geometry within which they are located. Queries of this type involve either entering the spatial coordinates for the bounding box, or by specifying a geometry. The only entities in the EPSG Registry that have an association with an area (of validity) are Coordinate Reference System, Datum and Coordinate Operation.

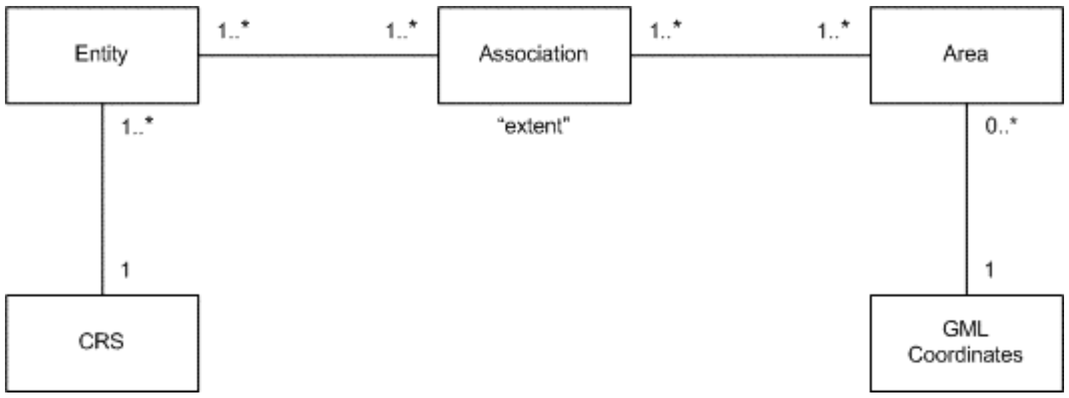


Figure 4: Object Relationships in Spatial Queries

The supported GML 3.1.1 Envelope and Geometry types 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 Find Entity by BBOX

The following query will return all entities related to the defined bounding box area that contains the city of Vancouver:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords
  xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    use a Spatial query (~BBOX) to match an Area record, then select the
    'Valid' Entity records which are related to the selected Area via an
    "Extent" Association.

    Note:  The bounding box is around the city of Vancouver, in British
Columbia, Canada.
  -->
  <Query
    typeNames="Association=extentAss_1 ClassificationNode=t_1
ExtrinsicObject=extent_1 ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- the target of the association is the area specified as
search criteria (~ BBOX or Envelope) -->
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- the association type must be correct -->

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- specify that the query is against EPSG Area records -->
              <ogc:PropertyName>/$extent_1/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- finally do the area search -->
              <ogc:PropertyName>/$extent_1/rim:Slot/@name</ogc:PropertyName>
              <ogc:Literal>http://www.opengis.net/gml/Envelope</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:BBOX xmlns:gml="http://www.opengis.net/gml">

<ogc:PropertyName>/$extent_1/rim:Slot/wrs:ValueList/wrs:AnyValue</ogc:PropertyN
ame>
              <gml:Envelope>
                <gml:coord>
                  <gml:X>-122.8</gml:X>
                  <gml:Y>49.3</gml:Y>
                </gml:coord>
                <gml:coord>
                  <gml:X>-123.3</gml:X>
                  <gml:Y>49.1</gml:Y>
                </gml:coord>
              </gml:Envelope>
            </ogc:BBOX>
          </ogc:And>
          <ogc:Not>
            <!-- constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- find the classification node for the object type -->
              <ogc:PropertyName>/$t_1/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@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>
```

4.3.2 Find Entity by Geometry

This type of query will find and return all entities that have a specific type of geometry. Filter operators may be combined with geometries to create more sophisticated queries.

Note: In the EPSG dataset, the only geometry data that is currently populated is BBOX.

The following query will return all entities that intersect with the region defined by the coordinates of the specified polygon around Stonehenge in the UK:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords
  xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    use a Spatial operator to match an Area related to an entity of interest;
    constrain the entity to be valid (= not deprecated).

    Note:  The region specified is around Stonehenge, UK.
  -->
  <Query
    typeNames="Association=extentAss_1 ExtrinsicObject=extent_1
ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- the target of the association is an area record -->
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- the association type must be 'Extent' -->

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- specify that the spatial component of the query is against
EPSG Area of Use records -->
              <ogc:PropertyName>/$extent_1/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:area</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <!-- the spatial operation -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent_1/rim:Slot/@name</ogc:PropertyName>
              <ogc:Literal>http://www.opengis.net/gml/Envelope</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Intersects>

<ogc:PropertyName>/$extent_1/rim:Slot/wrs:ValueList/wrs:AnyValue</ogc:PropertyN
ame>
              <gml:Polygon xmlns:gml="http://www.opengis.net/gml">
                <gml:outerBoundaryIs>
                  <gml:LinearRing>
                    <gml:coordinates decimal="." cs="," ts=" "
                      >51.17,-1.83 51.17,-1.815 51.181,-1.815 51.181,-1.83
51.17,-1.83</gml:coordinates>
                  </gml:LinearRing>
                </gml:outerBoundaryIs>
              </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 Used

The following query will find and return all the CRS entities that use the specified datum.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all CRS entities which used the specified Datum.
  -->
  <Query typeNames="Association=assoc ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsEqualTo>
            <!-- the source of the association is the object we return -->
            <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            <ogc:PropertyName>/$assoc/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$assoc/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:Datum</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- the ID of a Datum must be specified -->
            <ogc:PropertyName>/$assoc/@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 CRS by Base CRS

The following query will return all the CRS entities that use the base CRS with code “`4600`”:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all CRS entities which use the specified BaseCRS.
  -->
  <Query typeNames="Association=assoc ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsEqualTo>
            <!-- the source of the association is the object we return -->
            <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            <ogc:PropertyName>/$assoc/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$assoc/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:BaseCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- the ID of a BaseCRS must be specified -->
            <ogc:PropertyName>/$assoc/@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 Operation by Method Used

The following query will return operations that use the method with code "`9664`":

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all Coordinate Operation entities which use the specified Method.
  -->
  <Query typeNames="Association=assoc ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsEqualTo>
            <!-- the source of the association is the object we return -->
            <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            <ogc:PropertyName>/$assoc/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$assoc/@associationType</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:AssociationType:GML:Method</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- the ID of a Operation Method must be specified -->
            <ogc:PropertyName>/$assoc/@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 Operation by CRS Used

The following query will return all coordinate operations that use the CRS with code “4973”:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all Coordinate Operation entities which use the specified Source
CRS.
  -->
  <Query typeNames="Association=assoc ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsEqualTo>
            <!-- the source of the association is the object we return -->
            <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
            <ogc:PropertyName>/$assoc/@sourceObject</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>/$assoc/@associationType</ogc:PropertyName>

<ogc:Literal>urn:ogc:def:AssociationType:GML:SourceCRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- the ID of a CRS must be specified -->
            <ogc:PropertyName>/$assoc/@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 Survey and Positioning 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

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
```

```
    xmlns:wrs="http://www.opengis.net/cat/wrs"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:gmd="http://www.isotc211.org/2005/gmd"
    xmlns:gco="http://www.isotc211.org/2005/gco"
    startPosition="1" maxRecords="10"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
    <!--
      Purpose:
         Query Geographic 3D CRS records.
    -->
    <Query typeNames="ExtrinsicObject=entity">
      <ElementSetName typeNames="$entity">full</ElementSetName>
      <Constraint version="1.1.0">
        <ogc:Filter>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the type of the entity we are searching for is GeodeticCRS -->
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
              <ogc:Literal>urn:ogc:def:ObjectType:GML:GeodeticCRS</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <!-- the sub-type is specified in the EntitySubType slot -->
              <ogc:PropertyName>/$entity/Slot/@name</ogc:PropertyName>
              <ogc:Literal>EntitySubType</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/Slot/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 as 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 find and 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"
    xmlns:wrs="http://www.opengis.net/cat/wrs"
    xmlns:ogc="http://www.opengis.net/ogc"
    startPosition="1" maxRecords="50"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
  <!--
    Retrieve valid Version History records.
    Sort by the Version History Slot named 'VersionDate'; use Ascending order
    so that the last Version History record in the result list will represent
    the current version of the dataset.
  -->
  <Query typeNames="ExtrinsicObject=eo eo/rim:Slot=versionDate">
    <ElementSetName typeNames="$eo">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsLike wildCard="*" escapeChar="/" singleChar="?">
            <ogc:PropertyName>$eo/@id</ogc:PropertyName>
            <ogc:Literal>*:EPSG::*</ogc:Literal>
          </ogc:PropertyIsLike>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$eo/@objectType</ogc:PropertyName>
            <ogc:Literal>urn:x-ogp:def:ObjectType:EPSG:version-
history</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$eo/@status</ogc:PropertyName>
            <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Approved</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$eo/$versionDate/@name</ogc:PropertyName>
            <ogc:Literal>VersionDate</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:And>
      </ogc:Filter>
    </Constraint>
    <ogc:SortBy>
      <ogc:SortProperty>
        <ogc:PropertyName>$eo/$versionDate/ValueList/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"
    xmlns:ogc="http://www.opengis.net/ogc"
    startPosition="1" maxRecords="100"
    outputFormat="application/xml; charset=UTF-8"
    resultType="results">
  <!--
    Retrieve the ReleaseObject releated 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="Association=a RegistryObject=release">
    <ElementSetName typeNames="release">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <!-- find the correct association -->
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$a/targetObject</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:version-history:EPSG::6.15</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>$a/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>$a/sourceObject</ogc:PropertyName>
            <ogc:PropertyName>$release/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 CRS 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 with EPSG code “`2295`”:

```
<?xml version="1.0" encoding="UTF-8"?>
<rim:ExtrinsicObject
      xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
      xmlns:wrs="http://www.opengis.net/cat/wrs"
      id="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="DataSource" slotType="xsd:string">
        <rim:ValueList>
            <rim:Value>EPSG</rim:Value>
        </rim:ValueList>
    </rim:Slot>
```

```
        <rim:Slot name="isDeprecated" slotType="xsd:boolean">
            <rim:ValueList>
                <rim:Value>false</rim:Value>
            </rim:ValueList>
        </rim:Slot>
        <rim:Slot name="EntityAlias" slotType="xsd:string">
            <rim:ValueList>
                <rim:Value>[ATS77 / MTM NS zone 5]</rim:Value>
            </rim:ValueList>
        </rim:Slot>
        <rim:Slot name="SortKey" slotType="xsd:string">
            <rim:ValueList>
                <rim:Value>1ats77 / mtm nova scotia zone 5</rim:Value>
            </rim:ValueList>
        </rim:Slot>
        <rim:Slot name="AreaOfUse" slotType="xsd:string">
            <rim:ValueList>
                <rim:Value>Canada - Nova Scotia - W of 63°W</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: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" encoding="UTF-8"?>
    <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:0.1:dataset"
          xmlns:gml="http://www.opengis.net/gml"
          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"
          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="EPSG">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"/>
    </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 OGP 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 any CRS sub-type), 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 URN's for each group of entity types.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all CRS entities which contain the search term in their Name.
  -->
  <Query typeNames="ExtrinsicObject=entity ClassificationNode=cnode,baseType">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:PropertyIsEqualTo>
            <!-- the entity's objectType matches a ClassificationNode -->
            <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            <ogc:PropertyName>/$cnode/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- the objectType ClassificationNode has baseType as a parent -->
            <ogc:PropertyName>/$cnode/@parent</ogc:PropertyName>
            <ogc:PropertyName>/$baseType/@id</ogc:PropertyName>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsEqualTo>
            <!-- baseType URN must be CRS -->
            <ogc:PropertyName>/$baseType/@id</ogc:PropertyName>
            <ogc:Literal>urn:ogc:def:ObjectType:GML:CRS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
          <ogc:PropertyIsLike wildCard="*" singleChar="?" escapeChar="/">
            <!-- Name property contains the search-term -->

<ogc:PropertyName>/$entity/Name/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 Region by Name

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

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRecords xmlns="http://www.opengis.net/cat/csw"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return ANY Entity object related to an Area which matches the
       text search term specified.

    Note:  No sorting performed.
  -->
  <Query typeNames="Association=extentAss_1 ClassificationNode=t_1
ExtrinsicObject=extent_1 ExtrinsicObject=entity ">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent_1/@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_1/rim:Name/rim:LocalizedString@value</ogc:PropertyNa
me>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/rim:Description/rim:LocalizedString@value</ogc:Pro
pertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- find the classification node for the object type -->
              <ogc:PropertyName>/$t_1/@id</ogc:PropertyName>
              <ogc:PropertyName>$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>/$t_1/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@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>
```

<u>5.7 How To... Sort Result Sets (Using SortBy)</u>

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>
  <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, assuming the alias `$countrySlot` has been defined, looks like this:

```
<Query typeNames="ExtrinsicObject=eo eo/rim:Slot=countrySlot">
    ...
    <ogc:SortBy>
        <ogc:SortProperty>

<ogc:PropertyName>/$eo/$countrySlot/ValueList/Value</ogc:PropertyName>
            <ogc:SortOrder>DESC</ogc:SortOrder>
        </ogc:SortProperty>
        <ogc:SortProperty>

<ogc:PropertyName>/$eo/Name/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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all Entity objects related to an Area which matches the
       text search term specified.

    Note:  Added SortBy clause.
  -->
  <Query typeNames="r/rim:Slot=sortKeySlot_1 Association=extentAss_1
ClassificationNode=t_1 ExtrinsicObject=extent_1 ExtrinsicObject=entity">
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent_1/@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>/$entity/$sortKeySlot_1/@name</ogc:PropertyName>
              <ogc:Literal>SortKey</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsGreaterThan>

<ogc:PropertyName>/$entity/$sortKeySlot_1/ValueList/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_1/rim:Name/rim:LocalizedString@value</ogc:PropertyNa
me>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/rim:Description/rim:LocalizedString@value</ogc:Pro
pertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- find the classification node for the object type -->
              <ogc:PropertyName>/$t_1/@id</ogc:PropertyName>
              <ogc:PropertyName>/$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>/$t_1/@path</ogc:PropertyName>
                <ogc:Literal>/urn:oasis:names:tc:ebxml-
regrep:classificationScheme:ObjectType/RegistryObject/ExtrinsicObject/Geodetic-
Entity*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@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>/$entity/$sortKeySlot_1/ValueList/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="ExtrinsicObject=entity">
        <?indicio-distinct-values true?>
        <ElementSetName typeNames="$entity">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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return all Entity objects related to an Area which matches the
       text search term specified.

    Note:  No SortBy clause.
           Added filter clause to also match search term in Alias property.
           Requires use of DISTINCT processing instruction to exclude
duplicates.
  -->
  <Query typeNames="extent_1/rim:Slot=aliasSlot_5 Association=extentAss_1
ClassificationNode=t_1 ExtrinsicObject=extent_1 ExtrinsicObject=entity">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent_1/@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_1/rim:Name/rim:LocalizedString@value</ogc:PropertyNa
me>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/rim:Description/rim:LocalizedString@value</ogc:Pro
pertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:And>
                <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extent_1/$aliasSlot_5/@name</ogc:PropertyName>
                  <ogc:Literal>Alias</ogc:Literal>
                </ogc:PropertyIsEqualTo>
                <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                  <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/$aliasSlot_5/ValueList/Value</ogc:PropertyName>
```

```
                  <ogc:Literal>*scotia*</ogc:Literal>
                </ogc:PropertyIsLike>
              </ogc:And>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- find the classification node for the object type -->
              <ogc:PropertyName>/$t_1/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!--
              ensure that the ClassificationNode belongs to either the
              Geodetic object scheme, or the EPSG metadata scheme
              (this effectively screens out entity types which are not of
interest)
            -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@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>/$t_1/@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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:gco="http://www.isotc211.org/2005/gco"
  startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8"
  resultType="results">
  <!--
    Purpose:
       Return ANY Entity object which is related to an Area which matches the
       text search term specified.

    Note:  Added SortBy clause.
           Added filter clause to also match search term in Alias property.
           Requires use of DISTINCT processing instruction to exclude
duplicates.
  -->
  <Query typeNames="r/rim:Slot=sortKeySlot_1 extent_1/rim:Slot=aliasSlot_5
Association=extentAss_1 ClassificationNode=t_1 ExtrinsicObject=extent_1
ExtrinsicObject=entity">
    <?indicio-distinct-values true?>
    <ElementSetName typeNames="$entity">full</ElementSetName>
    <Constraint version="1.1.0">
      <ogc:Filter>
        <ogc:And>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- the source of the association is the object we return -->
              <ogc:PropertyName>/$entity/@id</ogc:PropertyName>
              <ogc:PropertyName>/$extentAss_1/@sourceObject</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extentAss_1/@associationType</ogc:PropertyName>
              <ogc:Literal>urn:x-
ogp:def:AssociationType:EPSG:Extent</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extentAss_1/@targetObject</ogc:PropertyName>
              <ogc:PropertyName>/$extent_1/@id</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$extent_1/@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>/$entity/$sortKeySlot_1/@name</ogc:PropertyName>
              <ogc:Literal>SortKey</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:PropertyIsGreaterThan>

<ogc:PropertyName>/$entity/$sortKeySlot_1/ValueList/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_1/rim:Name/rim:LocalizedString@value</ogc:PropertyNa
me>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/rim:Description/rim:LocalizedString@value</ogc:Pro
pertyName>
                <ogc:Literal>*scotia*</ogc:Literal>
              </ogc:PropertyIsLike>
              <ogc:And>
                <ogc:PropertyIsEqualTo>

<ogc:PropertyName>/$extent_1/$aliasSlot_5/@name</ogc:PropertyName>
                  <ogc:Literal>Alias</ogc:Literal>
                </ogc:PropertyIsEqualTo>
                <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                  <?indicio-case-sensitive false?>

<ogc:PropertyName>/$extent_1/$aliasSlot_5/ValueList/Value</ogc:PropertyName>
```

```
                  <ogc:Literal>*scotia*</ogc:Literal>
                </ogc:PropertyIsLike>
              </ogc:And>
            </ogc:Or>
          </ogc:And>
          <ogc:Not>
            <!-- constrain the Status of the entity to be Valid -->
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>/$entity/@status</ogc:PropertyName>
              <ogc:Literal>urn:oasis:names:tc:ebxml-
regrep:StatusType:Deprecated</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Not>
          <ogc:And>
            <ogc:PropertyIsEqualTo>
              <!-- find the classification node for the object type -->
              <ogc:PropertyName>/$t_1/@id</ogc:PropertyName>
              <ogc:PropertyName>/$entity/@objectType</ogc:PropertyName>
            </ogc:PropertyIsEqualTo>
            <!--
              ensure that the ClassificationNode belongs to either the
              Geodetic object scheme, or the EPSG metadata scheme
              (this effectively screens out entity types which are not of
interest)
            -->
            <ogc:Or>
              <ogc:PropertyIsLike wildCard="*" escapeChar="\" singleChar="?">
                <ogc:PropertyName>/$t_1/@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>/$t_1/@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>/$entity/$sortKeySlot_1/ValueList/Value[1]</ogc:PropertyName>
        <ogc:SortOrder>ASC</ogc:SortOrder>
      </ogc:SortProperty>
    </ogc:SortBy>
  </Query>
</GetRecords>
```

5.10 How To... Find Newer Valid Entities from 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.

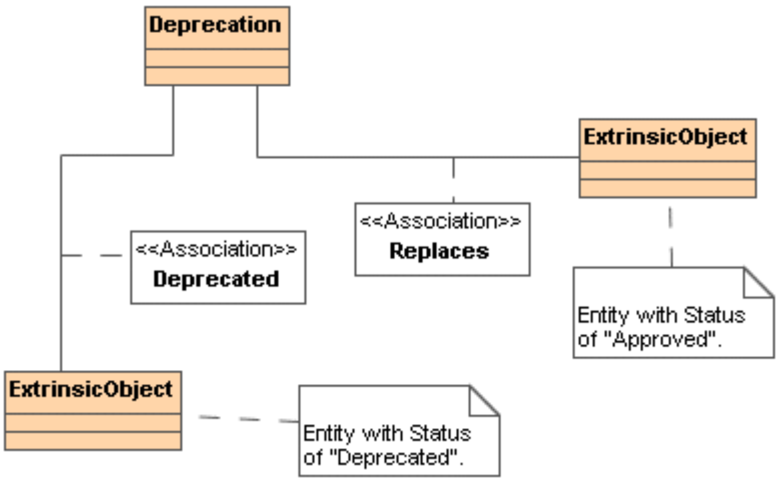


Figure 5: Associations Connecting a Deprecated Entity to its Valid Replacement Entity

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

```
<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:0.1:dataset"
             xmlns:gml="http://www.opengis.net/gml"
             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"
             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="EPSG">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"
  xmlns:wrs="http://www.opengis.net/cat/wrs"
  xmlns:ogc="http://www.opengis.net/ogc" startPosition="1" maxRecords="10"
  outputFormat="application/xml; charset=UTF-8" resultType="results">
  <!--
    Find any entities which have replaced an invalid (or deprecated) entity.
  -->
  <Query typeNames="ExtrinsicObject=depn,entity Association=depd,repl">
    <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

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 3.2.0 objects in XML format. The following sections provide brief descriptions and diagrams for:

- the GML 3.2.0 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 to a pre-release version of GML 3.2.0. 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 content of the zip file archive is:

- File EPSG.xsd:	the EPSG Registry Schema.
- Folder "gml":	the GML 3.2.0 Schema; the root document of the GML Schema is file "gml/gml.xsd"
- Folder "Smil":	this schema is here because there are errors in publicly available smil schemas at w3c site.
- Folder "xlink":	the imported W3C XLink schema (see W3C XLink 1.0)
- Folder "iso19139":	the imported GMD schema and contained schemas (see ISO/TS 19139)

The namespace for the EPSG Registry GML application schema is:

```
urn:x-ogp:spec:schema-xsd:EPSG:0.1:dataset
```

NOTE regarding GML 3.2.0. A major change introduced into GML 3.2.0 which occurred just prior to the EPSG Registry release was a namespace change. The namespace went from http://www.opengis.net/gml to http://www.opengis.net/gml/3.2. The EPSG Registry currently uses the former namespace, without the version number

There have been numerous other minor changes to the GML 3.2.0 specification, none of which affect the EPSG Registry application schema.[1]

[1] Note for future reference: Two issues which will affect the EPSG Registry application schema if it were to be upgraded to GML 3.2.1 or later (currently there are no plans for this) are:

- The CoordinateSystemAxis element has a "uom" attribute which in GML 3.2.1 would no longer use the "gml" prefix. In GML 3.2.0 it is expressed as <gml:CoordinateSystemAxis **gml:**uom="..."> while in GML 3.2.1 it is expressed as <gml:CoordinateSystemAxis **uom="..."**>

- The <isSphere> element on an Ellipsoid has had a change to the data-type of its value to be a boolean. GML 3.2.0 uses <isSphere>Sphere</isSphere> while GML 3.2.1 requires <isSphere>true</isSphere>

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)*".

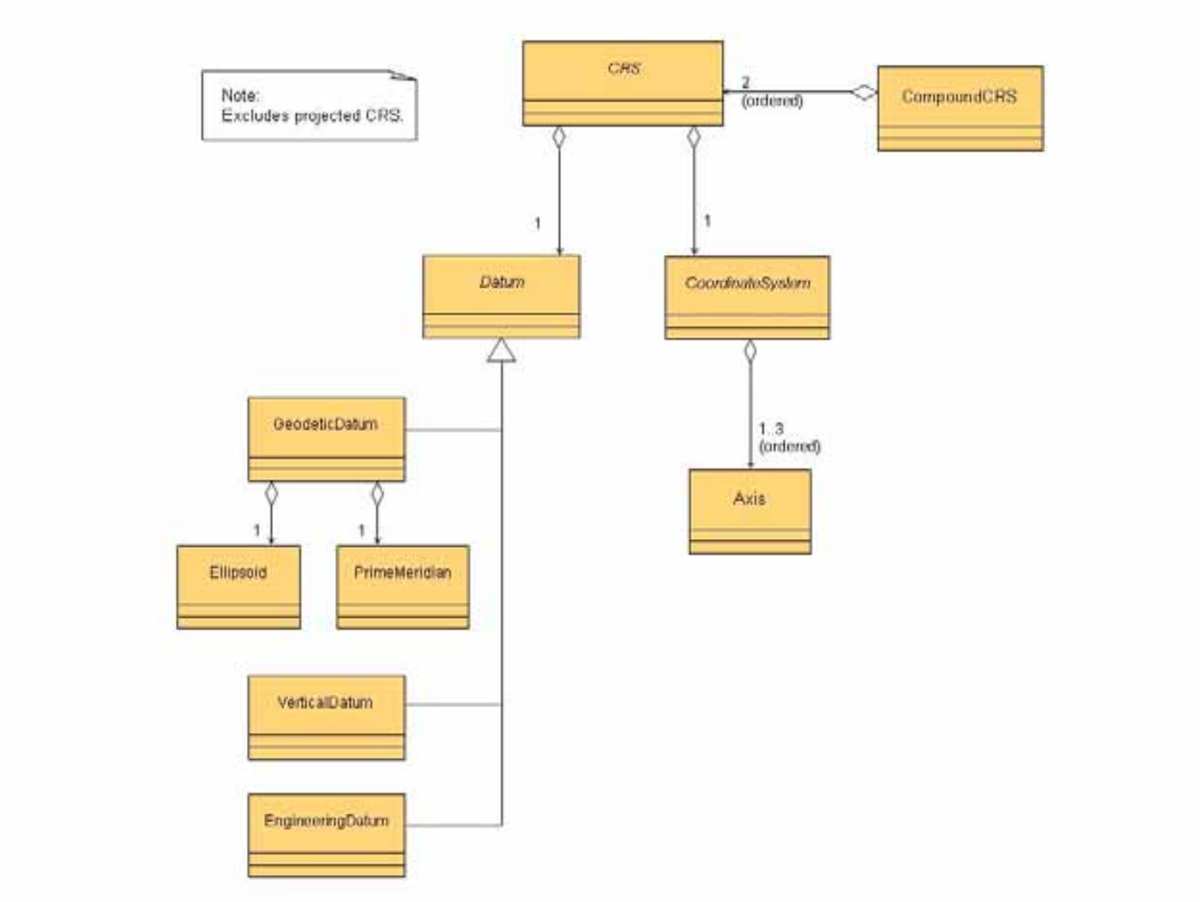


Figure A–1: Coordinate Reference System

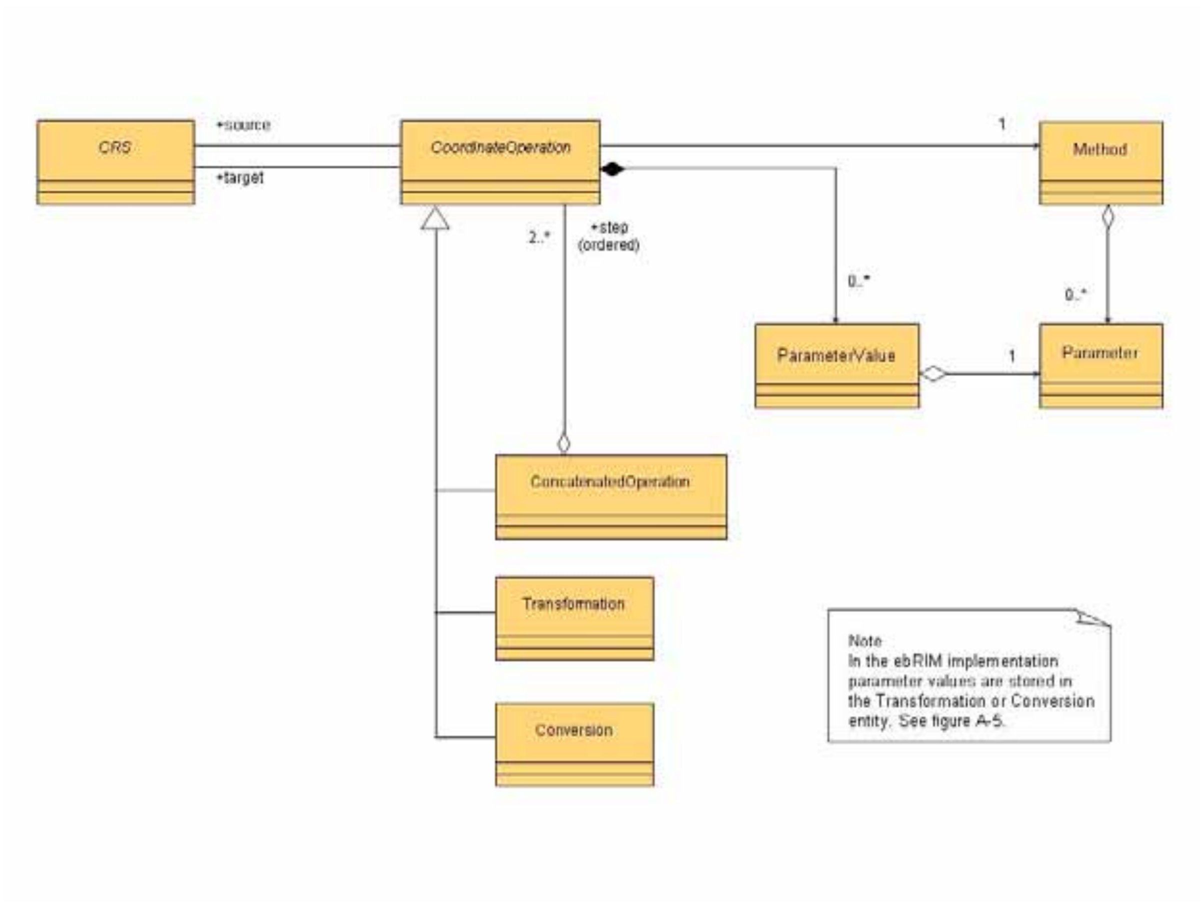


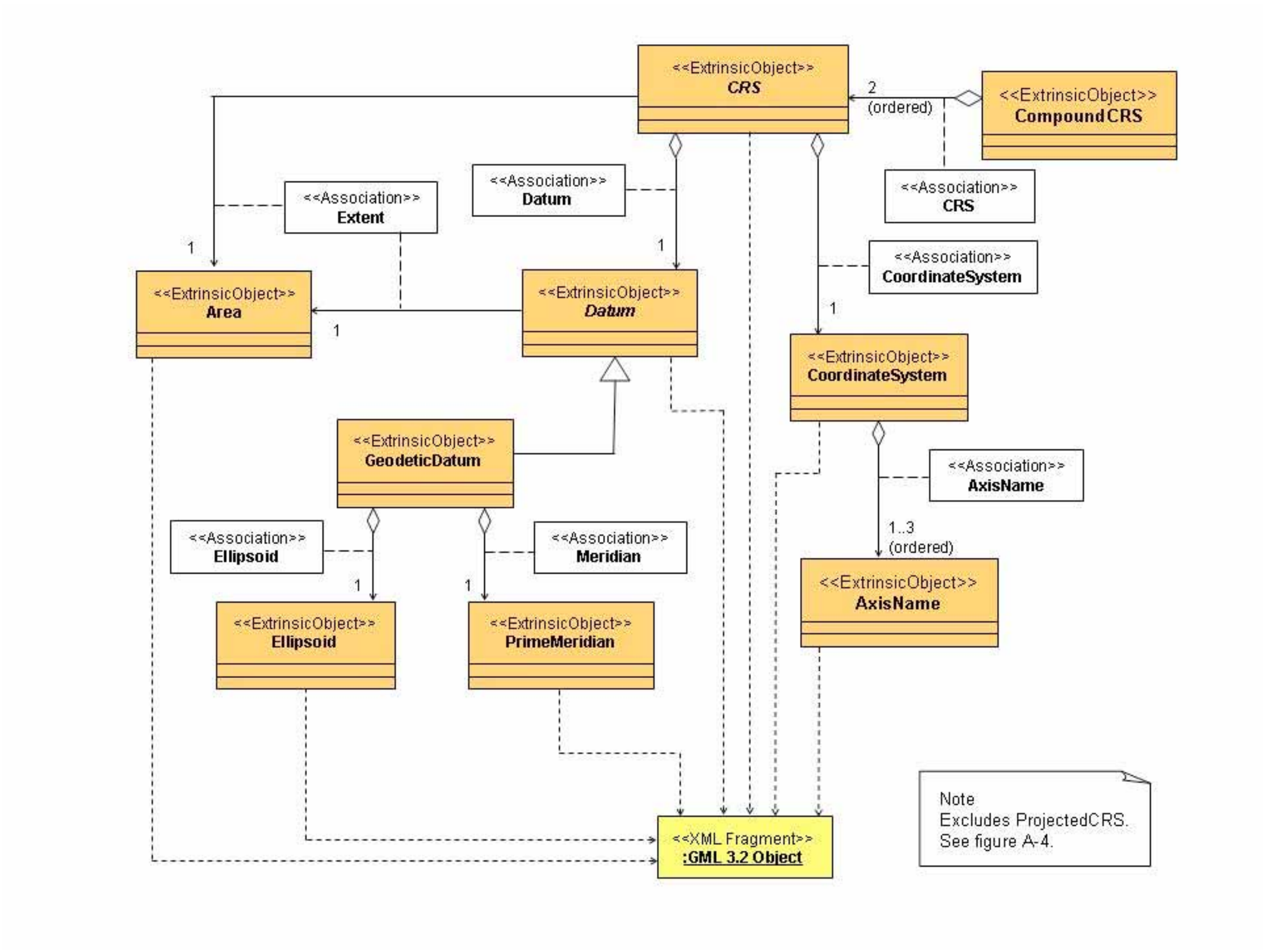
Figure A–2: Coordinate Operations

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 <u>ebRIM Model</u>

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.



<<ExtrinsicObject>> CRS
2 (ordered)
<<ExtrinsicObject>> CompoundCRS
<<Association>> CRS
<<Association>> Extent
<<Association>> Datum
<<Association>> CoordinateSystem
1
1
1
1
<<ExtrinsicObject>> Area
<<ExtrinsicObject>> Datum
<<ExtrinsicObject>> CoordinateSystem
<<ExtrinsicObject>> GeodeticDatum
<<Association>> AxisName
1..3 (ordered)
<<Association>> Ellipsoid
<<Association>> Meridian
1
1
<<ExtrinsicObject>> Ellipsoid
<<ExtrinsicObject>> PrimeMeridian
<<ExtrinsicObject>> AxisName
Note Excludes ProjectedCRS. See figure A-4.
<<XML Fragment>> :GML 3.2 Object

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

B.

Note A variation between ISO 19111 and the EPSG Registry exists: the domainOfValidity, defined in ISO 19111 as an EX_Extent data type, has been implemented in the EPSG Registry as the entity Area, containing an EX-Extent, which in the case of the EPSG Registry is the applicable minimum bounding box.

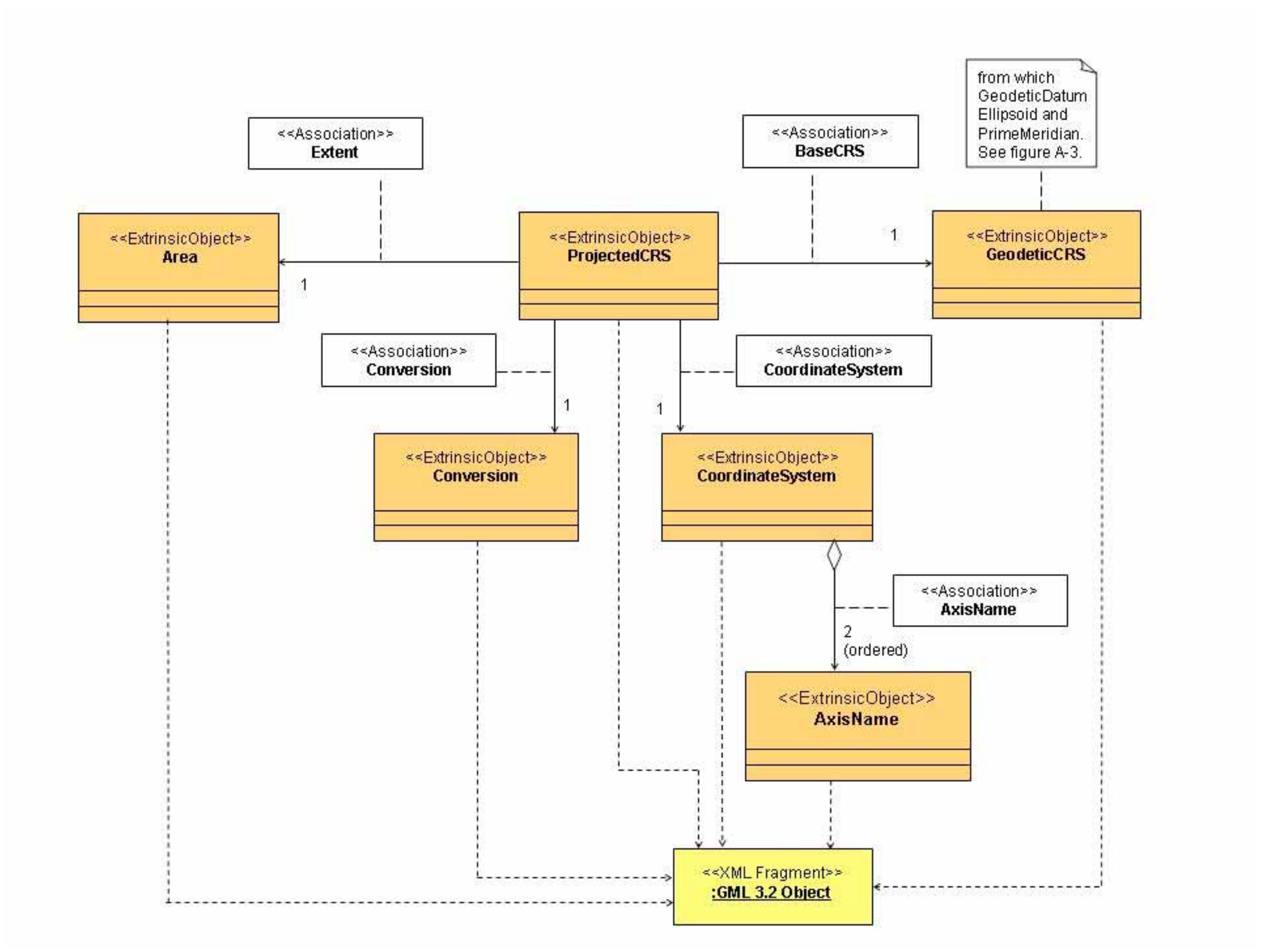


Figure A–4: EPSG Registry Model (ebRIM): ProjectedCRS

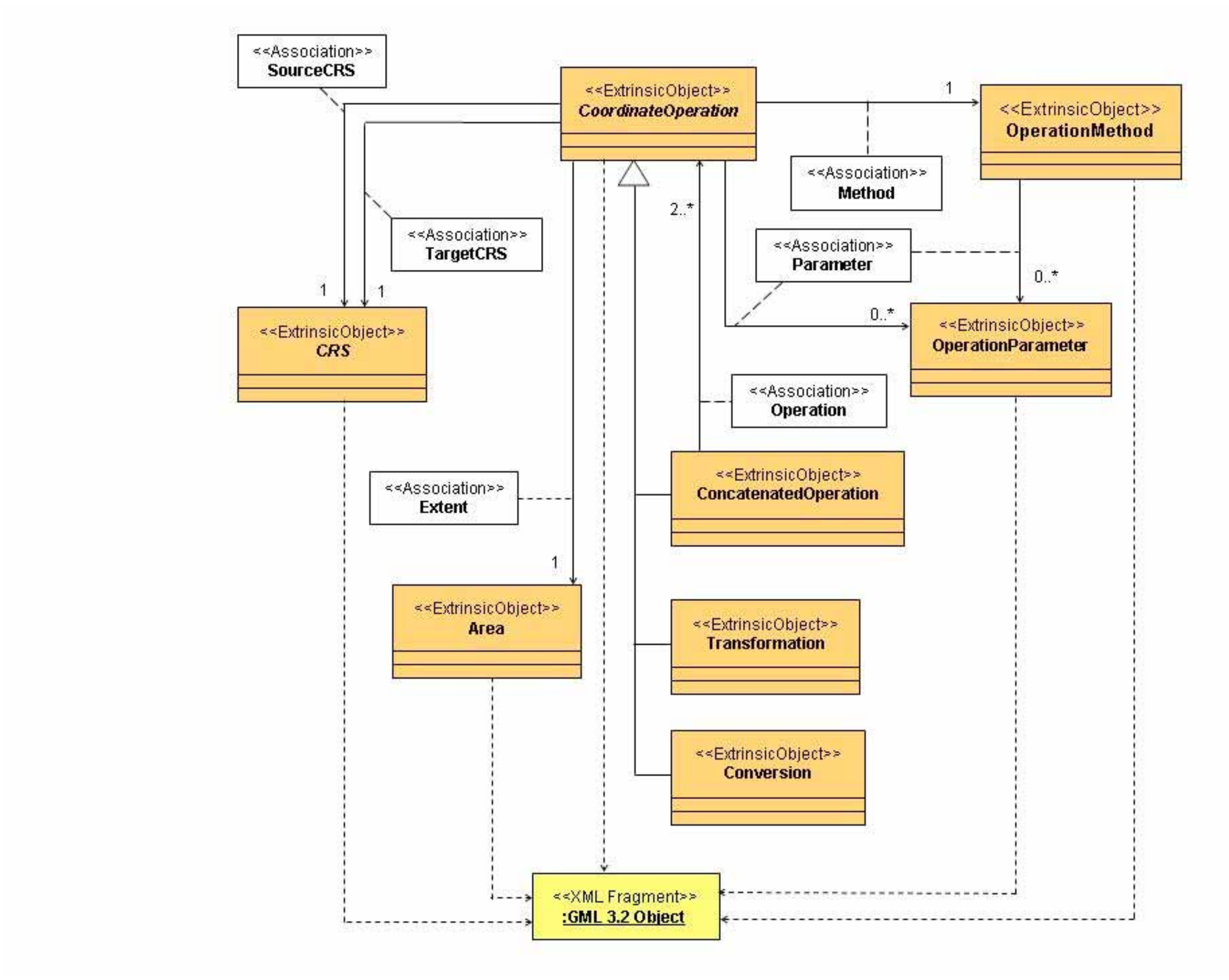

<<Association>> SourceCRS
<<ExtrinsicObject>> CoordinateOperation
1
<<ExtrinsicObject>> OperationMethod
<<Association>> Method
2..*
<<Association>> TargetCRS
<<Association>> Parameter
0..*
1
1
<<ExtrinsicObject>> CRS
0..*
<<ExtrinsicObject>> OperationParameter
<<Association>> Operation
<<Association>> Extent
<<ExtrinsicObject>> ConcatenatedOperation
1
<<ExtrinsicObject>> Area
<<ExtrinsicObject>> Transformation
<<ExtrinsicObject>> Conversion
<<XML Fragment>> :GML 3.2 Object

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

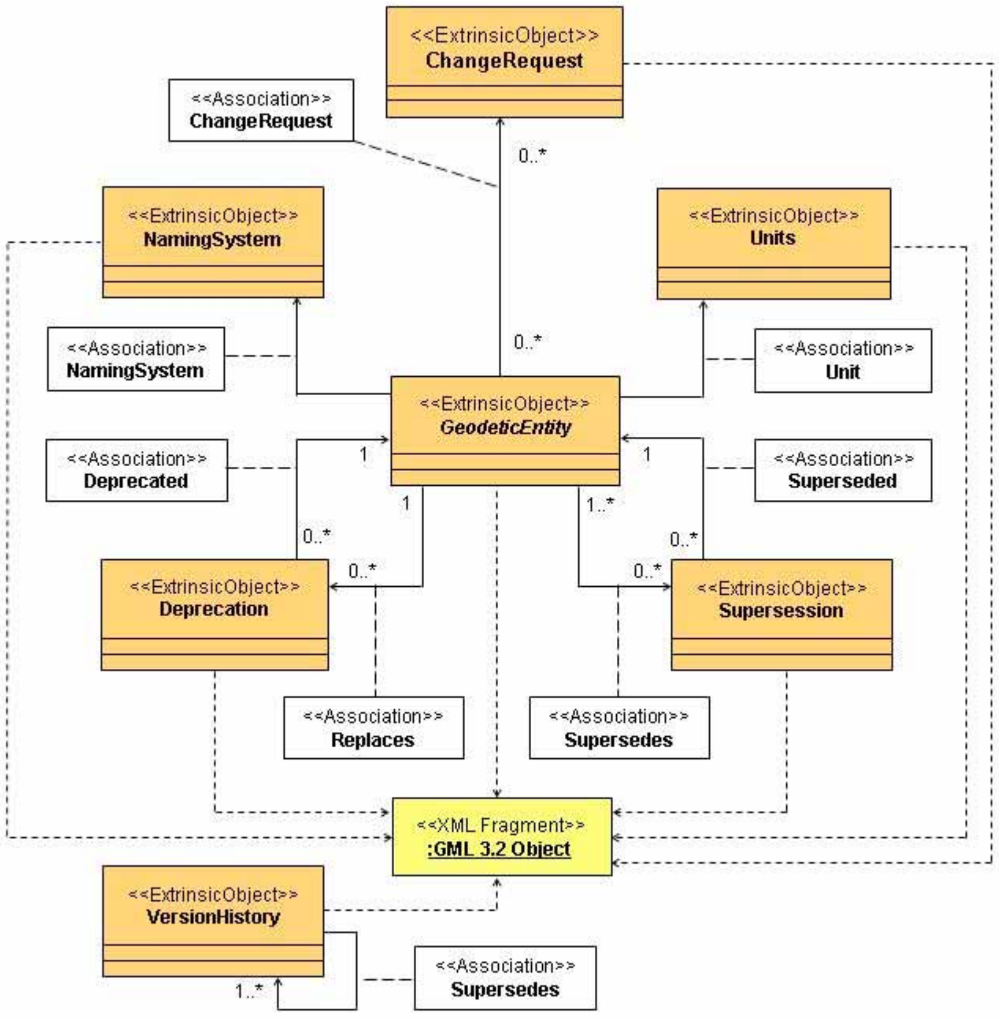

<<ExtrinsicObject>> ChangeRequest
<<Association>> ChangeRequest
0..*
<<ExtrinsicObject>> NamingSystem
<<ExtrinsicObject>> Units
<<Association>> NamingSystem
0..*
<<Association>> Unit
<<ExtrinsicObject>> GeodeticEntity
<<Association>> Deprecated
1
1
<<Association>> Superseded
1
1..*
0..*
0..*
0..*
0..*
<<ExtrinsicObject>> Deprecation
<<ExtrinsicObject>> Supersession
<<Association>> Replaces
<<Association>> Supersedes
<<XML Fragment>> :GML 3.2 Object
<<ExtrinsicObject>> VersionHistory
<<Association>> Supersedes
1..*

Figure A–6: EPSG Registry Model (ebRIM): Additional EPSG Metadata Objects

A.3.1 <u>Entity Types</u>

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
```

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
```

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.

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

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 GML specification:

Specification	Version	Document	Short Name
OpenGIS® Geography Markup Language (GML) Encoding Specification	3.2	OGC document 07-036	GML
OGC Abstract Specification Topic 2, Spatial referencing by coordinates	r3	OGC document 04-046r3	Topic 2