Ecological Metadata Language (EML)

Ecological Metadata Language (EML) is a metadata specification developed by the ecology discipline and for the ecology discipline. It is based on prior work done by the Ecological Society of America and associated efforts (Michener et al., 1997, Ecological Applications). EML is implemented as a series of XML document types that can by used in a modular and extensible manner to document ecological data. Each EML module is designed to describe one logical part of the total metadata that should be included with any ecological dataset.

Send any comments, errors, or suggestions to **eml-dev@ecoinformatics.org** or through the **EML Bug Tracking system**. The preferred way to submit problems with EML or feature requests is the bug tracking system.

EML Version 2.1.0

You can access the EML specification online by reading it in HTML format, or you can download the entire specification, including both the HTML documentation and the XML Schema files.

- **EML 2.1.0 Specification** -- Read it online

  OR

- **Download EML**
  The download consists of the EML modules, described in the **XML Schema** language. In addition, the full documentation on the modules is provided in HTML format.
- **Changes to EML in version 2.1.0**
- **EML Frequently Asked Questions (FAQ)**
- **Validation service for EML**
  In addition to the online service found at the previous link, the EML distribution itself contains the validation software for your use (see "lib/runEMLParser" for details on how to run it).

About the EML Project

The EML project is an open source, community oriented project dedicated to providing a high-quality metadata specification for describing data relevant to the ecological discipline. The project is completely comprised of **voluntary project members** who donate their time and experience in order to advance information management for ecology. Project decisions are made by consensus according to the voting procedures described in the **ecoinformatics.org Charter**.

We welcome contributions to this work in any form. Individuals who invest substantial amounts of time and make valuable contributions to the development and maintenance of EML (in the opinion of current project members) will be invited to become EML project members according to the rules set forth in the **ecoinformatics.org Charter**. Contributions can take many forms, including the development of the EML schemas, writing documentation, and helping with maintenance, among others.

Development Information

Developers may be interested in browsing the **source code SVN repository** that we use in developing EML. This always

contains the most recent development version of EML, and therefore may be in flux, or otherwise broken. It is unlikely that it will contain the same files that are in the current release (2.1.0). Use at your own risk. Write access to this repository is reserved for **EML project members**. We welcome contributions to this work in any form. Contributions can take many forms, including the development of the EML schemas, writing documentation, and helping with maintenance, among others. Non-project members can contribute by submitting their feedback, revisions, fixes, code, or any other contribution through the **eml-dev@ecoinformatics.org** mailing list, or through the **EML Bug Tracking system**. The preferred way to submit problems with EML or feature requests is the bug tracking system.

Older versions (deprecated)

The following versions are still available for reference purposes, although they have been superseded by the current version (2.1.0). Please make every effort to use the current version.

- **EML 2.0.1**
- **EML 2.0.0**
- **EML 1.4.1**

Information for EML 2.1.0 Document Authors

Table of Contents

Several modifications to the EML schema made in version 2.1.0 will require changes to how EML documents are structured, and these changes are highlighted here. EML authors should also refer to the affected sections in the normative schema documents for complete usage information and examples. Existing EML 2.0-series documents can be converted to EML 2.1.0 using the XSL stylesheet that accompanies this release, as described in section 2 below.

The EML 2.1.0 release addresses several errors with respect to W3C specifications for XML schema (http://www.w3.org/TR/xml). Although the changes are small, they are incompatible with EML 2.0.0 and 2.0.1 schemas, which necessitated advancing the version number to "2.1". The STMML schema was also found to be invalid with respect to XML Schema language, and the most reasonable fix for this bug also is incompatible with its earlier versions. EML users should note that the STMML schema error was *not* related to elements used directly by EML (i.e., <unitList> or <unitType>). However, EML imports all of STMML, and authors of EML documents may have made use of other parts of that schema. Therefore, it was decided to advance the namespace used for STMML-related imports to "stmml-1.1", in keeping with the EML version naming pattern. The STMML authors have been contacted, and they are interested in our development and use of STMML.

Other features and enhancements were added to this release that represent significant improvements. The XML data type requirements for several elements were changed, in some cases to constrain their content, and in other cases to increase flexibility. The names of two elements were changed to make them consistent throughout EML. In the literature schema two elements became optional so that EML could accommodate in-press publications where the volume and page range are not yet known. Support for two new optional elements was also added: a 'contact' tree can now be used in the literature module, and a ‘descriptive’ element can be used in distribution trees.

For the most part, EML 2.1.0 does not introduce major new features, or require a shift in use or implementation. There was a deliberate decision to balance the impact on instance document authors with necessary schema maintenance, and to prepare the schema for the next phase of planned improvements and features. Some of the changes to EML 2.1.0 are invisible to document authors; see the 'Readme' that accompanies the distribution for a complete list of the bugs addressed, and for information of interest to developers.

1. Changes and New Features in EML 2.1.0

1.1. EML Schema validity

EML allows authors to place any XML markup in <additionalMetadata> sections at the end of the document. The content model for <additionalMetadata> includes an optional <describes> element so that references to EML nodes can be included as necessary. In EML 2.0 this element was placed alongside the additional XML content; however, this construct is not allowed in XML Schema, and the error was not reported by XML parsers available at the time EML 2.0 was released. In EML 2.1.0, the error has been corrected by adding a required child element to the <additionalMetadata> section to contain the "<xs:any>" XML content.

<additionalMetadata> sections must include the child <metadata> to contain the additional XML markup. The

optional <describes> element may still be included to reference a particular node of the document. Multiple <describes> elements can be included if needed. Examples of documents written against 2.1.0 and 2.0.1 are below. Also see the **additionalMetadata normative documentation**.

In EML 2.0.1, an additionalMetadata section looked like this:

```
...
 <additionalMetadata>
 <describes>123</describes>
  <unitList>
   <unit name="speciesPerSquareMeter"
         unitType="arealDensity"
         id="speciesPerSquareMeter"
         parentSI="numberPerSquareMeter"
         multiplierToSI="1"/>
  </unitList>
 </additionalMetadata>...
```

In EML 2.1.0, the markup must be enclosed within <metadata> tags:

```
...
<additionalMetadata>
<describes>123</describes>
 <metadata>
  <unitList>
   <unit name="speciesPerSquareMeter"
         unitType="arealDensity"
         id="speciesPerSquareMeter"
         parentSI="numberPerSquareMeter"
         multiplierToSI="1"/>
  </unitList>
 </metadata>
</additionalMetadata>
...
```

1.2. STMML Schema validity

EML makes use of the Scientific Technical and Medical Markup Language schema (STMML, stmml.xsd) for describing units, and the STMML schema was also found to be invalid. The error was not related to elements used directly by EML (i.e., <unitList> or <unitType>), however some authors may have used other parts of stmml.xsd in their documents. The required schema changes were not compatible with STMML-1.0, and the EML development group is working with the STMML developers on this issue. Since EML now imports a version of STMML that is not identical to that available from its authors, it was decided to advance the namespace used by EML 2.1.0 for stmml-related files to "stmml-1.1". To import stmml.xsd into one of your EML 2.1.0 documents use the XML namespace declaration for STMML in the code below:

```
<?xml version="1.0"?>
<eml:eml
 packageId="eml.1.1" system="knb"
 xmlns:eml="eml://ecoinformatics.org/eml-2.1.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:stmml="http://www.xml-cml.org/schema/stmml-1.1"
 xsi:schemaLocation="eml://ecoinformatics.org/eml-2.1.0 eml.xsd">
 <dataset>
        ...
 </dataset>
```

```
 <additionalMetadata>
  <metadata>
   <stmml:unitList xmlns:stmml="http://www.xml-cml.org/schema/stmml-1.1"
    xsi:schemaLocation="http://www.xml-cml.org/schema/stmml-1.1 stmml.xsd">
    <stmml:unit name="gramsPerSquareMeter"
                unitType="arealMassDensity"
                id="gramsPerSquareMeter"
                parentSI="kilogramsPerSquareMeter"
                multiplierToSI=".001">
     ..
   </stmml:unitList>
  </metadata>
 </additionalMetadata>
</eml:eml>
```

1.3. Location of Access Control Trees

In EML 2.0.1 an <access> tree could be included in each top-level module (i.e. dataset, citation, software, or protocol) to control access to the entire metadata document. Additionally, to control access to individual entities, some authors put <access> trees in <additionalMetadata> sections and used <describes> elements to reference their <distribution> nodes. Authors may have inferred that access control could be applied to any node with this practice. However, node-level access control is problematic to implement, and in practice only access trees that reference distribution nodes are recognized (as was stated in EML 2.0.1 documentation). A better solution is to locate <access> nodes above or near the node to which the access rules should be applied. This feature has been added to EML 2.1.0.

In EML 2.1.0, access trees can be placed in 2 locations. To control the entire metadata document (i.e., "document-level access"), an <access> tree should be placed as a child of the root element (**EML image**). If a metadata author wishes to override the document-level control for a specific entity, an additional access tree may be placed as the last child of a <distribution> element within the <physical> tree of that entity (**Physical Distribution Type image**). The structure of the access module itself has not changed (**access module documentation**).

Example 1. To control access to all the metadata and by default to the data, use an <access> element at the top level:

```
<eml:eml>
 <access>
  ...
 </access>
 <dataset>
   ...
 </dataset>
 <additionalMetadata>
   ...
 </additionalMetadata>
</eml:eml>
```

Example 2. Access rules can still be specified for any data entity by placing an access tree under that entity's physical/distribution element. The following example illustrates how a dataTable's access tree can be used to override permissions set at the document level. If no access is specified in distribution then the document-level access rules are applied.

```
<eml:eml>
 <dataset>
```

```
  ...
 <dataTable>
   ...
  <physical>
    ...
   <distribution>
    ...
    <access>
     ...
    </access>
   </distribution>
  </physical>
   ...
  </dataTable>
 </dataset>
</eml:eml>
```

1.4. Typing of <gRing> corrected

The content of the <gRing> element was retyped to make these nodes more usable. This element is generally analogous to the FGDC component for ring. This element should now contain a string comprised of a comma-delimited sequence of longitude and latitude values for vertex coordinates (in decimal degrees), as in the example below. For more information, see the normative documents for gRing in the **coverage module**.

```
..
 <gRing>-119.453,35.0 -125,37.5555 -122,40 -119.453,35.0 </gRing>
..
```

1.5. Entity Attributes: <bounds> minimum and maximum are of type xs:float

In EML 2.0.1, <bounds> elements were typed as xs:decimal and did not support scientific notation. The base data type was changed to 'xs:float' in EML 2.1.0 to accommodate both decimal and scientific notation while maintaining backward compatibility. Authors should keep in mind that there are still advantages to using decimal numbers for bounds, because the decimal data type maintains precision during storage while the floating point type does not. An alternative type, "precisionDecimal" (corresponding to the IEEE type "floating-point decimal"), may be available in the next version of XML Schema (i.e., v1.1, a working draft as of late 2008). It combines features of both the decimal and float types in that it supports the values and notation of a float, but is treated as decimal in arithmetic and storage. The typing of this element may be changed to this new type in a future release of EML. For more information, see the normative documentation for **NumericDomainType**.

In EML 2.1.0, bounds can be written as:

```
<attribute>
 ...
 <numericDomain>
  <numberType>real</numberType>
  <bounds>
   <minimum>0</minimum>
   <maximum>1.234E15</maximum>
 </bounds>
</numericDomain>
</attribute>
```

1.6. Geographic Coverage: <altitudeUnits> use Standard Units of LengthType

In EML 2.0.0 and 2.0.1, altitude units were typed as xs:string, and EML authors were instructed to include a vertical

datum along with the unit. In EML 2.1.0 this has been revised. Altitudes are now restricted to lengths in Standard Units (e.g. meter, foot, etc), and the datum should be included as part of the textual geographicDescription element. Document authors should note that including any additional content in the <altitudeUnits> element other than a length value, such as the datum, is not valid in EML 2.1.0. For a list of allowable units, see **the normative description for <altitudeUnits>**.

```
..
<boundingCoordinates>
 ...
 <boundingAltitudes>
  <altitudeMinimum>0</altitudeMinimum>
  <altitudeMaximum>120</altitudeMaximum>
  <altitudeUnits>meter</altitudeUnits>
 </boundingAltitudes>
</boundingCoordinates>
..
```

1.7. Geographic Coverage: Latitude and Longitude are type xs:decimal, with appropriate ranges

In EML 2.0.1, latitude and longitude values in <geographicCoverage> elements were typed as a xs:string. In EML 2.1.0 these values are restricted to decimal numbers with realistic ranges (-90 to 90, and -180 to 180, respectively). Fractions of a degree in minutes and seconds should be converted to decimal format, and strings denoting direction or hemisphere (e.g., 'S' or 'south') are not allowed. South latitudes and west longitudes must be indicated by a minus sign (-) in front of the coordinate, as in the example below. These constraints are consistent with the intended use of this field, which is to support mapping the general geographic coverage of EML resources. Authors should keep in mind that very specific descriptions of spatial data can be accommodated by EML modules dedicated to that purpose. More information on bounding coordinates can be found in the **normative technical documents**.

```
..
<boundingCoordinates>
 <westBoundingCoordinate>-120.2534</westBoundingCoordinate>
 <eastBoundingCoordinate>-119.7550</eastBoundingCoordinate>
 <northBoundingCoordinate>34.2231</northBoundingCoordinate>
 <southBoundingCoordinate>34.1231</southBoundingCoordinate>
</boundingCoordinates>
..
```

1.8. Element content must be non-empty

In EML 2.0.1, elements of the string data type were allowed to be empty or contain only whitespace. This feature was occasionally exploited as a work-around to force incomplete documents to validate in XML editors and the Metacat harvester, but this practice may cause problems in document parsing or for EML tools such as Kepler. In EML 2.1.0, string content is now typed as "NonEmptyString" and string entities are required to have minimal non-whitespace content. So, whereas the following content would have been allowed in EML 2.0.1:

```
...
  <mediumName> </mediumName>
...
```

or

```
...
  <attributeName/>
...
```

In EML 2.1.0, empty (or whitespace) content is not allowed. Actual content must be provided.

```
...
  <attributeName>approx. temperature</attributeName>
...
```

1.9. An offline resource has a minimum of one element required (<mediumName>)

In EML 2.0.1, an author could describe an offline data resource, but include no information about the resource's distribution. In EML-2.1.0, minimal content (one element) is now required.

In EML 2.0.1, the distribution tree for an offline resource could have ended with no content:

```
...
<distribution>
 <offline/>
</distribution>
...
```

In EML 2.1.0, the element <mediumName> is required:

```
...
<distribution>
 <offline>
  <mediumName>Atlas of Lake Erie Shorelines</mediumName>
 </offline>
</distribution>
...
```

1.10. Methods elements are standardized to <methods>

In EML 2.0.1, both "<method>" and "<methods>" elements were included in the schema, which caused confusion for some authors. In EML 2.1.0, instances of the MethodsType have been standardized to "methods".

In EML 2.0.1, this path existed:

```
...
/eml/dataset/dataTable/attribute/method/
...
```

In EML 2.1.0, this path is now properly constructed as:

```
...
/eml/dataset/dataTable/attribute/methods/
...
```

1.11. Elements for date-time have been standardized to <dateTime>

In EML 2.0.1, both "<datetime>" and "<dateTime>" elements were included, which caused confusion for some authors. In EML 2.1.0, these instances have been standardized to "dateTime".

In EML 2.0.1, this path existed:

```
...
/eml/dataset/dataTable/attribute/measurementScale/datetime/
...
```

In EML 2.1.0, this path is now properly constructed as:

```
...
/eml/dataset/dataTable/attribute/measurementScale/dateTime/
...
```

1.12. For journal articles, the elements <volume> and <pageRange> are now optional

Two elements describing journal articles in the literature schema (eml-literature.xsd), <volume> and <pageRange>, are now optional to permit articles-in-press to be described in EML.

1.13. A Citation may have an optional <contact> tree

Also in eml-literature.xsd, an optional <contact> tree has been added to permit a contact to be designated for a publication. For example, a contact could be provided for reprint requests.

1.14. New optional element (<onlineDescription>) for a description of an online resource

A new element, <onlineDescription>, was added to support providing a brief description of the content of an online element's child. This optional element is available for both resource-level and physical-level distribution nodes, and is typed as a NonEmptyString. One possible use for the description is to provide optional content for the HTML anchor element that accompanies a URL.

2. Converting EML documents from v2.0.0/1 to v2.1.0

2.1. About the EML conversion stylesheet

An XSL stylesheet is provided with the EML Utilities to convert valid EML 2.0-series documents to EML 2.1.0 (see **http://knb.ecoinformatics.org/software/eml/**). The stylesheet performs basic tasks to create a template EML 2.1.0 document (below). For more information, see the Utilities documentation.

1. Updates namespaces to eml-2.1.0 and stmml-1.1
2. Encloses XML markup within <additionalMetadata> sections in <metadata> tags
3. Renames elements whose spelling has changed (<method> and <datetime>)
4. Copies access trees from <additionalMetadata> to other parts of the document (for common constructs)
5. Optionally replaces the content of the "packageId" attribute on the root element, <eml:eml>, using a parameter

2.2. Validity of new EML 2.1.0 documents

Because of the flexibility allowed in EML, the stylesheet may encounter EML 2.0.1 structures that cannot be transformed or that may result in invalid EML 2.1.0 after processing. For example, by design <additionalMetadata> sections are parsed laxly, and so it is possible for their content in EML-2.0.0/1 to contain <access> trees which are invalid. Also, the content of several elements has been more tightly constrained in EML 2.1.0 (e.g., latitude and longitude), and data types are not detectable by a stylesheet. Document authors are advised to check the validity of their new EML 2.1.0 after transformation. EML instance documents can be validated in these ways:

1. With the **online EML Parser**. The online parser will validate all versions of EML.
2. Using the Parser that comes with EML. To execute it, change into the 'lib' directory of the EML release and run the 'runEMLParser' script passing your EML instance file as a parameter. The script performs two actions: it checks the validity of references and id attributes, and it validates the document against the EML 2.1 schema. The EML parser included with the distribution is capable of checking only EML 2.1.0 documents, and *cannot* be used to validate earlier versions (e.g., EML 2.0.1).
3. If you are planning to contribute your EML 2.1.0 document to a Metacat repository, note that the Metacat servlet checks all versions of incoming EML for validity as part of the insertion process.

Ecological Metadata Language (EML) Specification

Table of Contents

List of Examples

Chapter 1. Preface

Table of Contents

1.1. Introduction

The Ecological Metadata Language (EML) is a metadata standard developed by the ecology discipline and for the ecology discipline. It is based on prior work done by the Ecological Society of America and associated efforts (Michener et al., 1997, Ecological Applications). EML is implemented as a series of XML document types that can by used in a modular and extensible manner to document ecological data. Each EML module is designed to describe one logical part of the total metadata that should be included with any ecological dataset.

1.2. Purpose Statement

To provide the ecological community with an extensible, flexible, metadata standard for use in data analysis and archiving that will allow automated machine processing, searching and retrieval.

1.3. Features

The architecture of EML was designed to serve the needs of the ecological community, and has benefitted from previous work in other related metadata languages. EML has adopted the strengths of many of these languages, but also addresses a number of short-comings that have proved to inhibit the automated processing and integration of dataset resources via their metadata.

The following list represents some of the features of EML:

- Modularity: EML was designed as a collection of modules rather than one large standard to facilitate future growth of the language in both breadth and depth. By implementing EML with an extensible architecture, groups may choose which of the core modules are pertinent to describing their data, literature, and software resources. Also, if EML falls short in a particular area, it may be extended by creating a new module that describes the resource (e.g. a detailed soils metadata profile that extends eml-dataset). The intent is to provide a common set of core modules for information exchange, but to allow for future customizations of the language without the need of going through a

lengthy 'approval' process.

- Detailed Structure: EML strives to balance the tradeoff of too much detail with enough detail to enable advanced services in terms of processing data through the parsing of accompanied metadata. Therefore, a driving question throughout the design was: 'Will this particular piece of information be machine-processed, just human readable, or both?' Information was then broken down into more highly structured elements when the answer involved machine processing.
- Compatibility: EML adopts much of it's syntax from the other metadata standards that have evolved from the expertise of groups in other disciplines. Whenever possible, EML adopted entire trees of information in order to facilitate conversion of EML documents into other metadata languages. EML was designed with the following standards in mind: Dublin Core Metadata Initiative, the Content Standard for Digital Geospatial Metadata (CSDGM from the US geological Survey's Federal Geographic Data Committee (FGDC)), the Biological Profile of the CSDGM (from the National Biological Information Infrastructure), the International Standards Organization's Geographic Information Standard (ISO 19115), the ISO 8601 Date and Time Standard, the OpenGIS Consortiums's Geography Markup Language (GML), the Scientific, Technical, and Medical Markup Language (STMML), and the Extensible Scientific Interchange Language (XSIL).
- Strong Typing: EML is implemented in an Extensible Markup Language (XML) known as **XML Schema**, which is a language that defines the rules that govern the EML syntax. XML Schema is an internet recommendation from the **World Wide Web Consortium**, and so a metadata document that is said to comply with the syntax of EML will structurally meet the criteria defined in the XML Schema documents for EML. Over and above the structure (what elements can be nested within others, cardinality, etc.), XML Schema provides the ability to use strong data typing within elements. This allows for finer validation of the contents of the element, not just it's structure. For instance, an element may be of type 'date', and so the value that is inserted in the field will be checked against XML Schema's definition of a date. Traditionally, XML documents (including previous versions of EML) have been validated against Document Type Definitions (DTDs), which do not provide a means to employ strong validation on field values through typing.
- There is a distinction between the content model (i.e. the concepts behind the structure of a document - which fields go where, cardinality, etc.) and the syntactic implementation of that model (the technology used to express the concepts defined in the content model). The normative sections below define the content model and the XML Schema documents distributed with EML define the syntactic implementation. For the foreseeable future, XML Schema will be the syntactic specification, although it may change later.

Chapter 2. Overview of EML modules and their use

Table of Contents

2.1. Module Overview Foreword

The following section briefly describes each EML module and how they are logically designed in order to document ecological resources. Some of the modules are dependent on others, while others may be used as stand-alone descriptions. This section describes the modules using a "top down" approach, starting from the top-level eml wrapper module, followed by modules of increasing detail. However, there are modules that may be used at many levels, such as eml-access. These modules are described when it is appropriate.

2.2. Root-level structure

2.2.1. The eml module - A metadata container

The eml module is a wrapper container that allows the inclusion of any metadata content in a single EML document. The eml module is used as a container to hold structured descriptions of ecological resources. In EML, the definition of a resource comes from the ***The Dublin Core Metadata Initiative*** , which describes a general element set used to describe "networked digital resources". The top-level structure of EML has been designed to be compatible with the Dublin Core syntax. In general, dataset resources, literature resources, software resources, and protocol resources comprise the list of information that may be described in EML. EML is largely designed to describe digital resources, however, it may also be used to describe non-digital resources such as paper maps and other non-digital media. *In EML, the definition of a "Data Package" is the combination of both the data and metadata for a resource.* So, data packages are built by using the <eml> wrapper, which will include all of the metadata, and optionally the data (or references to them). All EML packages must begin with the <eml> tag and end with the </eml> tag.

The eml module may be extended to describe other resources by means of its optional sub-field, <additionalMetadata>. This field is largely reserved for the inclusion of metadata that may be highly discipline specific and not covered in this version of EML, or it may be used to internally extend fields within the EML standard.

2.2.2. The eml-resource module - Base information for all resources

The eml-resource module contains general information that describes dataset resources, literature resources, protocol resources, and software resources. Each of the above four types of resources share a common set of information, but also have information that is unique to that particular resource type. Each resource type uses the eml-resource module to document the information common to all resources, but then extend eml-resource with modules that are specific to that particular resource type. For instance, all resources have creators, titles, and perhaps keywords, but only the dataset resource would have a "data table" within it. Likewise, a literature resource may have an "ISBN" number associated with it, whereas the other resource types would not.

The eml-resource module is exclusively used by other modules, and is therefore not a stand-alone module.

2.3. Top-level resources

The following four modules are used to describe separate resources: datasets, literature, software, and protocols. However, note that the dataset module makes use of the other top-level modules by importing them at different levels. For instance, a dataset may have been produced using a particular protocol, and that protocol may come from a protocol document in a library of protocols. Likewise, citations are used throughout the top-level resource modules by importing the literature module.

2.3.1. The eml-dataset module - Dataset specific information

The eml-dataset module contains general information that describes dataset resources. It is intended to provide overview information about the dataset: broad information such as the title, abstract, keywords, contacts, maintenance history, purpose, and distribution of the data themselves. The eml-dataset module also imports many other modules that are used to describe the dataset in fine detail. Specifically, it uses the eml-methods module to describe methodology used in collecting or processing the dataset, the eml-project module to describe the overarching research context and experimental design, the eml-access module to define access control rules for the data and metadata, and the eml-entity module to provide detailed information about the logical structure of the dataset. A dataset can be (and often is) composed of a series of data entities (tables) that are linked together by particular integrity constraints.

The eml-dataset module, like other modules, may be "referenced" via the <references> tag. This allows a dataset to be described once, and then used as a reference in other locations within the EML document via its ID.

2.3.2. The eml-literature module - Citation specific information

The eml-literature module contains information that describes literature resources. It is intended to provide overview information about the literature citation, including title, abstract, keywords, and contacts. Citation types follow the conventions laid out by **EndNote**, and there is an attempt to represent a compatible subset of the EndNote citation types. These citation types include: article, book, chapter, edited book, manuscript, report, thesis, conference proceedings, personal communication, map, generic, audio visual, and presentation. The "generic" citation type would be used when one of the other types will not work.

The eml-literature module, like other modules, may be "referenced" via the <references> tag. This allows a citation to be described once, and then used as a reference in other locations within the EML document via its ID.

2.3.3. The eml-software module - Software specific information

The eml-software module contains general information that describes software resources. This module is intended to fully document software that is needed in order to view a resource (such as a dataset) or to process a dataset. The software module is also imported into the eml-methods module in order to document what software was used to process or perform quality control procedures on a dataset.

The eml-software module, like other modules, may be "referenced" via the <references> tag. This allows a software resource to be described once, and then used as a reference in other locations within the EML document via its ID.

2.3.4. The eml-protocol module - Research protocol specific information

The EML Protocol Module is used to define abstract, prescriptive procedures for generating or processing data. Conceptually, a protocol is a standardized method.

Eml-protocol resembles eml-methods; however, eml-methods is descriptive (often written in the declarative mood: "I took five subsamples...") whereas eml-protocol is prescriptive (often written in the imperative mood: "Take five subsamples..."). A protocol may have versions, whereas methods (as used in eml-methods) should not.

2.4. Supporting Modules - Adding detail to top-level resources

The following six modules are used to qualify the resources being described in more detail. They are used to describe access control rules, distribution of the metadata and data themselves, parties associated with the resource, the geographic, temporal, and taxonomic extents of the resource, the overall research context of the resource, and detailed methodology used for creating the resource. Some of these modules are imported directly into the top-level resource modules, often in many locations in order to limit the scope of the description. For instance, the eml-coverage module may be used for a particular column of a dataset, rather than the entire dataset as a whole.

2.4.1. The eml-access module - Access control rules for resources

The eml-access module describes the level of access that is to be allowed or denied to a resource for a particular user or group of users, and can be described independently for metadata and data. The eml-access module uses a reference to a particular authentication system to determine the set of principals (users or groups) that can be specified in the access rules. The special principal 'public' can be used to indicate that any user or group has access permission, thereby making it easier to specify that anonymous access is allowed.

There are two mechanisms for including access control via the eml-access module:

1. 1) The top-level "eml" element may have an optional <access> element that is used to establish the default access control for the entire EML package. If this access element is omitted from the document, then the package submitter should be given full access to the package but all other users should be denied all access. To allow the package to be publicly viewable, the EML author must explicitly include a rule stating so.
2. 2) Exceptions for particular entity-level components of the package can be controlled at a finer grain by using an access description in that entity's physical/distribution tree. When access control rules are specified at this level, they apply only to the data in the parent distribution element, and not to the metadata. Thus, it will control access to the content of the <inline> element, as well as resources that are referenced by the <online/url> and <online/connection> paths. These exceptions to access for particular data resources are applied after the default access rules at the package-level have been applied, so they effectively override the default rules when they overlap.

In previous versions of EML access rules for entity-level distribution were contained in <additionalMetadata> sections and referenced via the <describes> tag. Although in theory these could have referenced any node, in application such node-level access control is problematic. Since the most common uses of access control rules were to limit access to specific data entities, the access tree has been placed there explicitly in EML 2.1.0.

Access is specified with a choice of child elements, either <allow> or <deny>. Within these rules, values can be assigned for each <principal> using the <permission> element. Users given "read" permission can view the resource; "write" allows changes to the resource excluding changes to the access rules; "changePermission" includes "write" plus the changing of access rules. Users allowed "all" permissions; may do all of the above.

An example is given below, with non-critical sections deleted:

```
<eml>
    <access
        authSystem="ldap://ldap.ecoinformatics.org:389/dc=ecoinformatics,dc=org"
        order="allowFirst">
      <allow>
        <principal>uid=alice,o=NASA,dc=ecoinformatics,dc=org</principal>
        <permission>read</permission>
        <permission>write</permission>
      <allow>
    </access>
    <dataset>
    ...
    ...
    <dataTable id="entity123">
    ...
      <physical>
      ...
        <distribution>
        ...
          <access id="access123"
          authSystem="ldap://ldap.ecoinformatics.org:389/dc=ecoinformatics,dc=org"
          order="allowFirst">
            <deny>
              <principal>uid=alice,o=NASA,dc=ecoinformatics,dc=org</principal>
              <permission>write</permission>
          </deny>
        </access>
       </distribution>
     </physical>
    </dataTable>
    <dataTable id="entity234">
      ...
      <physical>
      ...
        <distribution>
          ...
          <access>
            <references>access123</references>
          </access>
        </distribution>
      </physical>
    </dataTable>
    ...
  </dataset>
<eml>
```

In this example, the overall default access is to allow the user=alice (but no one else) to read and write all metadata and

data. However, under "entity123" and "entity234", there is an additional rule saying that user=alice does not have write permission. The net effect is that Alice can read and make changes to the metadata, but cannot make changes to the two data entities. In addition, Alice cannot change these access rules; although the submitter can.

This example also shows how the eml-access module, like other modules, may be "referenced" via the <references> tag. This allows an access control document to be described once, and then used as a reference in other locations within the EML document via its ID.

In summary, access rules can be applied in two places in an eml document. Default access rules are established in the top <access> element for the main eml document (e.g., "/eml/access"). These default rules can be overridden for particular data entities by adding additional <access> elements in the physical/distribution trees of those entities.

2.4.2. The eml-physical module - Physical file format

The eml-physical module describes the external and internal physical characteristics of a data object as well as the information required for its distribution. Examples of the external physical characteristics of a data object would be the filename, size, compression, encoding methods, and authentication of a file or byte stream. Internal physical characteristics describe the format of the data object being described. Both named binary or otherwise proprietary formats can be cited (e.g., Microsoft Access 2000), or text formats can be precisely described (e.g., ASCII text delimited with commas). For these text formats, it also includes the information needed to parse the data object to extract the entity and its attributes from the data object. Distribution information describes how to retrieve the data object. The retrieval information can be either online (e.g., a URL or other connection information) or offline (e.g., a data object residing on an archival tape).

The eml-physical module, like other modules, may be "referenced" via the <references> tag. This allows a physical document to be described once, and then used as a reference in other locations within the EML document via its ID.

2.4.3. The eml-party module - People and organization information

The eml-party module describes a responsible party and is typically used to name the creator of a resource or metadata document. A responsible party may be an individual person, an organization or a named position within an organization. The eml-party module contains detailed contact information. It is used throughout the other EML modules where detailed contact information is needed.

The eml-party module, like other modules, may be "referenced" via the <references> tag. This allows a party to be described once, and then used as a reference in other locations within the EML document via its ID.

2.4.4. The eml-coverage module - Geographic, temporal, and taxonomic extents of resources

The eml-coverage module contains fields for describing the coverage of a resource in terms of time, space, and taxonomy. These coverages (temporal, spatial, and taxonomic) represent the extent of applicability of the resource in those domains. The Geographic coverage section allows for 2 means of expressing coverage on the surface of the earth: 1) via a set of bounding coordinates that define the North, South, East and West points in a rectangular area, optionally including a bounding altitude, and 2) using a G-Ring polygon definition, where an irregularly shaped area may be defined using a ordered list of latitude/longitude coordinates. A G-Ring may also include an "inner G-Ring" that defines one or more "cut-outs" in the area, i.e. the donut hole concept.

The temporal coverage section allows for the definition of either a single date or time, or a range of dates or times. These may be expressed as a calendar date according to the ISO 8601 Date and Time Specification, or by using an alternate time scale, such as the geologic time scale. Currently, EML does not have specific fields to indicate that a data resource may be "ongoing." Two examples are data tables that are planned to be appended in the future, or resources with complex connection definitions (such as to a database) which may return data in real time. It is important that EML be able to handle data from both the "producer" and "consumer" points of view, although currently the temporal coverage modules are designed for the latter. There is no universally acceptable recommendation for describing "ongoing" data within EML. Some groups have chosen to use the <alternateTimeScale> node for the end date, with a value of "ongoing," although this practice is not endorsed by the EML authors. A better solution could be to use very general content for the endDate (such as only the current year) so that the data are accurately described, and searches return datasets as expected. A future version of EML will accommodate such data types with coverage elements specific to their needs.

The taxonomic coverage section allows for detailed description of the taxonomic extent of the dataset or resource. The taxonomic classification consists of a recursive set of taxon rank names, their values, and their common names. This construct allows for a taxonomic hierarchy to be built to show the level of identification (e.g. Rank Name = Kingdom, Rank Value = Animalia, Common Name = Animals, and so on down the hierarchy.) The taxonomic coverage module also allows

for the definition of the classification system in cases where alternative systems are used.

The eml-coverage module, like other modules, may be "referenced" via the <references> tag. This allows the coverage extent to be described once, and then used as a reference in other locations within the EML document via its ID.

2.4.5. The eml-project module - Research context information for resources

The eml-project module describes the research context in which the dataset was created, including descriptions of over-all motivations and goals, funding, personnel, description of the study area etc. This is also the module to describe the design of the project: the scientific questions being asked, the architecture of the design, etc. This module is used to place the dataset that is being documented into its larger research context.

The eml-project module, like other modules, may be "referenced" via the <references> tag. This allows a research project to be described once, and then used as a reference in other locations within the EML document via its ID.

2.4.6. The eml-methods module - Methodological information for resources

The eml-methods module describes the methods followed in the creation of the dataset, including description of field, laboratory and processing steps, sampling methods and units, quality control procedures. The eml-methods module is used to describe the *actual* procedures that are used in the creation or the subsequent processing of a dataset. Likewise, eml-methods is used to describe processes that have been used to define / improve the quality of a data file, or to identify potential problems with the data file. Note that the eml-protocol module is intended to be used to document a *prescribed* procedure, whereas the eml-method module is used to describe procedures that *were actually performed*. The distinction is that the use of the term "protocol" is used in the "prescriptive" sense, and the term "method" is used in the "descriptive" sense. This distinction allows managers to build a protocol library of well-known, established protocols (procedures), but also document what procedure was truly performed in relation to the established protocol. The method may have diverged from the protocol purposefully, or perhaps incidentally, but the procedural lineage is still preserved and understandable.

2.5. Data organization - Modules describing dataset structures

The following three modules are used to document the logical layout of a dataset. Many datasets are comprised of multiple entities (e.g. a series of tabular data files, or a set of GIS features, or a number of tables in a relational database). Each entity within a dataset may contain one or more attributes (e.g. multiple columns in a data file, multiple attributes of a GIS feature, or multiple columns of a database table). Lastly, there may be both simple or complex relationships among the entities within a dataset. The relationships, or the constraints that are to be enforced in the dataset, are described using the eml-constraint module. All entities share a common set of information (described using eml-entity), but some discipline specific entities have characteristics that are unique to that entity type. Therefore, the eml-entity module is extended for each of these types (dataTable, spatialRaster, spatialVector, etc...) which are described in the next section.

2.5.1. The eml-entity module - Entity level information within datasets

The eml-entity module defines the logical characteristics of each entity in the dataset. Entities are usually tables of data (eml-dataTable). Data tables may be ascii text files, relational database tables, spreadsheets or other type of tabular data with a fixed logical structure. Related to data tables are views (eml-view) and stored procedures (eml-storedProcedure). Views and stored procedures are produced by an RDBMS or related system. Other types of data such as: raster (eml-spatialRaster), vector (eml-spatialVector) or spatialReference image data are also data entities. An otherEntity element would be used to describe types of entities that are not described by any other entity type.

The eml-entity module, like other modules, may be "referenced" via the <references> tag. This allows an entity document to be described once, and then used as a reference in other locations within the EML document via its ID.

2.5.2. The eml-attribute module - Attribute level information

within dataset entities

The eml-attribute module describes all attributes (variables) in a data entity: dataTable, spatialRaster, spatialVector, storedProcedure, view or otherEntity. The description includes the name and definition of each attribute, its domain, definitions of coded values, and other pertinent information. Two structures exist in this module: 1. attribute is used to define a single attribute; 2. attributeList is used to define a list of attributes that go together in some logical way.

The eml-attribute module, like other modules, may be "referenced" via the <references> tag. This allows an attribute document to be described once, and then used as a reference in other locations within the EML document via its ID.

2.5.2.1. Philosophy of Attribute Units

The concept of "unit" represents one of the most fundamental categories of metadata. The classic example of data entropy is the case in which a reported numeric value loses meaning due to lack of associated units. Much of Ecology is driven by measurement, and most measurements are inherently comparative. Good data description requires a representation of the basis for comparison, i.e., the unit. In modeling the attribute element, the authors of EML drew inspiration from the **NIST Reference on Constants, Units, and Uncertainty**. This document defines a unit as "a particular physical quantity, defined and adopted by convention, with which other particular quantities of the same kind are compared to express their value." The authors of the EML 2.0 specification (hereafter "the authors") decided to make the unit element required, wherever possible.

Units may also be one of the most problematic categories of metadata. For instance, there are many candidate attributes that clearly have no units, such as named places and letter grades. There are other candidate attributes for which units are difficult to identify, despite some suspicion that they should exist (e.g. pH, dates, times). In still other cases, units may be meaningful, but apparently absent due to dimensional analysis (e.g. grams of carbon per gram of soil). The relationship between units and dimensions likewise is not completely clear.

The authors decided to sharpen the model of attribute by nesting unit under measurementScale. Measurement Scale is a data typology, borrowed from Statistics, that was introduced in the 1940's. Under the adopted model, attributes are classified as nominal, ordinal, interval, and ratio. Though widely criticized, this classification is well-known and provides at least first-order utility in EML. For example, nesting unit under measurementScale allows EML to prevent its meaningless inclusion for categorical data -- an approach judged superior to making unit universally required or universally optional.

The sharpening of the attribute model allowed the elimination of the unit type "undefined" from the standard unit dictionary (see eml-unitDictionary.xml). It seemed self-defeating to require the unit element exactly where appropriate, yet still allow its content to be undefined. An attribute that requires a unit definition is malformed until one is provided. The unit type "dimensionless" is preserved, however. In EML 2.0, it is synonymous with "unitless" and represents the case in which units cannot be associated with an attribute for some reason, despite the proper classification of that attribute as interval or ratio. Dimensionless may itself be an anomaly arising from the limitations of the adopted measurement scale typology.

Closely related to the concept of unit is the concept of attribute domain. The authors decided that a well-formed description of an attribute must include some indication of the set of possible values for that attribute. The set of possible values is useful, perhaps necessary, for interpreting any particular observed value. While universally required, attribute domain has different forms, depending on the associated measurement scale.

The element storageType has an obvious relationship to domain. It gives some indication of the range of possible values of an attribute, and also gives some (potentially critical) operability information about the way the attribute is represented or construed in the local storage system. The storageType element seems to fall in a gray area between the logical and physical aspects of stored data. Neither comfortable with eliminating it nor with making it required, the authors left it available but optional under attribute. In addition, it is repeatable so that different storage types can be provided for various systems (e.g., different databases might use different types for columns, even though the domain of the attribute is the same regardless of which database is used).

Attributes representing dates, times, or combinations thereof (hereafter "dateTime") were the most difficult to model in EML. Is dateTime of type interval or ordinal? Does it have units or not? Strong cases can be made on each side of the issue. The confusion may reflect the limitations of the measurement scale typology. The final resolution of the dateTime model is probably somewhat arbitrary. There was clearly a need, however, to allow for the interoperability of dateTime formats. EML 2.0 tries to provide an unambiguous mechanism for describing the format of dateTime values by providing a separate category for date and time values. This "dateTime" measurement scale allows users to explicitly label attributes that contain Gregorian date and time values, and allows them to provide the information needed to parse these values into their appropriate components (e.g., days, months, years)./

2.5.3. The eml-constraint module - Relationships among and within dataset entities

The eml-constraint schema defines the integrity constraints between entities (e.g., data tables) as they would be maintained in a relational management system. These constraints include primary key constraints, foreign key constraints, unique key constraints, check constraints, and not null constraints, among potential others.

2.6. Entity types - Detailed information for discipline specific entities

The following six modules are used to describe a number of common types of entities found in datasets. Each entity type uses the eml-entity module elements as it's base set of elements, but then extends the base with entity-specific elements. Note that the eml-spatialReference module is not an entity type, but is rather a common set of elements used to describe spatial reference systems in both eml-spatialRaster and eml-spatialVector. It is described here in relation to those two modules.

2.6.1. The eml-dataTable module - Logical information about data table entities

The eml-dataTable module is used to describe the logical characteristics of each tabular set of information in a dataset. A series of comma-separated text files may be considered a dataset, and each file would subsequently be considered a dataTable entity within the dataset. Since the eml-dataTable module extends the eml-entity module, it uses all of the common entity elements to describe the table, along with a few elements specific to just data table entities. The eml-dataTable module allows for the description of each attribute (column/field/variable) within the data table through the use of the eml-attribute module. Likewise, there are fields used to describe the physical distribution of the data table, its overall coverage, the methodology used in creating the data, and other logical structure information such as its orientation, case sensitivity, etc.

2.6.2. The eml-spatialRaster module - Logical information about regularly gridded geospatial image data

The eml-spatialRaster module allows for the description of entities composed of rectangular grids of data values that are usually georeferenced to a portion of the earth's surface. Specific attributes of a spatial raster can be documented here including the spatial organization of the raster cells, the cell data values, and if derived via imaging sensors, characteristics about the image and its individual bands.

2.6.3. The eml-spatialVector module - Logical information about non-gridded geospatial image data

The eml-spatialVector module allows for the description of spatial objects in a GIS system that are not defined in a regularly gridded pattern. These geometries include points and vectors and the relationships among them. Specific attributes of a spatial vector can be documented here including the vector's geometry type, count and topology level.

2.6.4. Schema for validating spatial referencing descriptions.

This module defines both projected and unprojected coordinate systems for referencing the spatial coordinates of a dataset to the earth. The schema is based on that used by Environmental Systems Research Inc (ESRI) for its .prj file format. EML provides a library of pre-defined coordinate systems that may be referred to by name in the horizCoordSysName element. A custom projection may be defined using this schema for any projection that does not appear in this dictionary.

2.6.5. The eml-storedProcedure module - Data tables resulting from procedures stored in a database

The storedProcedure module is meant to capture information on procedures that produce data output in the form of a data table. In an RDBMS one can code complex queries and transactions into stored procedures and then invoke them directly from front-end applications. It allows the optional description of any parameters that are expected to be passed to the procedure when it is called.

2.6.6. The eml-view module - Data tables resulting from a database query

The eml-view module describes a view from a database management system. A view is a query statement that is stored as a database object and executed each time the view is called.

2.7. Utility modules - Metadata documentation enhancements

The following modules are used to highlight the information being documented in each of the above modules where prose may be needed to convey the critical metadata. The eml-text module provides a number of text-based constructs to enhance a document (including sections, paragraphs, lists, subscript, superscript, emphasis, etc.)

2.7.1. The eml-text module - Text field formatting

The eml-text module is a wrapper container that allows general text descriptions to be used within the various modules of eml. It can include either structured or unstructured text blocks. It isn't really appropriate to use this module outside of the context of a parent module, because the parent module determines the appropriate context to which this text description applies. The eml-text module allows one to provide structure to a text description in order to convey concepts such as sections (paragraphs), hierarchy (ordered and unordered lists), emphasis (bold, superscript, subscript) etc. The structured elements are a subset of **DocBook** so the predefined DocBook stylesheets can be used to style EML fields that implement this module.

2.7.2. Dependency Chart

The multiple modules in EML all depend on each other in complex ways. To easily see these dependencies see the **EML Dependency Chart.**

Chapter 3. Technical Architecture (Normative)

Table of Contents

3.1. Introduction

This section explains the rules of EML. There are some rules that cannot be written directly into the XML Schemas nor enforced by an XML parser. These are guidelines that every EML package must follow in order for it to be considered EML compliant.

3.2. Module Structure

Each EML module, with the exception of "eml" itself, has a top level choice between the structured content of that modules or a "references" field. This enables the reuse of content previously defined elsewhere in the document. Methods for defining and referencing content are described in the **next** section

3.3. Reusable Content

EML allows the reuse of previously defined structured content (DOM sub-trees) through the use of key/keyRef type references. In order for an EML package to remain cohesive and to allow for the cross platform compatibility of packages, the following rules with respect to packaging must be followed.

- An ID is required on the eml root element.
- IDs are optional on all other elements.
- If an ID is not provided, that content must be interpreted as representing a distinct object.
- If an ID is provided for content then that content is distinct from all other content except for that content that references its ID.
- If a user wants to reuse content to indicate the repetition of an object, a reference must be used. Two identical ids with the same system attribute cannot exist in a single document.
- "Document" scope is defined as identifiers unique only to a single instance document (if a document does not have a system attribute or if scope is set to 'document' then all IDs are defined as distinct content).
- "System" scope is defined as identifiers unique to an entire data management system (if two documents share a system string, then any IDs in those two documents that are identical refer to the same object).
- If an element references another element, it must not have an ID itself. The system attribute must have the same value in both the target and referencing elements or it must be absent in both.
- All EML packages must have the 'eml' module as the root.
- The system and scope attribute are always optional except for at the 'eml' module where the scope attribute is fixed as 'system'. The scope attribute defaults to 'document' for all other modules.

3.3.1. ID and Scope Examples

3.3.1.1. EML Parser

Because some of these rules cannot be enforced in XML-Schema, we have written a parser which checks the validity of the references and IDs used in your document. This parser is included with the 2.1.0 release of EML. To run the parser, you must have Java 1.3.1 or higher. To execute it change into the lib directory of the release and run the 'runEMLParser' script passing your EML instance file as a parameter. There is also an **online version** of this parser which is publicly accessible. The online parser will both validate your XML document against the schema as well as check the integrity of your references.

3.3.1.2. Example Documents

Example 3.1. Invalid EML due to duplicate identifiers

```
<?xml version="1.0"?>
<eml:eml
    packageId="eml.1.1" system="knb"
    xmlns:eml="eml://ecoinformatics.org/eml-2.1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="eml://ecoinformatics.org/eml-2.1.0 eml.xsd">

  <dataset id="ds.1">
    <title>Sample Dataset Description</title>
    <!-- the two creators have the same id.  this should be an error-->
    <creator id="23445" scope="document">
      <individualName>
        <surName>Smith</surName>
      </individualName>
    </creator>
    <creator id="23445" scope="document">
      <individualName>
        <surName>Myer</surName>
      </individualName>
    </creator>
    ...
  </dataset>
</eml:eml>
```

This instance document is invalid because both creator elements have the same id. No two elements can have the same string as an id.

Example 3.2. Invalid EML due to a non-existent reference

```
<?xml version="1.0"?>
<eml:eml
    packageId="eml.1.1" system="knb"
    xmlns:eml="eml://ecoinformatics.org/eml-2.1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="eml://ecoinformatics.org/eml-2.1.0 eml.xsd">

  <dataset id="ds.1">
    <title>Sample Dataset Description</title>
    <creator id="23445" scope="document">
      <individualName>
        <surName>Smith</surName>
      </individualName>
    </creator>
    <creator id="23446" scope="document">
      <individualName>
        <surName>Myer</surName>
      </individualName>
    </creator>
    ...
    <contact>
      <references>23447</references>
    </contact>
  </dataset>
</eml:eml>
```

This instance document is invalid because the contact element references an id that does not exist. Any referenced id must exist.

Example 3.3. Invalid EML due to a conflicting id attribute and a <references> element

```
<?xml version="1.0"?>
<eml:eml
    packageId="eml.1.1" system="knb"
    xmlns:eml="eml://ecoinformatics.org/eml-2.1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="eml://ecoinformatics.org/eml-2.1.0 eml.xsd">

  <dataset id="ds.1">
    <title>Sample Dataset Description</title>
    <creator id="23445" scope="document">
      <individualName>
        <surName>Smith</surName>
      </individualName>
    </creator>
    <creator id="23446" scope="document">
      <individualName>
        <surName>Meyer</surName>
      </individualName>
    </creator>
    ...
    <contact id="522">
      <references>23445</references>
    </contact>
  </dataset>
</eml:eml>
```

This instance document is invalid because the contact element both references another element and has an id itself. If an element references another element, it may not have an id. This prevents circular references.

Example 3.4. A valid EML document

```
<?xml version="1.0"?>
<eml:eml
    packageId="eml.1.1" system="knb"
    xmlns:eml="eml://ecoinformatics.org/eml-2.1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="eml://ecoinformatics.org/eml-2.1.0 eml.xsd">

  <dataset id="ds.1">
    <title>Sample Dataset Description</title>
    <creator id="23445" scope="document">
      <individualName>
        <surName>Smith</surName>
      </individualName>
    </creator>
    <creator id="23446" scope="document">
      <individualName>
        <surName>Smith</surName>
```

```
      </individualName>
    </creator>
    ...
    <contact>
      <references>23446</references>
    </contact>
    <contact>
      <references>23445</references>
    </contact>
  </dataset>
</eml:eml>
```

This instance document is valid. Each contact is referencing one of the creators above and all the ids are unique.

Chapter 4. Module Descriptions (Normative)

Table of Contents

4.1. eml

Normative technical docs for **eml**

4.2. eml-access

Normative technical docs for **eml-access**

4.3. eml-attribute

Normative technical docs for **eml-attribute**

4.4. eml-constraint

Normative technical docs for **eml-constraint**

4.5. eml-coverage

Normative technical docs for **eml-coverage**

4.6. eml-dataset

Normative technical docs for **eml-dataset**

4.7. eml-dataTable

Normative technical docs for **eml-dataTable**

4.8. eml-entity

Normative technical docs for **eml-entity**

4.9. eml-literature

Normative technical docs for **eml-literature**

4.10. eml-methods

Normative technical docs for **eml-methods**

4.11. eml-party

Normative technical docs for **eml-party**

4.12. eml-physical

Normative technical docs for **eml-physical**

4.13. eml-project

Normative technical docs for **eml-project**

4.14. eml-protocol

Normative technical docs for **eml-protocol**

4.15. eml-resource

Normative technical docs for **eml-resource**

4.16. eml-software

Normative technical docs for **eml-software**

4.17. eml-spatialRaster

Normative technical docs for **eml-spatialRaster**

4.18. eml-spatialReference

Normative technical docs for **eml-spatialReference**

4.19. eml-spatialVector

Normative technical docs for **eml-spatialVector**

4.20. eml-storedProcedure

Normative technical docs for **eml-storedProcedure**

4.21. eml-text

Normative technical docs for **eml-text**

4.22. eml-unitTypeDefinitions

Normative technical docs for **eml-unitTypeDefinitions**

4.23. eml-view

Normative technical docs for **eml-view**

Index

A

D

P

spatialRaster-eml-dataset
spatialRaster-eml-spatialRaster
spatialReference-eml-spatialRaster
spatialReference-eml-spatialReference
spatialReference-eml-spatialVector
spatialSamplingUnits-eml-methods
spatialVector-eml-dataset
spatialVector-eml-spatialVector
specimen-eml-coverage
spheroid-eml-spatialReference
standardUnit-eml-attribute
storageType-eml-attribute
storedProcedure-eml-dataset
storedProcedure-eml-storedProcedure
studyAreaDescription-eml-project
studyExtent-eml-methods
subscript-eml-text
subscript-eml-text
subStep-eml-methods
superscript-eml-text
superscript-eml-text
surName-eml-party

T

taxonomicClassification-eml-coverage
taxonomicClassification-eml-coverage
taxonomicCompleteness-eml-coverage
taxonomicCoverage-eml-coverage
taxonomicProcedures-eml-coverage
taxonomicSystem-eml-coverage
taxonRankName-eml-coverage
taxonRankValue-eml-coverage
temporalCoverage-eml-coverage
text-eml-text
textDelimited-eml-physical
textDomain-eml-attribute
textFixed-eml-physical
textFormat-eml-physical
thesis-eml-literature
time-eml-coverage
timeScaleAgeEstimate-eml-coverage
timeScaleAgeExplanation-eml-coverage
timeScaleAgeUncertainty-eml-coverage
timeScaleCitation-eml-coverage
timeScaleName-eml-coverage
title-eml-project
title-eml-resource
title-eml-text
toneGradation-eml-spatialRaster
topologyLevel-eml-spatialVector
totalFigures-eml-literature
totalFigures-eml-literature
totalPages-eml-literature
totalPages-eml-literature
totalPages-eml-literature
totalPages-eml-literature
totalPages-eml-literature
totalrowbytes-eml-physical
totalTables-eml-literature
totalTables-eml-literature
triangulationIndicator-eml-spatialRaster

U

ulink-eml-text
uniqueKey-eml-constraint
unit-eml-attribute
unit-eml-attribute
unit-eml-spatialReference

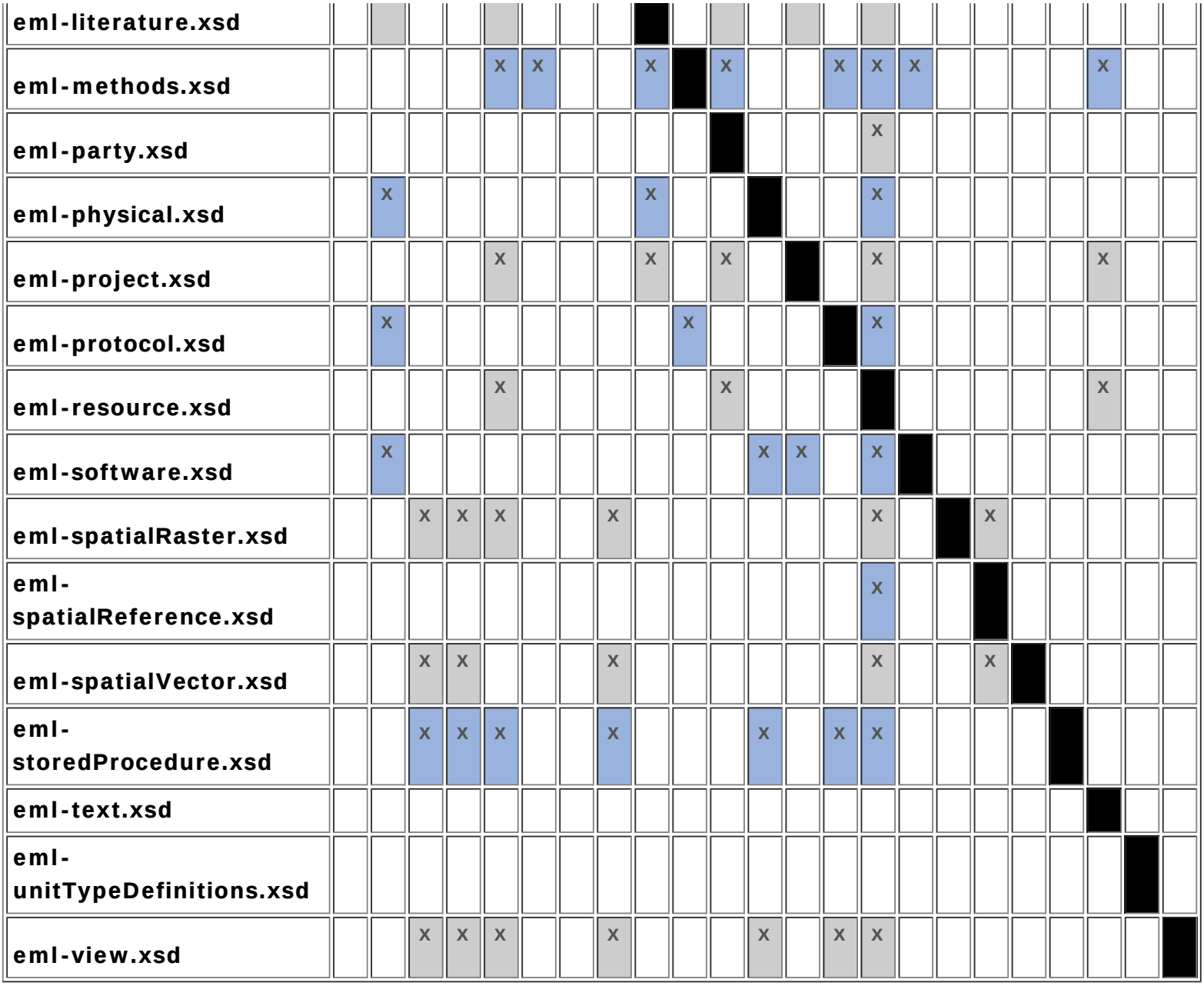
EML Dependency Chart

To read this chart, look at the row header and move across to the X. The relationship can be read *<row header> depends on <column header>* or *<row header> imports <column header>*

If you read the chart in the reverse fashion the relationship becomes *<column header> is depended on by <row header>* or *<column header> is imported by <row header>*

If one module depends on another, it is a good chance that you are going to be using at least a small subset of the other modules nodes within the module that you want to use. This can be seen clearly by looking at the diagrams associated with each **module**.

	eml.xsd	eml-access.xsd	eml-attribute.xsd	eml-constraint.xsd	eml-coverage.xsd	eml-dataset.xsd	eml-dataTable.xsd	eml-entity.xsd	eml-literature.xsd	eml-methods.xsd	eml-party.xsd	eml-physical.xsd	eml-project.xsd	eml-protocol.xsd	eml-resource.xsd	eml-software.xsd	eml-spatialRaster.xsd	eml-spatialReference.xsd	eml-spatialVector.xsd	eml-storedProcedure.xsd	eml-text.xsd	eml-unitTypeDefinitions.xsd	eml-view.xsd
eml.xsd		x				x			x					x	x	x							
eml-access.xsd															x								
eml-attribute.xsd					x				x	x					x							x	
eml-constraint.xsd															x								
eml-coverage.xsd									x		x				x							x	
eml-dataset.xsd		x					x	x		x	x		x		x		x		x	x	x		x
eml-dataTable.xsd			x	x				x							x								
eml-entity.xsd			x	x	x					x		x			x						x		
		x			x						x		x		x								


eml-literature.xsd
eml-methods.xsd
eml-party.xsd
eml-physical.xsd
eml-project.xsd
eml-protocol.xsd
eml-resource.xsd
eml-software.xsd
eml-spatialRaster.xsd
eml-spatialReference.xsd
eml-spatialVector.xsd
eml-storedProcedure.xsd
eml-text.xsd
eml-unitTypeDefinitions.xsd
eml-view.xsd

Module Documentation: eml

Back to EML Contents

The eml module - A metadata container

The eml module is a wrapper container that allows the inclusion of any metadata content in a single EML document. The eml module is used as a container to hold structured descriptions of ecological resources. In EML, the definition of a resource comes from the **The Dublin Core Metadata Initiative** , which describes a general element set used to describe "networked digital resources". The top-level structure of EML has been designed to be compatible with the Dublin Core syntax. In general, dataset resources, literature resources, software resources, and protocol resources comprise the list of information that may be described in EML. EML is largely designed to describe digital resources, however, it may also be used to describe non-digital resources such as paper maps and other non-digital media. In EML, the definition of a "Data Package" is the combination of both the data and metadata for a resource. So, data packages are built by using the <eml> wrapper, which will include all of the metadata, and optionally the data (or references to them). All EML packages must begin with the <eml> tag and end with the </eml> tag.

The eml module may be extended to describe other resources by means of its optional sub-field, <additionalMetadata>. This field is largely reserved for the inclusion of metadata that may be highly discipline specific and not covered in this version of EML, or it may be used to internally extend fields within the EML standard.

Module details

Recommended Usage:	all datasets
Stand-alone:	yes
Imports:	eml-documentation, eml-dataset, eml-literature, eml-software, eml-protocol, eml-resource, eml-access
Imported By:	
View an image of the schema:	**eml image**

Element Definitions:

eml	**This element has no default value.**

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
access	optional	
A choice of (		
dataset	required	
OR		
citation	required	
OR		
software	required	
OR		
protocol	required	
)		
additionalMetadata	optional	unbounded
)		

Attributes:	**Use:**	**Default Value:**
packageId	required	
system	required	
scope		

Description of this field:

The "eml" element allows for the inclusion of any metadata content in a single EML document. In general, dataset resources, literature resources, and software resources, or another type that extends eml-resource are described using an eml document. The eml document represents a "package" that can contain both metadata and data. It can optionally include non-EML metadata through the flexibility of the "additionalMetadata" element. Any additional metadata that is provided can provide a pointer into the EML metadata indicating what the context of the additional metadata is (i.e., what it describes). For example, a spatial raster image might be described in EML, and an FGDC CSDGM metadata document could be included in the additionalMetadata element with a pointer to the EML spatialRaster element to indicate that the FGDC metadata is providing supplemental documentation about that particular image entity. There is no validity constraint that restricts what metadata may be present in additionalMetadata.

access	This element has no default value.
Content of this field: **Type:** acc:AccessType	Description of this field: An optional access tree at this location controls access to the entire metadata document. If this access element is omitted from the document, then the package submitter should be given full access to the package but all other users should be denied all access.

dataset	This element has no default value.
Content of this field: **Type:** ds:DatasetType	Description of this field: A resource that describes a data set, which can include one or more data entities such as data tables and spatial images (raster and vector). If included, this represents the primary resource that is described in this eml document.

citation	This element has no default value.
Content of this field: **Type:** cit:CitationType	Description of this field: A resource that describes a literature citation that one might find in a bibliography. If included, this represents the primary resource that is described in this eml document.

software	This element has no default value.
Content of this field: **Type:** sw:SoftwareType	Description of this field: A resource that describes a software package, which can include commercial and non-commercial software as well as data processing programs. If included, this represents the primary resource that is described in this eml document.

protocol	This element has no default value.
Content of this field: **Type:** prot:ProtocolType	Description of this field: A resource that describes a scientific protocol, which can include one or more descriptions of methods and procedures. If included, this represents the primary resource that is described in this eml document.

additionalMetadata	This element has no default value.
Content of this field: **Elements:** **Use:** **How many:** A sequence of (**describes** optional unbounded **metadata** required) **Attributes:** **Use:** **Default Value:** **id** optional	Description of this field: A flexible field for including any other relevant metadata that pertains to the resource being described. This field allows EML to be extensible in that any XML-based metadata can be included in this element, including metadata from other standards such as the FGDC CSDGM. The "describes" element of this field allows the specific part of the resource which is described by the additional metadata to be indicated formally.

describes	This element has no default value.
Content of this field:	Description of this field:

Type: res:NonEmptyStringType

A pointer to the id attribute for the sub-portion of the resource that is described by this additional metadata. This is a formal field in that it is an error to provide a value in the "describes" element that does not correspond to the value of one of the "id" attributes in another eml module. This is designed to allow automated processors to discover the contextual relationship between the additional metadata and the resource it describes.

Example(s):

knb.343.22

metadata	This element has no default value.

Content of this field:

Elements: **Use:** **How many:**

A sequence of (

)

Description of this field:

This element contains the additional metadata to be included in the document. This element should be used for extending EML to include metadata that is not already available in another part of the EML specification, or to include site- or system-specific extensions that are needed beyond the core metadata. The additional metadata contained in this field describes the element referenced in the 'describes' element preceding it. The content of this field is any well-formed XML fragment. If that content contains namespace declarations, and if the namespace declaration can be resolved to a schema definition, then the content will be validated against that schema definition. If no namespace is present, or if no schema can be resolved, validation for this fragment will be skipped (validation is "lax").

Example(s):

<embargoDate>2006-10-10</embargoDate>

Attribute Definitions:

id

Type: res:IDType

Use: optional

packageId

Type: xs:string

Use: required

A unique identifier for this entire EML metadata document that can be used to reference it elsewhere. This identifier can be interpreted as the formal accession number for this EML package, and is therefore required. It must be unique within a particular data management system (see the "system" attribute).

Example(s):

knb.343.22

system

Type: res:SystemType

Use: required

scope

Type: res:ScopeType	The scope of the identifier. Scope is generally set to either "system", meaning that it is scoped according to the "system" attribute, or "document" if it is only to be in scope within this single document instance. In this particular use of scope, it is FIXED to be "system" because the packageId is required and always has the scope of the required "system". **Example(s):** system

Complex Type Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-access

Back to EML Contents

The eml-access module - Access control rules for resources

The eml-access module describes the level of access that is to be allowed or denied to a resource for a particular user or group of users, and can be described independently for metadata and data. The eml-access module uses a reference to a particular authentication system to determine the set of principals (users or groups) that can be specified in the access rules. The special principal 'public' can be used to indicate that any user or group has access permission, thereby making it easier to specify that anonymous access is allowed.

There are two mechanisms for including access control via the eml-access module:

1) The top-level "eml" element may have an optional <access> element that is used to establish the default access control for the entire EML package. If this access element is omitted from the document, then the package submitter should be given full access to the package but all other users should be denied all access. To allow the package to be publicly viewable, the EML author must explicitly include a rule stating so.

2) Exceptions for particular entity-level components of the package can be controlled at a finer grain by using an access description in that entity's physical/distribution tree. When access control rules are specified at this level, they apply only to the data in the parent distribution element, and not to the metadata. Thus, it will control access to the content of the <inline> element, as well as resources that are referenced by the <online/url> and <online/connection> paths. These exceptions to access for particular data resources are applied after the default access rules at the package-level have been applied, so they effectively override the default rules when they overlap.

In previous versions of EML access rules for entity-level distribution were contained in <additionalMetadata> sections and referenced via the <describes> tag. Although in theory these could have referenced any node, in application such node-level access control is problematic. Since the most common uses of access control rules were to limit access to specific data entities, the access tree has been placed there explicitly in EML 2.1.0.

Access is specified with a choice of child elements, either <allow> or <deny>. Within these rules, values can be assigned for each <principal> using the <permission> element. Users given "read" permission can view the resource; "write" allows changes to the resource excluding changes to the access rules; "changePermission" includes "write" plus the changing of access rules. Users allowed "all" permissions; may do all of the above.

An example is given below, with non-critical sections deleted:

```
<eml>
    <access
        authSystem="ldap://ldap.ecoinformatics.org:389/dc=ecoinformatics,dc=org"
        order="allowFirst">
      <allow>
        <principal>uid=alice,o=NASA,dc=ecoinformatics,dc=org</principal>
        <permission>read</permission>
        <permission>write</permission>
      <allow>
    </access>
    <dataset>
    ...
    ...
    <dataTable id="entity123">
    ...
      <physical>
      ...
        <distribution>
        ...
          <access id="access123"
          authSystem="ldap://ldap.ecoinformatics.org:389/dc=ecoinformatics,dc=org"
          order="allowFirst">
            <deny>
              <principal>uid=alice,o=NASA,dc=ecoinformatics,dc=org</principal>
              <permission>write</permission>
          </deny>
        </access>
       </distribution>
     </physical>
    </dataTable>
    <dataTable id="entity234">
      ...
      <physical>
      ...
        <distribution>
          ...
          <access>
            <references>access123</references>
          </access>
        </distribution>
      </physical>
    </dataTable>
    ...
  </dataset>
<eml>
```

In this example, the overall default access is to allow the user=alice (but no one else) to read and write all metadata and data. However, under "entity123" and "entity234", there is an additional rule saying that user=alice does not have write

permission. The net effect is that Alice can read and make changes to the metadata, but cannot make changes to the two data entities. In addition, Alice cannot change these access rules; although the submitter can.

This example also shows how the eml-access module, like other modules, may be "referenced" via the <references> tag. This allows an access control document to be described once, and then used as a reference in other locations within the EML document via its ID.

In summary, access rules can be applied in two places in an eml document. Default access rules are established in the top <access> element for the main eml document (e.g., "/eml/access"). These default rules can be overridden for particular data entities by adding additional <access> elements in the physical/distribution trees of those entities.

Module details

Recommended Usage:	all data where controlling user access to the dataset is an issue
Stand-alone:	yes
Imports:	eml-documentation, eml-resource
Imported By:	
View an image of the schema:	**eml-access image**

Element Definitions:

access	**This element has no default value.**
Content of this field: **Type:** **AccessType**	Description of this field: The access element contains a list of rules defining permissions for this resource. For descriptions of the individual elements, see the AccessType.The permission rules defined here can be overridden by rules added to an access tree in the PhysicalDistributionType at the entity level. **Example(s):** See the description of the AccessType.

allow	**This element has no default value.**
Content of this field: **Type:** **AccessRule**	Description of this field: The allow element indicates that a particular user or group is granted the defined permission. **Example(s):** allow

deny	**This element has no default value.**
Content of this field: **Type:** **AccessRule**	Description of this field: The deny element indicates that a particular user or group is not granted the defined permission. **Example(s):** deny

principal	**This element has no default value.**
Content of this field: **Type:** **res:NonEmptyStringType**	Description of this field: The principal element defines the user or group to which the access control rule applies. The users and groups must be defined in the authentication system described in the authSystem element. The special principal 'public' can be used to indicate that any user or group has a particular access permission, thereby making it easier to specify that anonymous access is allowed.

Example(s):
public
uid=alice,o=LTER,dc=ecoinformatics,dc=org

permission	This element has no default value.

Content of this field:

Description of this field:

The permission that is being granted or denied to a particular user or group for a given resource. The list of permissions come from a predetermined list:

'read' - allow or deny viewing of the resource,

'write' - allow or deny modification of the resource (except for access rules),

'changePermission' - modifications including access rules, and

'all' - all of the above.

This element also allows other permission values that may be applicable to some other authentication systems but are not defined in this specification (if these other values are used, access rule enforcement is indeterminate outside of the originating system).

Example(s):
read

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

order

Use: optional

Default value: allowFirst

Derived from: xs:string (by xs:restriction)

Allowed values:

- allowFirst

- denyFirst

  To obtain the desired access control, use the order attribute to define which rules should be applied first. The acceptable values are 'allowFirst' and 'denyFirst'. If 'allowFirst' is specified, then all 'allow' rules are processed, and then overridden by all 'deny' rules. If 'denyFirst' is specified, then all 'deny' rules are processed, and then overridden by all 'allow' rules.
  Example(s):
  allowFirst

authSystem	
Type: xs:string **Use:** required	The authentication system determines the set of principals (users + groups) that can be used in the access control list, and the membership of users in groups. This element is intended to provide a reference to the authentication system that is used to verify the user or group. This reference is typically in the form of a URI, which includes the connection protocol, Internet host, and path to the authentication mechanism. **Example(s):** ldap://ldap.ecoinformatics.org:389/dc=ecoinformatics,dc=org

Complex Type Definitions:

AccessType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A choice of (		
A choice of (		
allow	required	
OR		
deny	required	
)		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document
order	optional	allowFirst
authSystem	required	

The access element contains a list of rules that define the level of access for a resource. There are two uses of access trees: to control access to either metadata or data. To control access to metadata use the eml/access tree. By default, these rules will also apply to the contained data. To override the default controls for specific data entities, use the access element available in the entity's physical/distribution tree. A combination of access trees and their "order rules" (see description of the "order" attribute) allows EML authors to have fine control over permissions for individuals and groups.

AccessRule	

Content of this field:

Elements:	Use:	How many:
A sequence of (		
principal	required	unbounded
permission	required	unbounded
)		

Description of this field:

The AccessRule type defines a list of users that are derived from a particular authentication system (such as an LDAP directory), whether the user or group is allowed or denied access, the extent of their access (read, write , or changePermission access).

Simple Type Definitions:

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-attribute

Back to EML Contents

The eml-attribute module - Attribute level information within dataset entities

The eml-attribute module describes all attributes (variables) in a data entity: dataTable, spatialRaster, spatialVector, storedProcedure, view or otherEntity. The description includes the name and definition of each attribute, its domain, definitions of coded values, and other pertinent information. Two structures exist in this module: 1. attribute is used to define a single attribute; 2. attributeList is used to define a list of attributes that go together in some logical way.

The eml-attribute module, like other modules, may be "referenced" via the <references> tag. This allows an attribute document to be described once, and then used as a reference in other locations within the EML document via its ID.

Philosophy of Attribute Units

The concept of "unit" represents one of the most fundamental categories of metadata. The classic example of data entropy is the case in which a reported numeric value loses meaning due to lack of associated units. Much of Ecology is driven by measurement, and most measurements are inherently comparative. Good data description requires a representation of the basis for comparison, i.e., the unit. In modeling the attribute element, the authors of EML drew inspiration from the **NIST Reference on Constants, Units, and Uncertainty**. This document defines a unit as "a particular physical quantity, defined and adopted by convention, with which other particular quantities of the same kind are compared to express their value." The authors of the EML 2.0 specification (hereafter "the authors") decided to make the unit element required, wherever possible.

Units may also be one of the most problematic categories of metadata. For instance, there are many candidate attributes that clearly have no units, such as named places and letter grades. There are other candidate attributes for which units are difficult to identify, despite some suspicion that they should exist (e.g. pH, dates, times). In still other cases, units may be meaningful, but apparently absent due to dimensional analysis (e.g. grams of carbon per gram of soil). The relationship between units and dimensions likewise is not completely clear.

The authors decided to sharpen the model of attribute by nesting unit under measurementScale. Measurement Scale is a data typology, borrowed from Statistics, that was introduced in the 1940's. Under the adopted model, attributes are classified as nominal, ordinal, interval, and ratio. Though widely criticized, this classification is well-known and provides at least first-order utility in EML. For example, nesting unit under measurementScale allows EML to prevent its meaningless inclusion for categorical data -- an approach judged superior to making unit universally required or universally optional.

The sharpening of the attribute model allowed the elimination of the unit type "undefined" from the standard unit dictionary (see eml-unitDictionary.xml). It seemed self-defeating to require the unit element exactly where appropriate, yet still allow its content to be undefined. An attribute that requires a unit definition is malformed until one is provided. The unit type "dimensionless" is preserved, however. In EML 2.0, it is synonymous with "unitless" and represents the case in which units cannot be associated with an attribute for some reason, despite the proper classification of that attribute as interval or ratio. Dimensionless may itself be an anomaly arising from the limitations of the adopted measurement scale typology.

Closely related to the concept of unit is the concept of attribute domain. The authors decided that a well-formed description of an attribute must include some indication of the set of possible values for that attribute. The set of possible values is useful, perhaps necessary, for interpreting any particular observed value. While universally required, attribute domain has different forms, depending on the associated measurement scale.

The element storageType has an obvious relationship to domain. It gives some indication of the range of possible values of an attribute, and also gives some (potentially critical) operability information about the way the attribute is represented or construed in the local storage system. The storageType element seems to fall in a gray area between the logical and physical aspects of stored data. Neither comfortable with eliminating it nor with making it required, the authors left it available but optional under attribute. In addition, it is repeatable so that different storage types can be provided for various systems (e.g., different databases might use different types for columns, even though the domain of the attribute is the same regardless of which database is used).

Attributes representing dates, times, or combinations thereof (hereafter "dateTime") were the most difficult to model in EML. Is dateTime of type interval or ordinal? Does it have units or not? Strong cases can be made on each side of the issue. The confusion may reflect the limitations of the measurement scale typology. The final resolution of the dateTime model is probably somewhat arbitrary. There was clearly a need, however, to allow for the interoperability of dateTime formats. EML 2.0 tries to provide an unambiguous mechanism for describing the format of dateTime values by providing a separate category for date and time values. This "dateTime" measurement scale allows users to explicitly label attributes that contain Gregorian date and time values, and allows them to provide the information needed to parse these values into their appropriate components (e.g., days, months, years)./

Module details

Recommended Usage:	any dataset that uses dataTable, spatialRaster, spatialVector, storedProcedure, view or otherEntity or in a custom module where one wants to document an attribute (variable)
Stand-alone:	yes
Imports:	eml-documentation, eml-methods, eml-coverage, eml-literature, eml-resource, eml-unitTypeDefinitions

Imported By:

View an image of the schema: **eml-attribute image**

Element Definitions:

attribute	This element has no default value.

Content of this field:

Type: AttributeType

Description of this field:

The content model for attribute is a CHOICE between "references" and all of the elements that let you describe the attribute (e.g., attributeName, attributeDefinition, precision). The attribute element allows a user to document the characteristics that describe a 'field' or 'variable' in a data entity (e.g. dataTable). Complete attribute descriptions are perhaps the most important aspect to making data understandable to others. An attribute element describes a single attribute or an attribute element can contain a reference to an attribute defined elsewhere. Using a reference means that the referenced attribute is (semantically) identical, not just in name but identical in its complete description. For example, if attribute "measurement1" in dataTable "survey1" has a precision of 0.1 and you are documenting dataTable survey2 which has an attribute called "measurement1" but the survey2's measurement1 has a precision of 0.001 then these are different attributes and must be described separately.

attribute	This element has no default value.

Content of this field:

Type: AttributeType

Description of this field:

attributeName	This element has no default value.

Content of this field:

Description of this field:

Attribute name is official name of the attribute. This is usually a short, sometimes cryptic name that is used to refer to the attribute. Many systems have restrictions on the length of attribute names, and on the use of special

Type: res:NonEmptyStringType

characters like spaces in the name, so the attribute name is often not particularly useful for display (use attributeLabel for display). The attributeName is usually the name of the variable that is found in the header of a data file.

Example(s):

spden
spatialden
site
spcode

attributeLabel	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

A descriptive label that can be used to display the name of an attribute. This is often a longer, possibly multiple word name for the attribute than the attributeName. It is not constrained by system limitations on length or special characters. For example, an attribute with a name of 'spcode' might have an attributeLabel of 'Species Code'.

Example(s):

Species Density
Spatial Density
Name of Site
Species Code

attributeDefinition	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

This element gives a precise definition of attribute in the data entity (dataTable, spatialRaster, spatialVector, storedProcedure, view or otherEntity) being documented. It explains the contents of the attribute fully so that a data user could interpret the attribute accurately. Some additional information may also be found in the methods element as well.

Example(s):

"spden" is the number of individuals of all macro invertebrate species found in the plot

storageType	This element has no default value.

Content of this field:

Description of this field:

This element describes the storage type, for data in a RDBMS (or other data management system) field. As many systems do not provide for fine-grained restrictions on types, this type will often be a superset of the allowed domain defined in attributeDomain. Values for this field are by default drawn from the XML Schema Datatypes standard values, such as: integer, double, string, etc. If the XML Schema Datatypes are not used, the type system from which the values are derived should be listed in the 'typeSystem' attribute described below. This field represents a 'hint' to processing systems as to how the attribute might be represented in a system or language, but is distinct from the actual expression of the domain of the attribute. The field is repeatable so that the storageType can be indicated for multiple type systems (e.g., Oracle data types and Java data types).

Example(s):

integer
int

Attributes:	Use:	Default Value:
typeSystem	optional	http://www.w3.org/2001/XMLSchema-datatypes

measurementScale	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
nominal	required	
OR		
ordinal	required	
OR		
interval	required	
OR		
ratio	required	
OR		
dateTime	required	
)		

Description of this field:

The measurementScale element indicates the type of scale from which values are drawn for the attribute. This provides information about the scale in which the data was collected.

Example(s):

Nominal is used when numbers have only been assigned to a variable for the purpose of categorizing the variable. An example of a nominal scale is assigning the number 1 for male and 2 for female. Ordinal is used when the categories have a logical or ordered relationship to each other. These types of scale allow one to distinguish the order of values, but not the magnitude of the difference between values. An example of an ordinal scale is a categorical survey where you rank a variable 1=good, 2=fair, 3=poor. Interval is used for data which consist of equidistant points on a scale. The Celsius scale is an interval scale, since each degree is equal but there is no natural zero point (so, 20 C is not twice as hot as

10 C).
Ratio is used for data which consists not only of equidistant points but also has a meaningful zero point, which allows ratios to have meaning. An example of a ratio scale would be the Kelvin temperature scale (200K is half as hot as 400K), and length in meters (e.g., 10 meters is twice as long as 5 meters).

nominal	This element has no default value.

Content of this field:

Description of this field:

This field is used for defining the characteristics of this variable if it is a nominal scale variable, which are variables that are categorical in nature. Nominal is used when numbers have only been assigned to a variable for the purpose of categorizing the variable. An example of a nominal scale is assigning the number 1 for male and 2 for female.

Elements:	Use:	How many:
A sequence of (**nonNumericDomain**)	required	

nonNumericDomain	This element has no default value.

Content of this field:

Description of this field:

Type: NonNumericDomainType

ordinal	This element has no default value.

Content of this field:

Description of this field:

This field is used for defining the characteristics of this variable if it is an ordinal scale variable, which specify ordered values without specifying the magnitude of the difference between values. Ordinal is used when the categories have a logical or ordered relationship to each other. These types of scale allow one to distinguish the order of values, but not the magnitude of the difference between values. An example of an ordinal scale is a categorical survey where you rank a variable 1=good, 2=fair, 3=poor.

Elements:	Use:	How many:
A sequence of (**nonNumericDomain**)	required	

nonNumericDomain	This element has no default value.

Content of this field:

Description of this field:

Type: NonNumericDomainType

interval	This element has no default value.

Content of this field:

Description of this field:

This field is used for defining the characteristics of this variable if it is an interval scale variable, which specifies both the order and magnitude of values, but has no natural zero point. Interval is used for data which consist of equidistant points on a scale. The Celsius scale is an interval scale, since each degree is equal but there is no natural zero point (so, 20 C is not twice as hot as 10 C). zero point (so, 20 C is not twice as hot as 10 C).

Elements:	Use:	How many:
A sequence of (		
unit	required	
precision	optional	
numericDomain	required	
)		

unit	This element has no default value.

Content of this field:

Description of this field:

Type: UnitType

precision	This element has no default value.

Content of this field:

Description of this field:

Type: PrecisionType

numericDomain	This element has no default value.

Content of this field:

Description of this field:

Type: NumericDomainType

ratio	This element has no default value.

Content of this field:

Description of this field:

This field is used for defining the characteristics of this variable if it is a ratio scale variable, which specifies the order and magnitude of values and has a natural zero point, allowing for ratio comparisons to be valid. Ratio is used for data which consists not only of equidistant points but also has a meaningful zero point, which allows ratios to have meaning. An example of a ratio scale would be the Kelvin temperature scale (200K is half as hot as 400K), and length in meters (e.g., 10 meters is twice as long as 5 meters).

Elements:	Use:	How many:
A sequence of (		
unit	required	
precision	optional	
numericDomain	required	
)		

unit	This element has no default value.

Content of this field:

Description of this field:

Type: UnitType

precision	This element has no default value.

Content of this field:

Description of this field:

Type: PrecisionType

numericDomain	This element has no default value.

Content of this field:

Description of this field:

Type: NumericDomainType

dateTime	This element has no default value.

Content of this field:

Description of this field:

The dateTime field is used for defining the characteristics of the attribute if it contains date and time values. DateTime is used when the values fall on the Gregorian calendar system. DateTime values are special because the have properties of interval values (most of the time it is legitimate to treat them as interval values by converting them to a duration from a fixed point) but they sometimes only behave as ordinals (because the calendar is not predetermined, for some dateTime values one can only find out the order of the points and not the magnitude of the duration between those points). Thus, the dateTime scale provides the information necessary to properly understand and parse date and time values without improperly labeling them under one of the more traditional scales.

Date and time values are unlike any other measured values. Note that the dateTime field would not be used if one is recording time durations. In that case, one should use a standard unit such as seconds, nominalMinute or nominalDay, or a customUnit that defines the unit in terms of its relationship to SI second.

Elements:	Use:	How many:
A sequence of (		
formatString	required	
dateTimePrecision	optional	
dateTimeDomain	optional	
)		

formatString	This element has no default value.

Content of this field:

Description of this field:

A format string that describes the format for a dateTime value from the Gregorian calendar. DateTime values should be expressed in a format that conforms to the ISO 8601 standard. This field allows one to specify the format string that should be used to decode the date or time value. To describe the format of an attribute containing dateTime values, construct a string representation of the format using the following symbols:

```
             Y   year
             M   month
             W   month abbreviation
(e.g., JAN)
             D   day
             h   hour
             m   minute
             s   second
             T   time designator
(demarcates date and time parts of
date-time)
             Z   UTC designator,
indicating value is in UTC time
             .   indicates a
decimal fraction of a unit
             +/- indicates a
positive or negative number,
                 or a positive or
negative time zone adjustment
relative to UTC
             -   indicates a
separator between date components
             A/P am or pm
designator
```

Any other character in the format string is interpreted as a separator character. Here are some examples of the format strings that can be constructed.

```
Format string          Example value
                                  ---
----------------    -------------
-----
                ISO Date
YYYY-MM-DD             2002-10-14
                ISO Datetime
YYYY-MM-DDThh:mm:ss    2002-10-
14T09:13:45
                ISO Time
```

hh:mm:ss	17:13:45
	ISO Time
hh:mm:ss.sss	09:13:45.432
	ISO Time
hh:mm.mm	09:13.42
	Non-standard
DD/MM/YYYY	14/10/2002
	Non-standard
MM/DD/YYYY	10/14/2002
	Non-standard
MM/DD/YY	10/14/02
	Non-standard
YYYY-WWW-DD	2002-OCT-14
	Non-standard
YYYYWWWDD	2002OCT14
	Non-standard
YYYY-MM-DD hh:mm:ss	2002-10-14 09:13:45

Type: **res:NonEmptyStringType**

Some notes about these examples. First, the ISO 8601 standard is strict about the order of date components and the separators that are legal. Best practice is to follow the ISO 8601 format precisely. However, we recognize that existing data contain non-standard dates, and existing equipment (e.g., sensors) may still be producing non-standard dates. Consequently, we have provided the formatting string with additional characters to describe the date formats. In particular note that the use of a slash (/) to separate date components, a space to separate date and time components, using a twelve-hour time with am/pm designator, and placing any of the components out of descending order is non-standard according to ISO. Nevertheless, these formats can be described using the format string to accommodate existing data.

Decimal dateTime values can be extended by indicating in the format that additional decimals can be used. Only the final unit (e.g., seconds in a time value) can use the extended digits according to the ISO 8601 standard. For example, to show indicate that seconds are represented to the nearest 1/1000 of a second, the format string would be "hh:mm:ss.sss". Note that this only indicates the number of decimals used to

record the value, and not the precision of the measurement (see dateTimePrecision for that).

Date and time values are from an interval scale, but it is extremely complex because of the vagaries of the calendar (e.g., leap years, and leap seconds). The duration between date and time values in the future is not even deterministic because leap seconds are based on current measurements of the earth's orbit. Consequently, date and time values are unlike any other measured values. The format string for dateTime values allows one to accurately calculate the duration in SI second units between two measured dateTime values, assuming that the conversion software has a detailed knowledge of the Gregorian calendar.

Example(s):
YYYY-MM-DDThh:mm:ss
YYYY-MM-DD
YYYY
hh:mm:ss
hh:mm:ss.sss

dateTimePrecision	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

A quantitative indication of the precision of a date or time measurement. The precision should be interpreted in the smallest units represented by the dateTime format. For example, if a dateTime value has a format of "hh:mm:ss.sss", then "seconds" are the smallest unit and the precision should be expressed in seconds. Thus, a precision value of "0.01" would mean that measurements were precise to the nearest hundredth of a second, even though the format string might indicate that values were written down with 3 decimal places.

Example(s):
0.1
0.01

dateTimeDomain	This element has no default value.

Content of this field:

Description of this field:

Type: DateTimeDomainType

See the description for the type: DateTimeDomainType

missingValueCode	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
code	required	
codeExplanation	required	
)		

Description of this field:

This element is to specify missing value in the data of the field. It is repeatable to allow for multiple different codes to be present in the attribute. Note that missing value codes should not be considered when determining if the observed values of an attribute all fall within the domain of the attribute (i.e., missing value codes should be parsed out of the data stream before examining the data for domain violations.

code	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The code element is the missing value code itself. Each missing value code should be entered in a separate element instance. The value entered is what is placed into a data grid if the value is missing for some reason.

Example(s):

-9999
-1
N/A
MISSING

codeExplanation	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The codeExplanation element is an explanation of the meaning of the missing value code that was used, that is, the reason that there is a missing value. For example, an attribute might have a missing value code of '-99' to indicate that the data observation was not actually taken, and a code of '-88' to indicate that the data value was removed because of calibration errors.

Example(s):

Sensor down time.
Technician error.

accuracy	This element has no default value.

Content of this field:

Type: Accuracy

Description of this field:

The accuracy element represents the accuracy of the attribute. This information should describe any accuracy information that is known about the collection of this data attribute. The content model of this metadata is taken directly from FGDC FGDC-STD-001-1998 section 2 with the exception of processContact, sourceCitation, and timePeriodInformation which either user XMLSchema types or use predefined EML types for these purposes.

coverage	This element has no default value.

Content of this field:

Type: cov:Coverage

Description of this field:

An explanation of the coverage of the attribute. This specifically indicates the spatial, temporal, and taxonomic coverage of the attribute in question when that coverage deviates from coverages expressed at a higher level (e.g., entity or dataset). Please see the eml-coverage module for complete documentation.

methods	This element has no default value.

Content of this field:

Type: md:MethodsType

Description of this field:

An explanation of the methods involved in the collection of this attribute. These specifically supplement or possibly override methods provided at a higher level such as entity or dataset. Please see the eml-methods module for complete documentation.

attributeAccuracyReport	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeAccuracyReport element is an explanation of the accuracy of the observation recorded in this attribute. It will often include a description of the tests used to determine the accuracy of the observation. These reports are generally prepared for remote sensing or other measurement devices.

quantitativeAttributeAccuracyAssessment	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
attributeAccuracyValue	required	
attributeAccuracyExplanation	required	
)		

Description of this field:

The quantitativeAttributeAccuracyAssessment element is composed of two parts, a value that represents the accuracy of the recorded observation an explanation of the tests used to determine the accuracy.

attributeAccuracyValue	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeAccuracyValue element is an estimate of the accuracy of the identification of the entities and assignments of attribute values in the data set.

attributeAccuracyExplanation	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeAccuracyExplanation element is the identification of the test that yielded the Attribute Accuracy Value.

attributeList	This element has no default value.

Content of this field:

Type: AttributeListType

Description of this field:

This is the root element of the eml-attribute module. It is mainly used for testing, but can also be used for creating stand-alone eml-attribute modules where a list of attributes is needed.

standardUnit	This element has no default value.

Content of this field:

Type: unit:StandardUnitDictionary

Description of this field:

Use the standardUnit element if the unit for this attribute has been defined in the Standard Unit Dictionary. The list of "standard" units includes the SI base units and many compound units based on SI, plus and some commonly used units which are not SI. The list is by no means exhaustive. If the unit you need is not part of this list, then the customUnit field should be used instead. Standard units

have been described using STMML. See the documentation for the Type for more information.

Example(s):

meter
second
joule

customUnit	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The customUnit element is for units that are not part of the standard list provided with EML. The customUnit must correspond to an id in the document where its definition is provided using the STMML syntax. The customUnit definition will most likely be in the additionalMetadata section.

Example(s):

gramsPerOneThirdMeter

enumeratedDomain	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
codeDefinition	required	unbounded
OR		
externalCodeSet	required	
OR		
entityCodeList	required	
)		
Attributes:	**Use:**	**Default Value:**
enforced	optional	yes

Description of this field:

The enumeratedDomain element describes any code that is used as a value of an attribute. These codes can be defined here in the metadata as a list with definitions (preferred), can be referenced by pointing to an external citation or URL where the codes are defined, or can be referenced by pointing at an entity that contains the code value and code definition as two attributes. For example, data might have a variable named 'site' with values 'A', 'B', and 'C', and the enumeratedDomain would explain how to interpret those codes.

codeDefinition	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
code	required	
definition	required	
source	optional	
)		
Attributes:	**Use:**	**Default Value:**

Description of this field:

This element gives the value of a particular code and its definition. It is repeatable to allow for a list of codes to be provided.

order optional

code	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The code element specifies a code value that can be used in the domain

Example(s):

1
HIGH
BEPA
24

definition	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The definition describes the code with which it is associated in enough detail for scientists to interpret the meaning of the coded values.

Example(s):

high density, above 10 per square meter

source	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The source element is the name of the source from which this code and its associated definition are drawn. This is commonly used for identifying standard coding systems, like the FIPS standard for postal abbreviations for states in the US. In other cases, the coding may be the researcher's customized way of recording and classifying their data, and no external "source" would exist.

Example(s):

ISO country codes

externalCodeSet	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
codesetName	required	
A choice of (		
citation	required	

Description of this field:

The externalCodeSet element is a reference to an externally defined set of codes used in this attribute. This can either be a citation (using the eml-citation module) or a URL. Using an externally defined codeset (rather than a codeDefinition) means that interpretation of the data is dependent upon future users being able to obtain the code definitions,

OR		
codesetURL	required	
)		
)		

so care should be taken to only use highly standardized external code sets that will be available for many years. If at all possible, it is preferable to define the codes inline using the codeDefinition element.

codesetName	**This element has no default value.**

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

The codesetName element is the name of an externally defined code set.

Example(s):

FIPS State Abbreviation Codes

citation	**This element has no default value.**

Content of this field:

Type: **cit:CitationType**

Description of this field:

The citation element is a citation for the code set reference

codesetURL	**This element has no default value.**

Content of this field:

Type: xs:anyURI

Description of this field:

The codesetURL element is a URL for the code set reference.

entityCodeList	**This element has no default value.**

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
entityReference	required	
valueAttributeReference	required	
definitionAttributeReference	required	
orderAttributeReference	optional	
)		

Description of this field:

The entityCodeList is a list of codes and their definitions in a data entity that is present in this dataset. The fields specify exactly which entity it is, and which attributes of that entity contain the codes, their definitions, and the order of the values.

entityReference	**This element has no default value.**

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

The entityReference element is a reference to the id of the entity in which the code list has been defined. This entity must have been defined elsewhere in the metadata and have an id that matches the value of this element.

valueAttributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The valueAttributeReference element is a reference to the id of the attribute that contains the list of codes. This attribute must have been defined elsewhere in the metadata and have an id that matches the value of this element.

definitionAttributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The definitionAttributeReference element is a reference to the id of the attribute that contains the definition of codes. This attribute must have been defined elsewhere in the metadata and have an id that matches the value of this element.

orderAttributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The orderAttributeReference element is a reference to the id of the attribute that contains the order of codes. The values in this attribute are integers indicating increasing values of the categories. This attribute must have been defined elsewhere in the metadata and have an id that matches the value of this element.

textDomain	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
definition	required	
pattern	optional	unbounded
source	optional	
)		

Description of this field:

The textDomain element describes a free text domain for the attribute. By default, if a pattern is missing or empty, then any text is allowed. If a pattern is present, then it is interpreted as a regular expression constraining the allowable character sequences for the attribute. This domain type is most useful for describing extensive text domains that match a pattern but do not have a finite set of values. Another use is for describing the domain of textual fields like comments that allow any legal string value.

Example(s):

Typically, a text domain will have an

empty pattern or one that constrains allowable values. For example, '[0-9]{3}-[0-9]{3}-[0-9]{4}' allows for only numeric digits in the pattern of a US phone number.

definition	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The element definition provides the text domain definition, that is, what kinds of text expressions are allowed for this attribute. If there is a pattern supplied, the definition element expresses the meaning of the pattern, For example, a particular pattern may be meant to represent phone numbers in the US phone system format. A definition element may also be used to extend an enumerated domain.

Example(s):

US telephone numbers in the format "(999) 888-7777"

pattern	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The pattern element specifies a regular expression pattern that constrains the set of allowable values for the attribute. This is commonly used to define template patterns for data such as phone numbers where the attribute is text but the values are not drawn from an enumeration. If the pattern field is empty or missing, it defaults to '.*', which matches any string, including the empty string. Repeated pattern elements are combined using logical OR. The regular expression syntax is the same as that used in the XML Schema Datatypes Recommendation from the W3C.

Example(s):

'[0-9a-zA-Z]' matches simple alphanumeric strings and '(\d\d\d) \d\d\d-\d\d\d\d' represents telephone strings in the US of the form '(704) 876-1734'

source	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

The source element is the name of the source from which this text domain and its associated definition are drawn. This is commonly used for identifying standard coding systems, like the FIPS standard for postal abbreviations for states in the US. In other cases, the coding may be a researcher's custom way of recording and classifying their data, and no external "source" would exist.
Example(s):
ISO country codes

numberType	This element has no default value.

Content of this field:

Description of this field:

Type: NumberType

bounds	This element has no default value.

Content of this field:

Description of this field:

The bounds element in the BoundsGroup contains the minimum and maximum values of a numeric attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive). Either or both a minimum and maximum may be set, and each has an attribute "exclusive" to define how the value should be interpreted.

Elements:	Use:	How many:
A sequence of (		
minimum	optional	
maximum	optional	
)		

minimum	This element has no default value.

Content of this field:

Description of this field:

The minimum element specifies the minimum permitted value of a numeric attribute.

Attributes:	Use:	Default Value:
exclusive	required	

maximum	This element has no default value.

Content of this field:

Description of this field:

The maximum element specifies the maximum permitted value of a numeric attribute.

Attributes:	Use:	Default Value:
exclusive	required	

bounds	This element has no default value.

Content of this field:

Description of this field:

Elements:	Use:	How many:
A sequence of (		
minimum	optional	
maximum	optional	
)		

The bounds element in the BoundsDateGroup contains the minimum and maximum dates of a dateTime attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive). Either or both a minimum and maximum may be set, and each has an attribute "exclusive" to define how the value should be interpreted.

minimum

This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
exclusive	required	

Description of this field:

The minimum element specifies the minimum permitted value of a date attribute.

maximum

This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
exclusive	required	

Description of this field:

The maximum element specifies the maximum permitted value of a date attribute.

Attribute Definitions:

id

Type: res:IDType

Use: optional

typeSystem

Type: xs:string

Use: optional

Default value: http://www.w3.org/2001/XMLSchema-datatypes

The typeSystem attribute is the system used to define the storage types. This should be an identifier of a well known and published typing system. The default and recommended system is the XML Schema data type system. For details go to http://www.w3.org. If another system is used (such as Java or C++ types), typeSystem should be changed to match the system.

Example(s):

http://www.w3.org/2001/XMLSchema-datatypes
java
C
Oracle 8i

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

order

Type: xs:long

Use: optional

Ordinal scale measurements have a discrete list of values with a specific ordering of those values. This attributes specifies that order from low to high. For example, for LOW, MEDIUM, HIGH, the order attribute might be "LOW=1, MEDIUM=2 and HIGH=3".

enforced

Use: optional

Default value: yes

Derived from: xs:string (by xs:restriction)

Allowed values:

- yes
- no

Indicates whether the enumerated domain values are the only allowable values for the domain. In some exceedingly rare cases, users may wish to present a list of value codes in enumeratedDomain but not formally restrict the value space for the attribute to those values. If so, they can indicate this by setting the enforced attribute to the value no. Acceptable values are yes and no, and the default value is yes.

id

Type: res:IDType

Use: optional

id

Type: res:IDType

Use: optional

id

Type: res:IDType

Use: optional

exclusive

Type: xs:boolean

Use: required

If exclusive is set to true, then the value specifies a lower bound not including the value itself. Setting exclusive to true is the equivalent of using a less-than or greater-than operator, while setting it to false is the same as using a less-than-or-equals or greater-than-or-equals operator. For example, if the minimum is "5" and exclusive is false, then all values must be greater than or equal to 5, but if exclusive is true than all values must be greater than 5 (not including 5.0 itself).

exclusive

Type: xs:boolean

Use: required

If exclusive is set to true, then the value specifies a lower bound not including the value itself. Setting exclusive to true is the equivalent of using a less-than or greater-than operator, while setting it to false is the same as using a less-than-or-equals or greater-than-or-equals operator. For example, if the minimum is "5" and exclusive is false, then all values must be greater than or equal to 5, but if exclusive is true than all values must be greater than 5 (not including 5.0 itself).

exclusive

Type: xs:boolean

Use: required

If exclusive is set to true, then the value specifies a lower bound not including the value itself. Setting exclusive to true is the equivalent of using a less-than or greater-than operator, while setting it to false is the same as using a less-than-or-equals or greater-than-or-equals operator. For example, if the minimum is "5" and exclusive is false, then all values must be greater than or equal to 5, but if exclusive is true than all values must be greater than 5 (not including 5.0 itself).

exclusive

Type: xs:boolean

If exclusive is set to true, then the value specifies a lower bound not including the value itself. Setting exclusive to true is the equivalent of using a less-than or greater-than operator, while setting it to false is the same as using a less-than-or-

Use: required

equals or greater-than-or-equals operator. For example, if the minimum is "5" and exclusive is false, then all values must be greater than or equal to 5, but if exclusive is true than all values must be greater than 5 (not including 5.0 itself).

Complex Type Definitions:

AttributeListType

Content of this field:

Elements:	Use:	How many:
A choice of (		
attribute	required	unbounded
OR		
res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	

Description of this field:

This complexType defines the structure of the attributeList element. The content model is a choice between one or more attribute elements, and references. References links to an attribute list defined elsewhere.

AttributeType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
attributeName	required	
attributeLabel	optional	unbounded
attributeDefinition	required	
storageType	optional	unbounded
measurementScale	required	
missingValueCode	optional	unbounded
accuracy	optional	
coverage	optional	
methods	optional	
)		
OR		
res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

Description of this field:

Type definition for the content of an attribute (variable) that can be part of an entity.

Accuracy

Content of this field:

Elements:	Use:	How

Description of this field:

		many:
A sequence of (		
attributeAccuracyReport	required	
quantitativeAttributeAccuracyAssessment	optional	unbounded
)		

UnitType

Content of this field:

Elements:	Use:	How many:
A choice of (		
standardUnit	required	
OR		
customUnit	required	
)		

Description of this field:

This field identifies the unit of measurement for this attribute. It is a choice of either a standard unit, or a custom unit. If it is a custom unit, the definition of the unit must be provided in the document using the STMML syntax, and the name provided in the customUnit element must reference the id of its associated STMML definition precisely. For further information on STMML (http://www.xml-cml.org/stmml/) or see stmml.xsd which is included with the EML 2.0 distribution for details.

PrecisionType

Content of this field:

Derived from: xs:float (by xs:extension)

Elements:	Use:	How many:

Description of this field:

Precision indicates how close together or how repeatable measurements are. A precise measuring instrument will give very nearly the same result each time it is used. This means that someone interpreting the data should expect that if a measurement were repeated, most measured values would fall within the interval specified by the precision. The value of precision should be expressed in the same unit as the measurement. For example, for an attribute with unit "meter", a precision of "0.1" would be interpreted to mean that most repeat measurements would fall within an interval of 1/10th of a meter.

Example(s):

0.1
0.5
1

NonNumericDomainType

Content of this field:

Description of this field:

The non-numeric domain field describes the domain of the attribute being documented. It can describe two different types of domains: enumerated and text. Enumerated domains are lists of values

Elements:	Use:	How many:
A choice of (		
A choice of (		
enumeratedDomain	required	
OR		
textDomain	required	
)		
res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	

that are explicitly provided as legitimate values. Only values from that list should occur in the attribute. They are often used for response codes such as "HIGH" and "LOW". Text domains are used for attributes that allow more free-form text fields, but still permit some specification of the value-space through pattern matching. A text domain is usually used for comment and notes attributes, and other character attributes that don't have a precise set of constrained values. This is an important field for post processing and error checking of the dataset. It represents a formal specification of the value space for the attribute, and so there should never be a value for the attribute that falls outside of the set of values prescribed by the domain.

NumericDomainType

Content of this field:

Description of this field:

The numericDomain element specifies the minimum and maximum values of a numeric attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive). The information in numericDomain and in precision together constitute sufficient information to decide upon an appropriate system specific data type for representing a particular attribute. For example, an attribute with a numeric domain from 0-50,000 and a precision of 1 could be represented in the C language using a 'long' value, but if the precision is changed to '0.5' then a 'float' type would be needed.

Elements:	Use:	How many:
A choice of (		
A sequence of (		
numberType	required	
BoundsGroup		
)		
OR		
res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	

DateTimeDomainType

Content of this field:

Description of this field:

The DateTimeDomain specifies the minimum and maximum values of a dateTime attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive). The domain expressions should be in the same dateTime format as is used in the "formatString" description for the attribute. For example, if the

Elements:	Use:	How many:
A choice of (		
A sequence of (		
BoundsDateGroup		
)		
OR		

res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	

format string is "YYYY-MM-DD", then a valid minimum in the domain would be "2001-05-29". The "bounds" element is optional, and if it is missing then any legitimate value from the Gregorian calendar system is allowed in the attribute as long as its representation matches its corresponding formatString.

Simple Type Definitions:

NumberType

Derived from: xs:string (by xs:restriction)

Allowed values:

- natural
- whole
- integer
- real

This is the enumeration for the allowed values of the element numberType.

Group Definitions:

BoundsGroup

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
bounds	optional	unbounded
)		

Description of this field:

The bounds element contains the minimum and maximum values of a numeric attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive).

BoundsDateGroup

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
bounds	optional	unbounded
)		

Description of this field:

The BoundsDateGroup specifies the minimum and maximum dates allowed for a dateTime attribute. These are theoretical or permitted values (ie. prescriptive), and not necessarily the actual minimum and maximum observed in a given data set (descriptive). The domain expressions should be in the same dateTime format as is used in the attribute's "formatString". For example, if the format string is "YYYY-MM-DD", then a valid minimum in the domain would be "2001-05-29". The "bounds" element is optional, and if it is missing then any legitimate value from the Gregorian calendar system is allowed in the attribute as long as its representation

matches its corresponding formatString.

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-constraint

Back to EML Contents

The eml-constraint module - Relationships among and within dataset entities

The eml-constraint schema defines the integrity constraints between entities (e.g., data tables) as they would be maintained in a relational management system. These constraints include primary key constraints, foreign key constraints, unique key constraints, check constraints, and not null constraints, among potential others.

Module details

Recommended Usage:	All datasets where there are logical constraints between entities
Stand-alone:	no
Imports:	eml-documentation, eml-resource
Imported By:	
View an image of the schema:	**eml-constraint image**

Element Definitions:

primaryKey	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
ConstraintBaseGroup		
key	required	
)		

Description of this field:

The primaryKey element declares the primary key in the entity to which the defined constraint pertains.

Example(s):

date
site

key	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
attributeReference	required	unbounded
)		

Description of this field:

The key element defines the set of attributes to which this constraint applies. For a primary key or a unique key, the set of attributes must be identifying. For a foreign key, the set of attributes must match an identifying key in the referenced entity. For a 'not null' constraint, the key indicates the attribute which should not be null.

Example(s):

site
plot

attributeReference	This element has no default value.

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

The attributeReference element is the identifier of an attribute that can be found in the identified entity. This id will be unique within an entity and specifies that the attribute participates in the key that is being defined.

Example(s):

site

uniqueKey	This element has no default value.

Content of this field:

Description of this field:

The uniqueKey element represents a unique key within the

Elements:	Use:	How many:
A sequence of (		
ConstraintBaseGroup		
key	required	
)		

referenced entity. This is different from a primary key in that it does not form any implicit foreign key relationships to other entities, however it is required to be unique within the entity.

Example(s):

date

key

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
attributeReference	required	unbounded
)		

Description of this field:

The key element defines the set of attributes to which this constraint applies. For a primary key or a unique key, the set of attributes must be identifying. For a foreign key, the set of attributes must match an identifying key in the referenced entity. For a 'not null' constraint, the key indicates the attribute which should not be null.

Example(s):

date, site, plot

attributeReference

This element has no default value.

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

The attributeReference element is the identifier of an attribute that can be found in the identified entity. This id will be unique within an entity and specifies that the attribute participates in the key that is being defined.

Example(s):

site

checkConstraint

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
ConstraintBaseGroup		
checkCondition	required	
)		

Attributes:	Use:	Default Value:
language	optional	

Description of this field:

The checkConstraint element defines a constraint which checks a conditional clause within an entity.

Example(s):

if site>1 then plot>10

checkCondition

This element has no default value.

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

The checkCondition element defines an SQL statement or other language implementation of the condition for a check constraint. Generally this provides a means for constraining the values within and among entities.

Example(s):

(year > 1900 and year < 1990)

foreignKey

This element has no default value.

Content of this field:

Description of this field:

Elements:	Use:	How many:	The foreignKey element defines a foreign key relationship among entities which relates this entity to another's primary key.

joinCondition	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
ForeignKeyGroup		
referencedKey	required	
)		

Description of this field:

The joinCondition element describes any join of two tables that is not done with a primary/foreign key relationship.

Example(s):

JOIN code

referencedKey	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
attributeReference	required	unbounded
)		

Description of this field:

The referencedKey element defines set of attributes to which a foreign key constraint refers. If the key refers to the primary key in the referenced entity, then the "referencedKey" is optional. For a foreign key, the set of attributes must match an identifying key in the referenced entity.

Example(s):

site, plot

attributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeReference element is the identifier of an attribute that can be found in the identified entity. This id will be unique within an entity and specifies that the attribute participates in the key that is being defined.

Example(s):

site

notNullConstraint	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
ConstraintBaseGroup		
key	required	
)		

Description of this field:

The notNullConstraint element defines a constraint that indicates that no null values should be present for an attribute in this entity.

key	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
attributeReference	required	unbounded
)		

Description of this field:

The key element defines the set of attributes to which this constraint applies. For a primary key or a unique key, the set of attributes must be identifying. For a foreign key, the set of attributes must match an identifying key in the referenced entity. For a 'not null' constraint, the key indicates the attribute which should not be null.

attributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeReference element is the identifier of an attribute that can be found in the identified entity. This id will be unique within an entity and specifies that the attribute participates in the key that is being defined.

Example(s):

site

constraintName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The constraintName element is a name which represents a human readable and meaningful name for the constraint.

Example(s):

PrimaryKey_birdSurvey

constraintDescription	This element has no default value.

Content of this field:

Description of this field:

The constraintDescription element describes the nature of the constraint. It might be a description of a check condition, or a statement about the composition of a primary key or the nature of the relationship between two database tables or two ascii files.

Example(s):

1.Must be greater than 0 but less than 100 2. "The primary key of the table BIRD_SURVEY is composed of two attributes:speciesCode and observationDate 3. The species name associated with the species code in survey.txt can be found in the file speciesList.txt

key	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (attributeReference)	required	unbounded

Description of this field:

The key element defines the set of attributes to which this constraint applies. For a primary key or a unique key, the set of attributes must be identifying. For a foreign key, the set of attributes must match an identifying key in the referenced entity. For a 'not null' constraint, the key indicates the attribute which should not be null.

attributeReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The attributeReference element is the identifier of an attribute that can be found in the identified entity. This id will be unique within an entity and specifies that the attribute participates in the key that is being defined.

Example(s):

site

entityReference	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The entityReference element contains the id of the entity to which a foreign key refers, otherwise known as the parent-entity or parent-table. This should be an identifier that matches one of the "identifier" elements for an entity.

Example(s):

knb.79.4

relationshipType	This element has no default value.

Content of this field:

Description of this field:

The relationshipType element defines identifying relationships that propagate from the parent entity's primary key to the child's primary key. Non-identifying relationships propagate the parent's primary key as a non-key attribute of the child entity.

Example(s):

relationshipType code

Derived from: xs:string (by xs:restriction)

Allowed values:

- identifying
- non-identifying

cardinality	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
parentOccurences	required	
childOccurences	required	
)		

Description of this field:

The cardinality element represents a statement of the relationship between parent and child entities. Cardinality is expressed as the ratio of related parent and child entities. Cardinality 1 to N is a specific form of cardinality in which zero or one parent records are related to a specified number of child records. The cardinality ratio for the parent entity depends on whether the "existence" is mandatory (one or more) or optional (zero to ...).

Example(s):

One to many
One to 10
Zero or One to Many

parentOccurences	This element has no default value.

Content of this field:

Description of this field:

The parentOccurences element describes the Parent portion of a 1 to exactly N cardinality. May have a value of either 0 or 1. Value of 0 implies that the "existence" of a child record is optional. Value of 1 implies that the "existence" of a child record is mandatory.

Example(s):

One to 10, Zero or One to Many

Derived from: xs:integer (by xs:restriction)

Allowed values:

- 0
- 1

childOccurences	This element has no default value.
Content of this field: **Type:** CardinalityChildOccurancesType	Description of this field: The childOccurences element describes the child portion of a cardinality expression Allowed values are positive integers including zero or the string value "many" **Example(s):** 2, 15, many

Attribute Definitions:

language	
Type: xs:string **Use:** optional	The language element declares the language that is used to implement the check constraint. This is typically the name and version of a programming language such as Java, C, Perl, Basic, or other. Sometime it is the name and version of a scriptable analysis system such as SAS, Matlab, R, or SPlus. **Example(s):** Perl 5.6.1

id
Type: res:IDType **Use:** optional

system
Type: res:SystemType **Use:** optional

scope
Type: res:ScopeType **Use:** optional **Default value:** document

Complex Type Definitions:

ConstraintType	
Content of this field: **Elements:** **Use:** **How many:** A choice of (**primaryKey** required OR **uniqueKey** required OR	Description of this field: The ConstraintType type describes the relational integrity constraints of a relational database. This includes primary keys, foreign keys, unique keys, etc. When an eml-constraint module is created, it should be linked into a dataset using the "triple" element, and all of the entities that are referenced in the constraints should be accessible within that same package.

checkConstraint	required
OR	
foreignKey	required
OR	
joinCondition	required
OR	
notNullConstraint	required
)	

Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

Simple Type Definitions:

CardinalityChildOccurancesType
The CardinalityChildOccurancesType element defines the child portion of a cardinality expression. Allowed values are positive integers including zero or the string value "many". **Example(s):** 0,1, 2, 15,many

Group Definitions:

ConstraintBaseGroup	
Content of this field:	Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
constraintName	required	
constraintDescription	optional	
)		

ForeignKeyGroup	
Content of this field:	Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
ConstraintBaseGroup		
key	required	
entityReference	required	
relationshipType	optional	
cardinality	optional	
)		

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-coverage

Back to EML Contents

The eml-coverage module - Geographic, temporal, and taxonomic extents of resources

The eml-coverage module contains fields for describing the coverage of a resource in terms of time, space, and taxonomy. These coverages (temporal, spatial, and taxonomic) represent the extent of applicability of the resource in those domains. The Geographic coverage section allows for 2 means of expressing coverage on the surface of the earth: 1) via a set of bounding coordinates that define the North, South, East and West points in a rectangular area, optionally including a bounding altitude, and 2) using a G-Ring polygon definition, where an irregularly shaped area may be defined using a ordered list of latitude/longitude coordinates. A G-Ring may also include an "inner G-Ring" that defines one or more "cut-outs" in the area, i.e. the donut hole concept.

The temporal coverage section allows for the definition of either a single date or time, or a range of dates or times. These may be expressed as a calendar date according to the ISO 8601 Date and Time Specification, or by using an alternate time scale, such as the geologic time scale. Currently, EML does not have specific fields to indicate that a data resource may be "ongoing." Two examples are data tables that are planned to be appended in the future, or resources with complex connection definitions (such as to a database) which may return data in real time. It is important that EML be able to handle data from both the "producer" and "consumer" points of view, although currently the temporal coverage modules are designed for the latter. There is no universally acceptable recommendation for describing "ongoing" data within EML. Some groups have chosen to use the <alternateTimeScale> node for the end date, with a value of "ongoing," although this practice is not endorsed by the EML authors. A better solution could be to use very general content for the endDate (such as only the current year) so that the data are accurately described, and searches return datasets as expected. A future version of EML will accommodate such data types with coverage elements specific to their needs.

The taxonomic coverage section allows for detailed description of the taxonomic extent of the dataset or resource. The taxonomic classification consists of a recursive set of taxon rank names, their values, and their common names. This construct allows for a taxonomic hierarchy to be built to show the level of identification (e.g. Rank Name = Kingdom, Rank Value = Animalia, Common Name = Animals, and so on down the hierarchy.) The taxonomic coverage module also allows for the definition of the classification system in cases where alternative systems are used.

The eml-coverage module, like other modules, may be "referenced" via the <references> tag. This allows the coverage extent to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	all datasets where spatial, temporal or taxonomic coverage is important
Stand-alone:	no
Imports:	eml-literature, eml-documentation, eml-party, eml-resource, eml-unitTypeDefinitions
Imported By:	
View an image of the schema:	**eml-coverage image**

Element Definitions:

geographicCoverage	**This element has no default value.**
Content of this field: **Type:** GeographicCoverage	Description of this field: Geographic Coverage is a container for spatial information about a project, a resource, or an entity within a resource. It allows a bounding box for the overall coverage (in lat long), and also allows description of arbitrary polygons with exclusions. **Example(s):** Please see the individual sub fields for specific examples.

temporalCoverage	**This element has no default value.**
Content of this field: **Derived from:** TemporalCoverage (by xs:extension)	Description of this field: This field specifies temporal coverage, and allows coverages to be a single point in time, multiple points in time, or a range of dates. Dates can be

Elements:	Use:	How many:
system	optional	
scope	optional	document

expressed in terms of both calendar dates and geologic dating systems.
Example(s):
Please see the individual sub fields for specific examples.

taxonomicCoverage

This element has no default value.

Content of this field:

Derived from: TaxonomicCoverage (by xs:extension)

Elements:	Use:	How many:
system	optional	
scope	optional	document

Description of this field:

Taxonomic Coverage is a container for Taxonomic information about a project, a resource, or an entity within a resource. It includes a list of species names (or higher level ranks) from one or more classification systems.
Example(s):
Please see the individual sub fields for specific examples.

singleDateTime

This element has no default value.

Content of this field:

Type: SingleDateTimeType

Description of this field:

The singleDateTime field is intended to describe a single date and time for an event. There is a choice between two options: a calendar date with a time, or a geologic age.
Example(s):
Please see the individual sub-elements for example.

rangeOfDates

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
beginDate	required	
endDate	required	
)		

Description of this field:

The 'RangeOfDatesType' field is intended to be used for describing a range of dates and/or times. It may be used multiple times to document multiple date ranges. It allows for two 'singleDateTime' fields, the first to be used as the beginning dateTime, and the second to be used as the ending dateTime of the range.
Example(s):
Please see the examples from the 'singleDateTime' field for specific examples.

beginDate

This element has no default value.

Content of this field:

Type: SingleDateTimeType

Description of this field:

A single time stamp signifying the beginning of some time period. There is a choice between two options: a calendar date with a time, or a geologic age.
Example(s):
Please see the individual sub-elements for example.

endDate

This element has no default value.

Content of this field:

Description of this field:

A single time stamp signifying the end of some time

Type: SingleDateTimeType

period. There is a choice between two options: a calendar date with a time, or a geologic age.
Example(s):
Please see the individual sub-elements for example.

calendarDate	This element has no default value.

Content of this field:

Type: res:yearDate

Description of this field:

The calendar date field is used to express a date, giving the year, month, and day. The format should be one that complies with the International Standards Organization's standard 8601. The recommended format for EML is YYYY-MM-DD, where Y is the four digit year, M is the two digit month code (01 - 12, where January = 01), and D is the two digit day of the month (01 - 31). This field can also be used to enter just the year portion of a date.
Example(s):
2001-01-01
2001-10-12
2001
1895

time	This element has no default value.

Content of this field:

Type: xs:time

Description of this field:

The time field is used to express the hour (and optionally minute, or minute and second) of the day for an event, and should comply with the International Standards Organization's standard 8601. The recommended format for EML is hh:mm:ssTZD, where hh is the two digit hour of the day, mm is the two digit minute of the hour, and ss is the two digit second of the minute. TZD stands for Time Zone Designator which is used to handle time zone offsets. Times may be expressed in two ways: 1) UTC (Coordinated Universal Time, also known as Greenwich Mean Time, or GMT), with a special UTC designator ("Z"), 2) local time, together with a time zone offset in hours and minutes. A time zone offset of "+hh:mm" indicates that the date or time uses a local time zone which is "hh" hours and "mm" minutes ahead of UTC. A time zone offset of "-hh:mm" indicates a local time zone which is "hh" hours and "mm" minutes behind UTC.
Example(s):
1) 08:31:22Z , which means eight thirty one and 22 seconds in the morning at Coordinated Universal Time (Greenwich Mean Time). 2) 14:06:09-08:00 , which means six minutes, nine seconds past two o'clock p.m., Pacific Standard Time (which is offset eight hours behind UTC)

alternativeTimeScale	This element has no default value.

Content of this field:

Description of this field:

A name, code, or date describing an event or period in an alternative time scale, for instance as an absolute date calculated using a named dating method, or as a relative date that is drawn from stratigraphy or biostratigraphy. Calendar dates as provided in the ISO 8601 dating system used in the standard CSDGM are not adequate to describe geologic time periods. Absolute geologic time is usually measured in millions of years before present, but may use different units and relative base times. Relative geologic time is measured by subdivisions of the earth's geology. in an order based upon relative age, most commonly, vertical or stratigraphic position. The actual dating systems used in geologic studies often tie relative times measured through stratigraphy or biostratigraphy to a particular absolute time using radioisotope dating techniques, among others. As these methods for absolute dating have improved, the estimates of the dates for strata have changed, consequently, it would be inaccurate to record absolute dates in situations where relative dates were measured. This structure is provided as an optional alternative to the standard calendar dates provided by ISO 8601.

Example(s):

Please see the individual sub-fields for specific examples.

Elements:	**Use:**	**How many:**
A sequence of (		
timeScaleName	required	
timeScaleAgeEstimate	required	
timeScaleAgeUncertainty	optional	
timeScaleAgeExplanation	optional	
timeScaleCitation	optional	unbounded
)		

timeScaleName

This element has no default value.

Content of this field:

Type: **res:NonEmptyStringType**

Description of this field:

Name of a recognized alternative time scale. This includes 'Absolute' as the name of the time scale for measuring geologic dates before the present and names of geologic dating systems that are arrangements of symbols or names in order of relative geologic time.

Example(s):

'Absolute', 'Geomagnetic Polarity Time Scale', 'International Geological Time Scale', 'Oxygen-Isotope'

timeScaleAgeEstimate

This element has no default value.

Content of this field:

Description of this field:

Either an absolute date or a relative age name describing an event or period in an alternative time scale such as the Geologic Time Scale.

Example(s):

For example, '300 Ma' (300 million years before present) is a Geologic_Age_Estimate based on the Absolute Geologic_Time_Scale, 'C28r' is a chron name from the Geomagnetic Polarity Time Scale, and

Type: res:NonEmptyStringType

'Maastrichtian' and 'Jurassic' are names from the International Geological Time Scale. Since different relative geologic time scales are often not aligned, multiple geologic dates may need to be specified. For example, the Geomagnetic Polarity Time Scale chron 'C29r', at the K/T boundary lies in both the 'Maastrichtian' and the 'Danian' stages from the International Geological Time Scale, thus if you were documenting this event using the International Geological Time Scale, both 'Maastrichtian' and 'Danian' should be included here.

timeScaleAgeUncertainty	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The error estimate for the alternative time. This should include the units of measure, a description of what the error estimate represents and how it was calculated.

Example(s):

+/- 5 Ma (Million Years)

timeScaleAgeExplanation	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The name and/or description of the method used to calculate the age estimate. Detailed information about the method may be provided through references contained in the Time Scale Citation field.

timeScaleCitation	This element has no default value.

Content of this field:

Type: cit:CitationType

Description of this field:

Citation for works providing detailed information about any element of the time scale age.

Example(s):

For example, a publication describing the methodology used for carbon dating or describing the basic geologic time scale in more detail could be cited here.

geographicDescription	This element has no default value.

Content of this field:

Description of this field:

geographicDescription is a short text description of a dataset's geographic areal domain. A text description is especially important to provide a geographic setting when the extent of the data set cannot be well described by the "boundingCoordinates", or in the case of data which are not specifically geospatial. Assuming the "boundingCoordinates" do not adequately describe the extent of the data set, the discrepancy can be identified and described here. The coordinates may

define a rectangle around a country, with this geographicDescription element containing a disclaimer and/or further details concerning the border. A study of the diseases of salmon may not have a specific geographic extent associated with it, but the salmon were collected in the states of Washington and Oregon. The "boundingCoordinates" might form a general rectangle around the states of Washington and Oregon, but the "geographicDescription" might describe the fact that the study took place only along certain rivers within those states.

This data element differs from the standard data element "Place_Keyword" in that it allows a free text description of the geographic extent, rather than just a list of words or phrases useful as an index of location names associated with the data set.

This element can also contain information about the collection of the boundingCoordinates, e.g., an altitude value that is referenced to Mean Lower Low Water, or the projection system that the latitude and longitude coordinates were taken from.

Type: res:NonEmptyStringType

Example(s):
"Manistee River watershed"
"extent of 7 1/2 minute quads containing any property belonging to Yellowstone National Park"
"ponds and reservoirs larger than 2 acres in Jefferson County, Colorado".

boundingCoordinates	This element has no default value.

Content of this field:

Description of this field:

Bounding Coordinates are the four margins (N, S, E, W) of a bounding box, or when considered in lat-lon pairs, the corners of the box. These elements are meant to convey general information and are not for accurate mapping. More specific information may be included by using the elements in the spatialReference schema. The limits of coverage of a data set should be expressed as decimal latitudes and longitudes, and in the order western-most, eastern-most, northern-most, and southern-most. By convention, latitudes and longitudes are referenced to the Equator and to the Prime Meridian (the datums), respectively. By definition, the 0 and 180 meridians themselves do not belong in either hemisphere, but local conventions may place them in either. All coordinates are typed as decimals. Since all four elements are required, a bounding area that is a single point should use the same values for northBoundingCoordinate and

Elements:	Use:	How many:
A sequence of (		
westBoundingCoordinate	required	
eastBoundingCoordinate	required	
northBoundingCoordinate	required	
southBoundingCoordinate	required	

boundingAltitudes)	optional	southBoundingCoordinate, and likewise for westBoundingCoordinate and eastBoundingCoordinate. In the case of a data set that comprises all longitudes (e.g., a horizontal band between 2 parallels that fully encompasses the earth), please use a westBoundingCoordinate of -180.0, and an eastBoundingCoordinate of 180.0 (or +180.0). In this case, it could be considered geographically appropriate to specify both values as "180" (or any other meridian), but this could also be interpreted as only the meridian itself, so this is not recommended **Example(s):** Please see the individual sub-fields.

westBoundingCoordinate	**This element has no default value.**
Content of this field:	Description of this field: The westBoundingCoordinate field defines the longitude of the western-most point of the bounding box that is being described. A longitude coordinate is typed as a decimal, i.e., decimal degrees from -180 to 180, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'W' or 'west') are not allowed. Longitudes east of the prime meridian must be specified by a plus sign (+), or by the absence of a minus sign (-), and longitudes west of the meridian shall be prefixed with minus sign (-). In the case of a data set that comprises all longitudes (e.g., a horizontal band between 2 parallels that fully encompasses the earth), please use a westBoundingCoordinate of -180.0, and an eastBoundingCoordinate of 180.0 (or +180.0). In this case, it could be considered geographically appropriate to specify both values as "180" (or any other meridian), but this could also be interpreted as only the meridian itself, so this is not recommended. **Example(s):** -118.25 +25 45.24755

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -180.0
- **Maximum:** 180.0

eastBoundingCoordinate	**This element has no default value.**
Content of this field:	Description of this field:

The eastBoundingCoordinate field defines the longitude of the eastern-most point of the bounding box that is being described. A longitude coordinate is typed as a decimal, i.e., decimal degrees from -180 to 180, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'W' or 'west') are not allowed. Longitudes east of the prime meridian must be specified by a plus sign (+), or by the absence of a minus sign (-), and longitudes west of the meridian shall be prefixed with minus sign (-). In the case of a data set that comprises all longitudes (e.g., a horizontal band between 2 parallels that fully encompasses the earth), please use a westBoundingCoordinate of -180.0, and an eastBoundingCoordinate of 180.0 (or +180.0). In this case, it could be considered geographically appropriate to specify both values as "180" (or any other meridian), but this could also be interpreted as only the meridian itself, so this is not recommended.

Example(s):

-118.25
+25
45.24755

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -180.0
- **Maximum:** 180.0

northBoundingCoordinate	This element has no default value.

Content of this field:

Description of this field:

The northBoundingCoordinate field defines the latitude of the northern-most point of the bounding box that is being described. A latitude coordinate is typed as a decimal, i.e., decimal degrees from -180 to 180, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'N' or north') are not allowed. Latitudes north of the equator must be denoted by a plus sign (+), or by the absence of a minus sign (-), and latitudes south of the equator shall be prefixed with minus sign (-). A location with latitude of +90 (90) or -90 degrees will specify the position at the North or South Pole, respectively.

Example(s):

-18.25
+25

65.24755

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -90.0
- **Maximum:** 90.0

southBoundingCoordinate	This element has no default value.

Content of this field:

Description of this field:

The southBoundingCoordinate field defines the latitude of the southern-most point of the bounding box that is being described. A latitude coordinate is typed as a decimal, i.e., decimal degrees from -180 to 180, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'N' or north') are not allowed. Latitudes north of the equator must be denoted by a plus sign (+), or by the absence of a minus sign (-), and latitudes south of the equator shall be prefixed with minus sign (-). A location with latitude of +90 (90) or -90 degrees will specify the position at the North or South Pole, respectively.

Example(s):

-118.25
+25
84.24755

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -90.0
- **Maximum:** 90.0

boundingAltitudes	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
altitudeMinimum	required	
altitudeMaximum	required	
altitudeUnits	required	
)		

Description of this field:

The bounding altitude field is intended to contain altitudinal (elevation) measurements for the bounding box being described. It allows for minimum and maximum altitude fields, as well as a field for the units of measure. The combination of these fields provide the vertical extent information for the bounding box.

Example(s):

Please see the individual sub-fields for specific examples.

altitudeMinimum	This element has no default value.

Content of this field:

Description of this field:

Type: xs:decimal

The minimum altitude extent of coverage for the bounding box that is being described. The minimum altitude should be in reference to a known datum (e.g., Mean Sea Level), which should be part of the geographicDescription.

Example(s):

100.6

-12

altitudeMaximum

This element has no default value.

Content of this field:

Type: xs:decimal

Description of this field:

The maximum altitude extent of coverage for the bounding box that is being described. The maximum altitude should be in reference to a known datum, which should be part of the geographicDescription.

Example(s):

100.6

-10

altitudeUnits

This element has no default value.

Content of this field:

Type: unit:LengthUnitType

Description of this field:

The unit that the altitude is expressed in. See the description under the Type definition

datasetGPolygon

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
datasetGPolygonOuterGRing	required	
datasetGPolygonExclusionGRing	optional	unbounded
)		

Description of this field:

This construct creates a spatial ring with a hollow center. This doughnut shape is specified by the outer ring (datasetGPolygonOuterRing) and the inner exclusion zone (datasetGPolygonExclusionGRing) which can be thought of as the hole in the center of a doughnut. This is useful for defining areas such as the shores of a pond where you only want to specify the shore excluding the pond itself.

Example(s):

Please see the individual sub-fields for specific examples.

datasetGPolygonOuterGRing

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
gRingPoint	required	unbounded

Description of this field:

The outer containment loop of a datasetGPolygon. This is the outer part of the doughnut shape that encompasses the broadest area of coverage. It can be created either by a gRing (list of points) or 3 or more gRingPoints. See the sub-elements and their Type definitions for more specific information.

This element is generally analogous to the FGDC outer ring although somewhat differently specified. Documentation for an FGDC G-Ring states that 4 points are required to define a

)		
OR		
gRing	required	
)		

polygon, and the first and last should be identical. However this is not enforceable in XML Schema, and so in EML a minimum of 3 <gRingPoint>s is required to define a polygon, and it can be assumed that a polygon is closed by joining the last point to the first. XSL stylesheets that transform EML instances to the FGDC specification should repeat the first gRingPoint node as the last when creating a list of points.

gRingPoint	**This element has no default value.**

Content of this field:

Type: GRingPointType

Description of this field:

A single geographic location. As a child of <datasetGPolygonOuterGRing> a minimum of 3 are required to define a polygon. The polygon is presumed to be closed. Please see the sub elements and the Type description for more information about creating a point location.

gRing	**This element has no default value.**

Content of this field:

Type: GRingType

Description of this field:

A set of ordered pairs of floating-point numbers, The number of points in the string is not enforced by EML. However, authors should note that in order for this field is to be directly translated to FGDC, 4 points should be included in the string. See the Type for more information on constructing the string.

datasetGPolygonExclusionGRing	**This element has no default value.**

Content of this field:

Elements:	**Use:**	**How many:**
A choice of (		
gRingPoint	required	unbounded
OR		
gRing	required	
)		

Description of this field:

the closed nonintersecting boundary of a void area (or hole in an interior area). This is the center of the doughnut shape created by the datasetGPolygon. It can be created either by a gRing (list of points) or one or more gRingPoints. See the sub-elements and their Type definitions for more information.

This element is generally analogous to an FGDC exclusion ring "Data Set G-Polygon Exclusion G-Ring", although it's children are somewhat differently described. Documentation for the FGDC component states that 4 points are required to define a polygon, and the first and last should be identical. However this EML element requires only one point so that a single point can be excluded, presumably, a single station. If multiple single stations are to be excluded, then authors should include multiple <datasetGPolygonExclusionGRing>s.

gRingPoint	This element has no default value.
Content of this field: **Type: GRingPointType**	Description of this field: A single geographic location. This is useful if you register your datasets by a single geospatial point, such as the lat/long of your research station. Please see the sub elements and the Type description for more information on constructing a gRingPoint

gRing	This element has no default value.
Content of this field: **Type: GRingType**	Description of this field: A set of ordered pairs of floating-point numbers, See the Type for more information

gRingLatitude	This element has no default value.
Content of this field:	Description of this field: A latitude coordinate is typed as a decimal, i.e., decimal degrees from -90 to 90, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'S' or 'south') are not allowed. Latitudes north of the equator must be specified by a plus sign (+), or by the absence of a minus sign (-), and latitudes south of the equator shall be prefixed with minus sign (-). **Example(s):** 34.123 -18.25 +78.25

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -90.0
- **Maximum:** 90.0

gRingLongitude	This element has no default value.
Content of this field:	Description of this field: The longitude of a point of the g-ring A longitude coordinate is typed as a decimal, i.e., decimal degrees from -180 to 180, inclusive. Decimal degrees may be expressed to any precision desired. Fractions of a degree in minutes and seconds should be converted to degree fractions. Strings denoting direction or hemisphere (e.g., 'W' or 'west') are not allowed. Longitudes east of the prime meridian must be specified by a plus sign (+), or by the absence of a minus sign (-), and longitudes west of the meridian shall be prefixed with minus sign (-). **Example(s):** -118.25

+25
45.24755

Derived from: xs:decimal (by xs:restriction)

Allowed values:

- **Minimum:** -180.0
- **Maximum:** 180.0

taxonomicSystem	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
classificationSystem	required	unbounded
identificationReference	optional	unbounded
identifierName	required	unbounded
taxonomicProcedures	required	
taxonomicCompleteness	optional	
vouchers	optional	unbounded
)		

Description of this field:

Documentation of taxonomic sources, procedures, and treatments.

classificationSystem	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
classificationSystemCitation	required	
classificationSystemModifications	optional	
)		

Description of this field:

Information about the classification system or authority used.

Example(s):

Flora of North America

classificationSystemCitation	This element has no default value.

Content of this field:

Type: cit:CitationType

Description of this field:

Relevant literature for documenting the used classification system.

classificationSystemModifications	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

A description of any modifications or exceptions made to the classification system or authority used.

identificationReference	This element has no default value.

Content of this field:

Type: cit:CitationType

Description of this field:

Information on any non-authoritative materials (e.g. field guides) useful for reconstructing the actual identification process.

identifierName	This element has no default value.

Content of this field:

Type: rp:ResponsibleParty

Description of this field:

Information about the individual(s) responsible for the identification(s) of the specimens or sightings, etc.

taxonomicProcedures	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Description of the methods used for the taxonomic identification.
Example(s):
specimen processing, comparison with museum materials, keys and key characters, chemical or genetic analyses

taxonomicCompleteness	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Information concerning the proportions and treatment of unidentified materials, estimates of the importance and possible identities of uncertain determinations, synonyms or other incorrect usages, taxa not well treated or requiring further work, and expertise of field workers.
Example(s):
materials sent to experts, and not yet determined

vouchers	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
specimen	required	
repository	required	
)		

Description of this field:

Information on the types of specimen, the repository, and the individuals who identified the vouchers.

specimen	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

A word or phrase describing the type of specimen collected.
Example(s):
herbarium specimens, blood samples, photographs, individuals, or batches

repository	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
originator	required	unbounded
)		

Description of this field:

Information about the curator or contact person and/or agency responsible for the specimens.

originator	This element has no default value.

Content of this field:	Description of this field:
Type: rp:ResponsibleParty	The 'originator' element provides the full name of the person, organization, or position associated with the resource. Typically, the originator role is set to "owner" to indicate the list of parties who "own" the resource, but other roles such as "principal investigator", "author", and "editor" are provided. **Example(s):** Please see the examples within the sub fields for the responsible party.

generalTaxonomicCoverage	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	A description of the range of taxa addressed in the data set or collection. **Example(s):** "All vascular plants were identified to family or species, mosses and lichens were identified as moss or lichen."

taxonomicClassification	This element has no default value.
Content of this field:	Description of this field:
Type: TaxonomicClassificationType	Information about the range of taxa addressed in the data set or collection. See the Type definition for more information.

taxonRankName	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The name of the taxonomic rank for which the Taxon rank value is provided. This field allows for the name one of the accepted levels of Taxa. **Example(s):** 'Kingdom', 'Division/Phylum', 'Class', 'Order', 'Family', 'Genus', and 'Species'

taxonRankValue	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The name representing the taxonomic rank of the taxon being described. The values included may be referenced from an authoritative source such as the Integrated Taxonomic Information System (ITIS)in the U.S. (http://www/itis.usda.gov) and in Canada (http://sis.agr.gc.ca/pls/itisca/taxaget). Also, Species2000 is another source of taxonomic information, found at (http://www.sp2000.org) **Example(s):** Acer would be an example of a genus rank value, and rubrum would be an example of a species rank value, together indicating the common name of red maple. It is recommended to start with Kingdom and

include ranks down to the most detailed level possible.

commonName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Specification of applicable common names. These common names may be general descriptions of a group of organisms if appropriate.

Example(s):

insects, vertebrate, grasses, waterfowl, vascular plants, red maple.

taxonomicClassification	This element has no default value.

Content of this field:

Type: TaxonomicClassificationType

Description of this field:

Information about the range of taxa addressed in the data set or collection. See the Type definition for more information.

Attribute Definitions:

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

id

Type: res:IDType

Use: optional

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

id

Type: res:IDType

Use: optional

Complex Type Definitions:

Coverage

Content of this field:

Elements:	Use:	How many:
A choice of (		
A choice of (		
geographicCoverage	required	
OR		
temporalCoverage	required	
OR		
taxonomicCoverage	required	
)		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

This field is a container for the spatial, temporal and taxonomic coverages that apply to various resources, often dataset resources. Please see the individual descriptions of the sub fields for more detail.

Example(s):

Please see the individual sub fields for specific examples.

TemporalCoverage

Content of this field:

Elements:	Use:	How many:
A choice of (		
A choice of (		
singleDateTime	required	unbounded
OR		
rangeOfDates	required	
)		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	

Description of this field:

The temporal coverage fields are intended to be used in describing the date and time of an event. It allows for three general descriptions: a single date or time, multiple dates or times, and a range of date or times.

Example(s):

Please see the individual sub fields for specific examples.

SingleDateTimeType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
calendarDate	required	
time	optional	
)		
OR		
alternativeTimeScale	required	
)		

Description of this field:

The SingleDateTimeType field is intended to describe a single date and time for an event. There is a choice between two options: a calendar date with a time, or a geologic age.

Example(s):

Please see the individual sub-elements for example.

GeographicCoverage

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
geographicDescription	required	
boundingCoordinates	required	
datasetGPolygon	optional	unbounded
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

Geographic Coverage is a container for spatial information about a a project, a resource, or an entity within a resource. It is meant for general information and not for accurate mapping. More specific information, including mapping projections, is covered by EML in the spatialReference schema.

Example(s):

Please see the individual sub-elements for specific examples.

GRingPointType

Content of this field:

Elements:	Use:	How many:

Description of this field:

A single geographic location. This is useful if you register your datasets by a single geospatial point,

A sequence of (		
gRingLatitude	required	
gRingLongitude	required	
)		

such as the lat/long of your research station.

TaxonomicCoverage

Content of this field:

Elements:	**Use:**	**How many:**
A choice of (		
A sequence of (		
taxonomicSystem	optional	
generalTaxonomicCoverage	optional	
taxonomicClassification	required	unbounded
)		
OR		
res:ReferencesGroup		
)		

Attributes:	**Use:**	**Default Value:**
id	optional	

Description of this field:

Taxonomic Coverage is a container for taxonomic information about a project, a resource, or an entity within a resource.

Example(s):

Please see the individual sub-fields for specific examples.

TaxonomicClassificationType

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
taxonRankName	optional	
taxonRankValue	optional	
commonName	optional	unbounded
taxonomicClassification	optional	unbounded
)		

Description of this field:

Information about the range of taxa addressed in the data set or collection. It is recommended that one provide information starting from the taxonomic rank of kingdom, to a level which reflects the data set or collection being documented. The levels of Kingdom, Division/Phylum, Class, Order, Family, Genus, and Species should be included as ranks as appropriate. Because the taxonomic ranks are hierarchical, the Taxonomic Classification field is self-referencing to allow for an arbitrary depth of rank, down to species.

Example(s):

The Taxonomic Classification field consists of a sequence of 4 fields: taxonomic rank, taxonomic rank value, common name, and finally Taxonomic Classification (self-referencing). Please see the sub-fields for specific examples.

Simple Type Definitions:

GRingType

A set of ordered pairs of floating-point numbers, separated by commas, in which the first number in each pair is the longitude of a point and the second is the latitude of the point. Longitude and latitude are specified in decimal degrees with north latitudes positive and south negative, east

longitude positive and west negative

Note on the relationship to FGDC: This element is generally analogous to the FGDC component for ring, although implemented somewhat differently. Documentation for FGDC states that 4 points are required to define a polygon, and the first and last should be identical, although this is not enforceable in XML Schema. In addition, EML does not enforce any pattern on the string used for a GRingType, so that it may be used differently as a child of EML's <datasetGPolygonOuterGRing> or <datasetGPolygonExclusionGRing> elements. If authors of EML instance documents wish the contents of this element to be directly translated to FGDC, they should comply with the example below when constructing their strings. Alternatively, in most cases, a sequence of gRingPoints can be used in EML instances, which can be processed into content for an FGDC Data Set G-Polygon G-Ring.

Derived from: xs:string (by xs:restriction)

Allowed values:

Example(s):

This is an acceptable gRing:
12, 2.0987 12, -7.5555 34.345,10.40
However, for translation to FGDC, construct your string like this:
-119.453,35.0 -125,37.5555 -122,40 -119.453,35.0

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-dataset

Back to EML Contents

The eml-dataset module - Dataset specific information

The eml-dataset module contains general information that describes dataset resources. It is intended to provide overview information about the dataset: broad information such as the title, abstract, keywords, contacts, maintenance history, purpose, and distribution of the data themselves. The eml-dataset module also imports many other modules that are used to describe the dataset in fine detail. Specifically, it uses the eml-methods module to describe methodology used in collecting or processing the dataset, the eml-project module to describe the overarching research context and experimental design, the eml-access module to define access control rules for the data and metadata, and the eml-entity module to provide detailed information about the logical structure of the dataset. A dataset can be (and often is) composed of a series of data entities (tables) that are linked together by particular integrity constraints.

The eml-dataset module, like other modules, may be "referenced" via the <references> tag. This allows a dataset to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	all datasets
Stand-alone:	yes
Imports:	eml-documentation, eml-resource, eml-party, eml-access, eml-entity, eml-dataTable, eml-project, eml-methods, eml-spatialRaster, eml-spatialVector, eml-storedProcedure, eml-text, eml-view
Imported By:	
View an image of the schema:	**eml-dataset image**

Element Definitions:

dataset	**This element has no default value.**
Content of this field:	Description of this field:
Type: DatasetType	The dataset field encompasses all information about a single dataset. A dataset is defined as all of the information describing a data collection event. This event may take place over some period of time and include many actual collections (a time series or remote sensing application) or it could be just one actual collection (a day in the field).

purpose	**This element has no default value.**
Content of this field:	Description of this field:
Type: txt:TextType	A description of the purpose of this dataset. Note that this element requires DocBook style formatting. See eml-text for more information.

maintenance	**This element has no default value.**
Content of this field:	Description of this field:
Type: MaintenanceType	A description of the maintenance of this data resource. This includes information about the frequency of update, and whether there is ongoing data collection.

contact	**This element has no default value.**
Content of this field:	Description of this field:

Type: rp:ResponsibleParty

The contact field contains contact information for this dataset. This is the person or institution to contact with questions about the use, interpretation of a data set.

publisher	This element has no default value.

Content of this field:

Type: rp:ResponsibleParty

Description of this field:

The publisher of this data set. At times this is a traditional publishing house, but it may also simply be an institution that is making the data available in a published (ie, citable) format.

pubPlace	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The pubPlace field is the location where the resource was published, which may be different from where the resource was created.

Example(s):

San Francisco, CA, USA

New York, NY, USA

methods	This element has no default value.

Content of this field:

Type: md:MethodsType

Description of this field:

The methods field documents scientific methods used in the collection of this dataset. It includes information on items such as tools, instrument calibration and software.

project	This element has no default value.

Content of this field:

Type: proj:ResearchProjectType

Description of this field:

The project field contains information on the project in which this dataset was collected. It includes information such as project personnel, funding, study area, project design and related projects. The project description can also contain documentation on subprojects.

dataTable	This element has no default value.

Content of this field:

Type: dat:DataTableType

Description of this field:

The dataTable field documents the dataTable(s) that make up this dataset. A dataTable could be anything from a Comma Separated Value (CSV) file to a spreadsheet to a table in an RDBMS.

spatialRaster	This element has no default value.

Content of this field:

Type: sr:SpatialRasterType

Description of this field:

The spatialRaster field describes any spatial raster images included in this dataset.

spatialVector	This element has no default value.

Content of this field:

Description of this field:

Type: sv:SpatialVectorType

The spatialVector field describes any spatial vectors included in this dataset.

storedProcedure	This element has no default value.

Content of this field:

Type: sp:StoredProcedureType

Description of this field:

The storedProcedure field contains information about any stored procedures included with this dataset. This usually implies that the dataset is stored in a DBMS or some other data management system capable of processing your dataset.

view	This element has no default value.

Content of this field:

Type: v:ViewType

Description of this field:

The view field contains information about any view included with this dataset. This usually implies that the dataset is stored in a DBMS or some other data management system capable of processing your dataset.

otherEntity	This element has no default value.

Content of this field:

Type: ent:OtherEntityType

Description of this field:

The otherEntity field contains information about any entity in the dataset that is not any of the preceding entities. (i.e. it is not a table, spatialRaster, spatialVector, storedProcedure or view.) OtherEntity allows the documentation of basic entity fields as well as a plain text field to allow you to type your entity.

description	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

A text description of the maintenance of this data resource. Note that this field must be marked up using DocBook like tagging. See eml-text for more information.

maintenanceUpdateFrequency	This element has no default value.

Content of this field:

Type: MaintUpFreqType

Description of this field:

Frequency with which changes and additions are made to the dataset after the initial dataset is completed. The values for this field must come from the enumeration MaintUpFreqType.

changeHistory	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
changeScope	required	
oldValue	required	

Description of this field:

A description of changes made to the data since its release.

changeDate required
comment optional
)

changeScope	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The expression should unambiguously identify the entity(s) and attribute(s) that were changed.

oldValue	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The previous value or an expression that describes the previous value of the data.

changeDate	This element has no default value.
Content of this field: **Type:** xs:date	Description of this field: The date the changes were applied.

comment	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Explanation or justification for the change made to the data.

Attribute Definitions:

id

Type: res:IDType
Use: optional

system

Type: res:SystemType
Use: optional

scope

Type: res:ScopeType
Use: optional
Default value: document

Complex Type Definitions:

DatasetType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
res:ResourceGroup		

Description of this field:

DatasetType is the base type for the dataset element. The dataset field encompasses all information about a single dataset. A dataset is defined as all of the information describing a data collection event. This event may take place over some period of time and

purpose	optional	
maintenance	optional	
contact	required	unbounded
publisher	optional	
pubPlace	optional	
methods	optional	
project	optional	
A choice of (		
dataTable	required	
OR		
spatialRaster	required	
OR		
spatialVector	required	
OR		
storedProcedure	required	
OR		
view	required	
OR		
otherEntity	required	
)		
)		
OR		
res:ReferencesGroup		
)		

include many actual collections (a time series or remote sensing application) or it could be just one actual collection (a day in the field).

Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

MaintenanceType

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
description	required	
maintenanceUpdateFrequency	optional	
changeHistory	optional	unbounded
)		

Description of this field:

The maintenance type defines the fields for the maintenance element.

Simple Type Definitions:

MaintUpFreqType

Derived from: xs:string (by xs:restriction)

Allowed values:

- annually
- asNeeded
- biannually
- continually
- daily
- irregular
- monthly
- notPlanned
- weekly
- unkown
- otherMaintenancePeriod

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-dataTable

Back to EML Contents

The eml-dataTable module - Logical information about data table entities

The eml-dataTable module is used to describe the logical characteristics of each tabular set of information in a dataset. A series of comma-separated text files may be considered a dataset, and each file would subsequently be considered a dataTable entity within the dataset. Since the eml-dataTable module extends the eml-entity module, it uses all of the common entity elements to describe the table, along with a few elements specific to just data table entities. The eml-dataTable module allows for the description of each attribute (column/field/variable) within the data table through the use of the eml-attribute module. Likewise, there are fields used to describe the physical distribution of the data table, its overall coverage, the methodology used in creating the data, and other logical structure information such as its orientation, case sensitivity, etc.

Module details

Recommended Usage:	The EML dataTable Module is used to document datasets with one or more data tables.
Stand-alone:	yes
Imports:	eml-documentation, eml-resource, eml-entity, eml-attribute, eml-constraint
Imported By:	
View an image of the schema:	**eml-dataTable image**

Element Definitions:

dataTable	This element has no default value.
Content of this field: **Type:** DataTableType	Description of this field: The dataTable element is a descriptor of one entity in the dataset identified by its name. This element can contain information about the dataTable's orientation, number of records, case sensitivity, and temporal, geographic and taxonomic coverage. Because the dataTable element refers to the complex type 'DataTableType', it must contain all the elements defined for the type. description for DataTableType to review component element rules.

attributeList	This element has no default value.
Content of this field: **Type:** att:AttributeListType	Description of this field: The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	This element has no default value.
Content of this field: **Type:** con:ConstraintType	Description of this field: Description of any relational constraints on ' this entity. For more information see the eml-constraint module.

caseSensitive	This element has no default value.
Content of this field:	Description of this field: The caseSensitive element specifies text case sensitivity of the data in the dataTable. The valid values are yes or no. If it is set to yes, then values of attributes that differ only in case (e.g., LOW, low) represent distinct values. If set to no, then values of attributes that differ only in case represent the same value.

Example(s):
yes
no

Derived from: xs:string (by xs:restriction)

Allowed values:

- yes
- no

numberOfRecords	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The numberOfRecords element contains a count of the number of records in the dataTable. This is typically an integer value, and only includes records that represent observations. It would not include any details of physical formatting such as the number of header lines (see eml-physical for that information). **Example(s):** 975

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

DataTableType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
ent:EntityGroup		
attributeList	required	
constraint	optional	unbounded
caseSensitive	optional	

numberOfRecords	optional

)

OR

res:ReferencesGroup

)

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Simple Type Definitions:

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-entity

Back to EML Contents

The eml-entity module - Entity level information within datasets

The eml-entity module defines the logical characteristics of each entity in the dataset. Entities are usually tables of data (eml-dataTable). Data tables may be ascii text files, relational database tables, spreadsheets or other type of tabular data with a fixed logical structure. Related to data tables are views (eml-view) and stored procedures (eml-storedProcedure). Views and stored procedures are produced by an RDBMS or related system. Other types of data such as: raster (eml-spatialRaster), vector (eml-spatialVector) or spatialReference image data are also data entities. An otherEntity element would be used to describe types of entities that are not described by any other entity type.

The eml-entity module, like other modules, may be "referenced" via the <references> tag. This allows an entity document to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	This module is used to describe data entities.
Stand-alone:	Only when 'otherEntity' is used.
Imports:	eml-documentation, eml-coverage, eml-physical, eml-methods, eml-attribute, eml-resource, eml-constraint, eml-text
Imported By:	
View an image of the schema:	**eml-entity image**

Element Definitions:

otherEntity	This element has no default value.
Content of this field: **Type:** OtherEntityType	Description of this field: The other entity element is a descriptor of a not-otherwise-defined entity in the dataset, identified by its name. The element can contain information about the entity's basic identity, its temporal, geographic and taxonomic coverage, and its type. **Example(s):** Photograph of rocky intertidal plot 12 from Santa Cruz Island

attributeList	This element has no default value.
Content of this field: **Type:** att:AttributeListType	Description of this field: The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	This element has no default value.
Content of this field: **Type:** con:ConstraintType	Description of this field: Description of any relational constraints on ' this entity. For more information see the eml-constraint module.

entityType	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The entityType field contains the name of the entity's type. The entity's type is typically the name of the type of data represented in the entity, such as "photograph". This field is used only if this is an 'other' entity and you want to specify

the kind of "other" entity this is.
Example(s):
Photograph

alternateIdentifier	This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
system	optional	

Description of this field:

An additional, secondary identifier for this entity. The primary identifier belongs in the "id" attribute, but additional identifiers that are used to label this entity, possibly from different data management systems, can be listed here.
Example(s):
VCR3465

entityName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The name identifies the entity in the dataset: file name, name of database table, etc.
Example(s):
SpeciesAbundance1996

entityDescription	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Text generally describing the entity, its type, and relevant information about the data in the entity.
Example(s):
Species abundance data for 1996 at the VCR LTER site

physical	This element has no default value.

Content of this field:

Type: phys:PhysicalType

Description of this field:

Information on the physical format of this entity. Each logical entity can be serialized in one or more physical formats, which can be described here. For each physical format, provide a "physical" element that describes the format and how to obtain that version. Two physical elements MUST describe the same entity precisely (i.e., after obtaining and parsing the physical data stream described under each physical element, one MUST end with an identical logical entity). For more information see the eml-physical module.

coverage	This element has no default value.

Content of this field:

Type: cov:Coverage

Description of this field:

Information on the geographic, spatial and temporal coverages used in this entity. Please see the eml-coverage module for more information.

methods	This element has no default value.

Content of this field:

Description of this field:

Information on the specific methods used to collect information in this entity. Please see the eml-methods

Type: md:MethodsType

module for more information.
Example(s):

additionalInfo	This element has no default value.
Content of this field: **Type:** txt:TextType	Description of this field: This field provides any information that is not characterized by the other entity metadata fields. **Example(s):** Multiple sampling events represented

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

system

Type: res:SystemType

Use: optional

The information management system within which this identifier has relevance. Generally, the identifier would be unique within the "system" and would be sufficient to retrieve the entity from the system. The system is often a URL or URI that identifies the main entry point for the information management system.
Example(s):
http://knb.ecoinformatics.org/knb/

Complex Type Definitions:

OtherEntityType

Content of this field:

Description of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
EntityGroup		
A sequence of (		
attributeList	optional	
constraint	optional	unbounded

entityType	required

)

)

OR

res:ReferencesGroup

)

Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

Group Definitions:

EntityGroup

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
alternateIdentifier	optional	unbounded
entityName	required	
entityDescription	optional	
physical	optional	unbounded
coverage	optional	
methods	optional	
additionalInfo	optional	unbounded

)

Description of this field:

The EntityGroup defines the common structure for descriptions of any type of entity, including the name and attributes of the entity. The term entity is used in the sense of an entity in a relational model. The most common entity is a table, something with columns and rows, but can also represent an image or other type of data. See 'eml-attribute' for descriptions of the required attribute fields.

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-literature

Back to EML Contents

The eml-literature module - Citation specific information

The eml-literature module contains information that describes literature resources. It is intended to provide overview information about the literature citation, including title, abstract, keywords, and contacts. Citation types follow the conventions laid out by **EndNote**, and there is an attempt to represent a compatible subset of the EndNote citation types. These citation types include: article, book, chapter, edited book, manuscript, report, thesis, conference proceedings, personal communication, map, generic, audio visual, and presentation. The "generic" citation type would be used when one of the other types will not work.

The eml-literature module, like other modules, may be "referenced" via the <references> tag. This allows a citation to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	All datasets with literary citations
Stand-alone:	yes
Imports:	eml-documentation, eml-resource, eml-coverage, eml-party, eml-access, eml-project
Imported By:	
View an image of the schema:	**eml-literature image**

Element Definitions:

citation	**This element has no default value.**
Content of this field: **Type: CitationType**	Description of this field: The citation element contains general information about a literature resource that is being documented, or a piece of literature that is being cited in support of a given resource, such as a dataset. It contains sub-elements that are specific to a literature resource such as a book, a journal article, a thesis, etc. It extends the generic resource elements with literature specific fields.

contact	**This element has no default value.**
Content of this field: **Type: rp:ResponsibleParty**	Description of this field: The contact field contains information about an alternate person to be contacted about this citation. Usually, the first author serves as the contact for a citation resource, e.g., a reprint request. In some cases, an alternate individual(s) may serve that function, and can be indicated here. Since contact is of the type rp:ResponsibleParty, a reference may be used.

article	**This element has no default value.**
Content of this field: **Type: Article**	Description of this field: The article field provides sub-fields for a full citation of an article in a journal or other periodical.

book	**This element has no default value.**
Content of this field: **Type: Book**	Description of this field: The book field provides sub-fields for a full citation of a

book.

chapter	This element has no default value.
Content of this field: **Type:** Chapter	Description of this field: The book chapter allows citation of a single chapter or section of a book. The "creator" for a book chapter are the chapter's authors, while the "editor" is the book editors. Likewise, "title" is the chapter title, while "bookTitle" is the title of the whole book.

editedBook	This element has no default value.
Content of this field: **Type:** Book	Description of this field: The edited book represents a book which was edited by one or more editors, but whose chapters were possibly authored by others. The editors of an edited book should be listed in the "creator" field.

manuscript	This element has no default value.
Content of this field: **Type:** Manuscript	Description of this field: The manuscript field provides sub-fields for a full citation of an unpublished manuscript.

report	This element has no default value.
Content of this field: **Type:** Report	Description of this field: The report may be self published by the institution or through a publisher. They usually are available by request to the institution or can be purchased from the publisher.

thesis	This element has no default value.
Content of this field: **Type:** Thesis	Description of this field: Information about a thesis that has been written as part of a degree requirement and is frequently published in small numbers by the degree awarding institution.

conferenceProceedings	This element has no default value.
Content of this field: **Type:** ConferenceProceedings	Description of this field: The published notes, papers, presentations, etc..., of a conference.

personalCommunication	This element has no default value.
Content of this field: **Type:** PersonalCommunication	Description of this field: This could be a widely distributed memo, an e-mail, a transcript from a conversation or interview, etc...

map	This element has no default value.
Content of this field: **Type:** Map	Description of this field: This element describes the map that is being cited or cataloged, including its geographic coverage and scale.

generic	This element has no default value.
Content of this field: **Type:** Generic	Description of this field: This reference type was created for references that do not fit in to the other existing reference types.

audioVisual	This element has no default value.
Content of this field: **Type:** AudioVisual	Description of this field: This reference type is meant to cover all forms of audio and visual media, including film, video, broadcast, other electronic media.

presentation	This element has no default value.
Content of this field: **Type:** Presentation	Description of this field: An unpublished presentation from a conference, workshop, workgroup, symposium, etc. It will be provided upon request in either in paper and/or electronic form. If the presentation was actually published in a proceedings, use the conferenceProceedings type instead.

journal	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The name of the journal, magazine, newspaper, zine, etc... in which the article was published. **Example(s):** "Ecology" "New York Times" "Harper's"

volume	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The volume field is used to describe the volume of the journal in which the article appears. **Example(s):** "Volume I"

issue	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The issue field is used to describe the issue of the journal in which the article appears. **Example(s):** November 2001

pageRange	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The page range field is used for the beginning and ending pages of the journal article that is being documented. **Example(s):**

115-122

publisher	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The organization that physically puts together the article and publishes it. **Example(s):** Harper Collins University Of California Press

publicationPlace	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The location at which the work was published. This is usually the name of the city in which the publishing house produced the work. **Example(s):** New York London

ISSN	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The ISSN, or International Standard Serial Number that has been assigned to this literature resource. **Example(s):** ISSN 1234-5679

publisher	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The organization that physically puts together the book and publishes it. **Example(s):** Harper Collins University Of California Press

publicationPlace	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The location at which the work was published. This is usually the name of the city in which the publishing house produced the work. **Example(s):** New York London

edition	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The edition field is to document the edition of the book that is being described. **Example(s):** Second Edition

volume	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The volume field is used to describe the volume number of a book that is part of a multi-volume series of books. **Example(s):** Volume 2

numberOfVolumes	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Number of volumes in a collection **Example(s):** 12

totalPages	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total pages field is used to describe the total number of pages in the book that is being described. **Example(s):** 628

totalFigures	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: the total figures field is used to describe the total number of figures, diagrams, and plates in the book that is being documented. **Example(s):** 45

totalTables	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total tables field is used to describe the total number of tables that are present in the book that is being documented. **Example(s):** 10

ISBN	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The ISBN, or International Standard Book Number that has been assigned to this literature resource. **Example(s):** ISBN 1-861003-11-0

chapterNumber	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The chapter number of the chapter of a book that is being described. **Example(s):** 7

editor	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The book editor field is used to document the name of the editor of the book that is being described. The editor may be a person, organization, or a role within an organization. **Example(s):** Tom Christiansen Institute of Marine Science Publication Manager

bookTitle	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The book title field is used to document the title of the book that is being described. **Example(s):** War and Peace

pageRange	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The page range field is used to document the beginning and ending pages of a chapter in a book. **Example(s):** 25-122

conferenceName	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The name of the conference whose proceedings have been published. **Example(s):** North American Science Symposium

conferenceDate	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The date the conference was held. **Example(s):** November 1-6, 1998

conferenceLocation	This element has no default value.
Content of this field: **Type:** rp:Address	Description of this field: The location where the conference was held.

institution	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The institution information field is used to provide contact and address information that would be needed to request an unpublished manuscript **Example(s):**

Please see the individual sub-fields for specific examples.

totalPages	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total pages field is used to describe the total number of pages in the manuscript that is being described. **Example(s):** 628

reportNumber	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The report number field is used to describe the unique identification number that has been issued by the report institution for the report being described. **Example(s):** 22

publisher	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The organization that physically put together the report and publishes it. **Example(s):** Harper Collins University Of California Press

publicationPlace	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The location at which the work was published. This is usually the name of the city in which the publishing house produced the work. **Example(s):** New York London

totalPages	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total pages field is used to describe the total number of pages in the report that is being described. **Example(s):** 628

publisher	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The organization that physically puts together the communication and publishes it. **Example(s):** Harper Collins University Of California Press

publicationPlace	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The location at which the work was published. This is usually the name of the city in which the publishing house produced the work. **Example(s):** New York London

communicationType	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The type of personal communication. Could be an email, letter, memo, transcript of conversation either hardcopy or online. **Example(s):** memo letter email

recipient	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The person, place or thing the personal communication was sent to. **Example(s):** Schmedley, Joe jschmedley@lternet.edu

publisher	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The organization that physically puts together the map and publishes it. **Example(s):** Harper Collins

edition	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The edition field is to document the edition of the map that is being described. **Example(s):** Second Edition

geographicCoverage	This element has no default value.
Content of this field: **Type:** cov:GeographicCoverage	Description of this field: This element describes the geographic area which the map covers. Could be descriptive text or Cartesian coordinates of the area.

scale	This element has no default value.
Content of this field:	Description of this field:

Type: res:NonEmptyStringType

The Map's scale
Example(s):
1:25,000

publisher	This element has no default value.

Content of this field:

Type: rp:ResponsibleParty

Description of this field:

Organization which actually distributes the video, film, the broadcaster etc...
Example(s):
LTER Network Office
Public Broadcasting
Pacifica Radio

publicationPlace	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The location at which the work was published. This is usually the name of the city in which the publishing house produced the work.
Example(s):
New York
London

performer	This element has no default value.

Content of this field:

Type: rp:ResponsibleParty

Description of this field:

The performers involved in acting, narrating, or shown in the audio visual production.
Example(s):
Jim Nabors
Sir Lawrence Olivier

ISBN	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The ISBN, or International Standard Book Number that has been assigned to this literature resource.
Example(s):
ISBN 1-861003-11-0

publisher	This element has no default value.

Content of this field:

Type: rp:ResponsibleParty

Description of this field:

The organization which physically puts together the reference and publishes it.
Example(s):
Harper Collins
University Of California Press

publicationPlace	This element has no default value.

Content of this field:

Description of this field:

The location at which the work was published. This is usually the name of the city in which the publishing house produced

Type: res:NonEmptyStringType

the work.
Example(s):
New York
London

referenceType	This element has no default value.

Content of this field:

Description of this field:

The reference type describes the type of reference this generic type is being used to represent.
Example(s):
zine
film
radio program

volume	This element has no default value.

Content of this field:

Description of this field:

The volume field is used to describe the volume number of a reference that is part of a multi-volume series of references.

Type: res:NonEmptyStringType

Example(s):
Volume 2

numberOfVolumes	This element has no default value.

Content of this field:

Description of this field:

Number of volumes in a collection

Type: res:NonEmptyStringType

Example(s):
"12"

totalPages	This element has no default value.

Content of this field:

Description of this field:

The total pages field is used to describe the total number of pages in the references that is being described.

Type: res:NonEmptyStringType

Example(s):
628

totalFigures	This element has no default value.

Content of this field:

Description of this field:

The total figures field is used to describe the total number of figures, diagrams, and plates in the reference that is being documented.

Type: res:NonEmptyStringType

Example(s):
45

totalTables	This element has no default value.

Content of this field:

Description of this field:

The total tables field is used to describe the total number of tables that are present in the reference that is being documented.

Type: res:NonEmptyStringType

Example(s):
10

edition	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The edition field is to document the edition of the generic reference type that is being described. **Example(s):** Second Edition

originalPublication	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Supplemental information about the original publication of the current reference. **Example(s):** Date Publisher

reprintEdition	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Reference for current edition that was originally published under a different title. **Example(s):** Stream Research in the LTER Network, 1993

reviewedItem	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Use for articles, chapters, audio visual, etc. that are critical review of books, cinema, art, or other works. **Example(s):** Structure and Function of an Alpine Ecosystem Niwot Ridge, Colorado Edited by WILLIAM D. BOWMAN and TIMOTHY R. SEASTEDT, University of Colorado, Boulder

ISBN	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The ISBN, or International Standard Book Number that has been assigned to this literature resource. **Example(s):** ISBN 1-861003-11-0

ISSN	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The ISSN, or International Standard Serial Number that has been assigned to this literature resource. **Example(s):** ISSN 1234-5679

degree	This element has no default value.
Content of this field:	Description of this field: The degree field is used to describe the name or degree level

Type: res:NonEmptyStringType	for which the thesis was completed. **Example(s):** Ph.D. M.S. Master of Science

institution	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The degree institution field is used to name the institution from which the degree was awarded for the thesis being described. **Example(s):** Western Washington University

totalPages	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total pages field is used to document the number of pages that are present in the thesis that is being described. **Example(s):** 356

conferenceName	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The name of the conference at which this presentation was given. **Example(s):** North American Science Symposium

conferenceDate	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The date the conference was held. **Example(s):** November 1-6, 1998

conferenceLocation	This element has no default value.
Content of this field: **Type:** rp:Address	Description of this field: The location where the conference was held.

Attribute Definitions:

id
Type: res:IDType **Use:** optional

system
Type: res:SystemType **Use:** optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

CitationType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
res:ResourceGroup		
contact	optional	unbounded
A choice of (		
article	required	
OR		
book	required	
OR		
chapter	required	
OR		
editedBook	required	
OR		
manuscript	required	
OR		
report	required	
OR		
thesis	required	
OR		
conferenceProceedings	required	
OR		
personalCommunication	required	
OR		
map	required	
OR		
generic	required	
OR		
audioVisual	required	
OR		
presentation	required	
)		
)		
OR		

res:ReferencesGroup

)

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Article

Content of this field:

Elements:	Use:	How many:
A sequence of (		
journal	required	
volume	optional	
issue	optional	
pageRange	optional	
publisher	optional	
publicationPlace	optional	
ISSN	optional	
)		

Description of this field:

The article field provides sub-fields for a full citation of an article in a journal or other periodical.

Book

Content of this field:

Elements:	Use:	How many:
A sequence of (		
publisher	required	
publicationPlace	optional	
edition	optional	
volume	optional	
numberOfVolumes	optional	
totalPages	optional	
totalFigures	optional	
totalTables	optional	
ISBN	optional	
)		

Description of this field:

The book field provides sub-fields for a full citation of a book.

Chapter

Content of this field:

Derived from: **Book** (by xs:extension)

Derived from: **Book** (by xs:extension)

Elements:	Use:	How many:
A sequence of (		
chapterNumber	optional	
editor	required	unbounded
bookTitle	required	

Description of this field:

The book chapter allows citation of a single chapter or section of a book. The "creator" for a book chapter are the chapter's authors, while the "editor" is the book editors. Likewise, "title" is the chapter title, while "bookTitle" is the title of the whole book.

pageRange	optional

)

ConferenceProceedings

Content of this field:

Derived from: **Chapter** (by xs:extension)

Derived from: **Chapter** (by xs:extension)

Elements:	**Use:**	**How many:**
A sequence of (		
conferenceName	optional	
conferenceDate	optional	
conferenceLocation	optional	

)

Description of this field:

The published notes, papers, presentations, etc..., of a conference.

Manuscript

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
institution	required	unbounded
totalPages	optional	

)

Description of this field:

The manuscript field provides sub-fields for a full citation of an unpublished manuscript.

Report

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
reportNumber	optional	
publisher	optional	
publicationPlace	optional	
totalPages	optional	

)

Description of this field:

The report may be self published by the institution or through a publisher. They usually are available by request to the institution or can be purchased from the publisher.

PersonalCommunication

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
publisher	optional	
publicationPlace	optional	
communicationType	optional	
recipient	optional	unbounded

)

Description of this field:

This could be a widely distributed memo, an e-mail, a transcript from a conversation or interview, etc...

Map

Content of this field:

Elements:	**Use:**	**How many:**
A sequence of (		

Description of this field:

This element describes the map that is being cited or cataloged, including its geographic coverage and scale.

publisher	optional	
edition	optional	
geographicCoverage	optional	unbounded
scale	optional	

)

AudioVisual

Content of this field:

Description of this field:

This reference type is meant to cover all forms of audio and visual media, including film, video, broadcast, other electronic media.

Elements:	**Use:**	**How many:**
A sequence of (		
publisher	required	
publicationPlace	optional	unbounded
performer	optional	unbounded
ISBN	optional	

)

Generic

Content of this field:

Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
publisher	required	
publicationPlace	optional	
referenceType	optional	
volume	optional	
numberOfVolumes	optional	
totalPages	optional	
totalFigures	optional	
totalTables	optional	
edition	optional	
originalPublication	optional	
reprintEdition	optional	
reviewedItem	optional	
A choice of (		
ISBN	required	
OR		
ISSN	required	
)		
)		

Thesis

Content of this field:

Description of this field:

Information about a thesis that has been written as part of a degree requirement and is frequently published in small numbers by the degree awarding institution.

Elements:	**Use:**	**How many:**
A sequence of (		
degree	required	
institution	required	
totalPages	optional	

)

Presentation	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A sequence of (		
conferenceName	optional	
conferenceDate	optional	
conferenceLocation	optional	
)		

An unpublished presentation from a conference, workshop, workgroup, symposium, etc. It will be provided upon request in either in paper and/or electronic form by contacting the authors. If the presentation was actually published in a proceedings, use the conferenceProceedings type instead.

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-methods

Back to EML Contents

The eml-methods module - Methodological information for resources

The eml-methods module describes the methods followed in the creation of the dataset, including description of field, laboratory and processing steps, sampling methods and units, quality control procedures. The eml-methods module is used to describe the actual procedures that are used in the creation or the subsequent processing of a dataset. Likewise, eml-methods is used to describe processes that have been used to define / improve the quality of a data file, or to identify potential problems with the data file. Note that the eml-protocol module is intended to be used to document a prescribed procedure, whereas the eml-method module is used to describe procedures that were actually performed. The distinction is that the use of the term "protocol" is used in the "prescriptive" sense, and the term "method" is used in the "descriptive" sense. This distinction allows managers to build a protocol library of well-known, established protocols (procedures), but also document what procedure was truly performed in relation to the established protocol. The method may have diverged from the protocol purposefully, or perhaps incidentally, but the procedural lineage is still preserved and understandable.

Module details

Recommended Usage:	All datasets
Stand-alone:	no
Imports:	eml-documentation, eml-dataset, eml-resource, eml-software, eml-protocol, eml-party, eml-text, eml-coverage, eml-literature
Imported By:	
View an image of the schema:	**eml-methods image**

Element Definitions:

methods

This element has no default value.

Content of this field:

Type: MethodsType

Description of this field:

methodStep

This element has no default value.

Content of this field:

Derived from: ProcedureStepType (by xs:extension)

Elements:	Use:	How many:
A sequence of (		
dataSource	optional	unbounded
)		

Description of this field:

The methodStep field allows for repeated sets of elements that document a series of procedures followed to produce a data object. These include text descriptions of the procedures, relevant literature, software, instrumentation, source data and any quality control measures taken.

Example(s):

Please see the examples for the sub-fields.

dataSource

This element has no default value.

Content of this field:

Type: ds:DatasetType

Description of this field:

A source of data used by this methodStep.

sampling

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
studyExtent	required	
samplingDescription	required	

Description of this field:

Description of sampling procedures including the geographic, temporal and taxonomic coverage of the study. See individual elements for more detailed descriptions.

spatialSamplingUnits	optional	
citation	optional	unbounded

)

studyExtent	**This element has no default value.**

Content of this field:

Elements:	**Use:**	**How many:**
A choice of (		
coverage	required	
OR		
description	required	
)		

Description of this field:

The field studyExtent represents both a specific sampling area and the sampling frequency (temporal boundaries, frequency of occurrence). The geographic studyExtent is usually a surrogate (representative area of) for the larger area documented in the "studyAreaDescription". The studyExtent can be entered either in non-structured textual form or using the structure of the coverage element.

coverage	**This element has no default value.**

Content of this field:

Type: **cov:Coverage**

Description of this field:

The field studyExtent represents both a specific sampling area and the sampling frequency (temporal boundaries, frequency of occurrence). The geographic studyExtent is usually a surrogate (representative area of) for the larger area documented in the "studyAreaDescription". The studyExtent can be entered either in non-structured textual form or using the structure of the coverage element. See eml-coverage for more information.

description	**This element has no default value.**

Content of this field:

Type: **txt:TextType**

Description of this field:

The coverage field allows for a textual description of the specific sampling area, the sampling frequency (temporal boundaries, frequency of occurrence), and groups of living organisms sampled (taxonomic coverage).

Example(s):

The study was conducted on the North Platte River, starting 6 miles downstream and ending 9 miles downstream of the route 132 bridge in Evanston, ND.

samplingDescription	**This element has no default value.**

Content of this field:

Type: **txt:TextType**

Description of this field:

The samplingDescription field allows for a text-based/human readable description of the sampling procedures used in the research project. The content of this element would be similar to a description of sampling procedures found in the methods section of a journal article.

spatialSamplingUnits	**This element has no default value.**

Content of this field:

Description of this field:

A spatial sampling unit describes the specific geographic areas sampled. In the case of a study in which the measurements from several disbursed point collection devices are aggregated, then the sampling unit would be the

Elements:	Use:	How many:
A choice of (		
referencedEntityId	required	
OR		
coverage	required	
)		

area of that aggregation. Spatial sampling units can either be described by filling out the structured coverage element or by reference to the values in a data table (usually a GIS layer)

Example(s):

If a researcher places a single light source at a specific point in a research location in order to attract insects to derive an estimate of the insect population, then the sampling unit is the area illuminated by the light source (in actual practice there might be multiple sampling units in this case since different species have different attraction rates).

The bounding box of a specific 3-meter square plot.

The location of a weather station.

referencedEntityId	This element has no default value.

Content of this field:

Description of this field:

A value of a referencedEntityId element is a reference to the identifier of the entity module that provides the metadata for a data table (RDBMS, GIS or ascii text) that has the actual spatial sampling unit values. The referencedEntityId field is an indirect pointer to the actual values. The referencedEntityId can be thought of as a foreign key in a relational database.

Example(s):

x

coverage	This element has no default value.

Content of this field:

Type: **cov:GeographicCoverage**

Description of this field:

Structured description of each sampling unit location

citation	This element has no default value.

Content of this field:

Type: **cit:CitationType**

Description of this field:

The citation field allows to either reference a literature resource or enter structured literature information

qualityControl	This element has no default value.

Content of this field:

Type: **ProcedureStepType**

Description of this field:

The qualityControl field provides a location for the description of actions taken to either control or assess the quality of data resulting from the associated method step. A quality control description should identify a quality goal and describe prescriptive steps taken to ensure that the data meet those standards and/or postscriptive steps taken to assess the extent to which they are met. A quality control statement is associated with the methodStep that could have affected the targeted quality goal.

description	This element has no default value.

Content of this field:

Description of this field:

The description field allows for repeated text that describes

Type: txt:TextType	the methodology for a project, experiment, or particular data table or to describe the steps taken to control or assure the quality of the data. Likewise, a literature citation may be provided that describes the methodology that was employed. Or the information my be provided by either referencing a protocol resource or entering the structured protocol information **Example(s):** 1.Collect tissues from algae of interest. a. We are currently collecting Egregia menziezii, Mazzaella splendens, M. flaccida, Hedophyllum sessile, Postelsia palmaeformis and Fucus gardneri.We stopped collecting Neorhodomela larix and Odonthalia floccosa because they can be heavily fouled and we feared that would skew the results. b. We collect a 7-10 cm blade or branch from each plant. For Egregia, try to sample small, young plants or take the base of the blade. For Postelsia, take a few of the blades. The other plants are small enough so a whole blade can be taken.

citation	This element has no default value.
Content of this field:	Description of this field:
Type: cit:CitationType	The citation field allows to either reference a literature resource or enter structured literature information

protocol	This element has no default value.
Content of this field:	Description of this field:
Type: pro:ProtocolType	The protocol field is used to either reference a protocol resource or describe methods and identify the processes that have been used to define / improve the quality of a data file, also used to identify potential problems with the data file.

instrumentation	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The Instrumentation field allows the description of any instruments used in the data collection or quality control and quality assurance. The description should include vendor, model number, optional equipment, etc. **Example(s):** LACHAT analyzer, model XYX.

software	This element has no default value.
Content of this field:	Description of this field:
Type: sw:SoftwareType	The software element allows reference to any software used to process data.

subStep	This element has no default value.
Content of this field:	Description of this field:
Type: ProcedureStepType	This fields allows the nesting of additional method steps within this step. This is useful for hierarchical method descriptions.

Complex Type Definitions:

MethodsType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A sequence of (		
methodStep	required	unbounded
sampling	optional	
qualityControl	optional	unbounded
)		

ProcedureStepType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A sequence of (		
A sequence of (		
description	required	
A choice of (		
citation	required	
OR		
protocol	required	
)		
)		
instrumentation	optional	unbounded
software	optional	unbounded
subStep	optional	unbounded
)		

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-party

Back to EML Contents

The eml-party module - People and organization information

The eml-party module describes a responsible party and is typically used to name the creator of a resource or metadata document. A responsible party may be an individual person, an organization or a named position within an organization. The eml-party module contains detailed contact information. It is used throughout the other EML modules where detailed contact information is needed.

The eml-party module, like other modules, may be "referenced" via the <references> tag. This allows a party to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	all datasets
Stand-alone:	yes
Imports:	eml-documentation, eml-resource
Imported By:	
View an image of the schema:	**eml-party image**

Element Definitions:

individualName	**This element has no default value.**
Content of this field:	Description of this field:
Type: Person	The individualName field contains subfields so that a person's name can be broken down into parts. Note that the the content model for the containing type allows a sequence of choices for the first element(s): <individualName>, <organizationName> and/or <positionName>. This means that a parent element (e.g., creator) may use combinations of the 3 sub-elements to make up a single logical party. For example, a creator with only the individualName of 'Joe Smith' is NOT the same as a creator with the individualName of 'Joe Smith' and the organizationName of 'NSF'. To include both a positionName and an organizationName as children of a <contact> implies that anyone currently occupying that positionName at that organizationName is an appropriate contact. The positionName should not be used in conjunction with individualName unless only that specific individual at that position would be considered appropriate for that designation. **Example(s):** Because this is an 'elementOnly' field, please look at the examples for the subfields 'givenName' and 'surName'.

organizationName	**This element has no default value.**
Content of this field:	Description of this field:
	The responsible party field contains the full name of the organization that is associated with the resource. This field is intended to describe which institution or overall

organization is associated with the resource being described.

Note that the the content model for the containing type allows a sequence of choices for the first element(s): <individualName>, <organizationName> and/or <positionName>. This means that a parent element (e.g., creator) may use combinations of the 3 sub-elements to make up a single logical party. For example, a creator with only the individualName of 'Joe Smith' is NOT the same as a creator with the individualName of 'Joe Smith' and the organizationName of 'NSF'. To include both a positionName and an organizationName as children of a <contact> implies that anyone currently occupying that positionName at that organizationName is an appropriate contact. The positionName should not be used in conjunction with individualName unless only that specific individual at that position would be considered appropriate for that designation.

Type: res:NonEmptyStringType

Example(s):
National Center for Ecological Analysis and Synthesis

positionName	This element has no default value.

Content of this field:

Description of this field:

This field is intended to be used instead of a particular person or full organization name. If the associated person who holds the role changes frequently, then Position Name would be used for consistency.

Note that the the content model for the containing type allows a sequence of choices for the first element(s): <individualName>, <organizationName> and/or <positionName>. This means that a parent element (e.g., creator) may use combinations of the 3 sub-elements to make up a single logical party. For example, a creator with only the individualName of 'Joe Smith' is NOT the same as a creator with the individualName of 'Joe Smith' and the organizationName of 'NSF'. To include both a positionName and an organizationName as children of a <contact> implies that anyone currently occupying that positionName at that organizationName is an appropriate contact. The positionName should not be used in conjunction with individualName unless only that specific individual at that position would be considered appropriate for that designation.

Type: res:NonEmptyStringType

Example(s):
Niwot Ridge Data Manager

address	This element has no default value.

Content of this field:

Description of this field:

Type: Address

The address field is a container for multiple subfields that describe the physical or electronic address of the responsible party for a resource.
Example(s):
Please see the subfield examples.

phone	This element has no default value.

Content of this field:

Description of this field:

Attributes:	Use:	Default Value:
phonetype	optional	voice

The phone field describes information about the responsible party's telephone, be it a voice phone, fax, or TTD/TTY type telephone. This field contains an attribute used to identify the type.
Example(s):
805-555-2500

electronicMailAddress	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

The electronic mail address is the email address for the party. It is intended to be an Internet SMTP email address, which should consist of a username followed by the @ symbol, followed by the email server domain name address. Other address types are allowable.
Example(s):
my-email@mydomain.edu

onlineUrl	This element has no default value.

Content of this field:

Description of this field:

Type: xs:anyURI

A link to associated online information, usually a web site. When the party represents an organization, this is the URL to a website or other online information about the organization. If the party is an individual, it might be their personal web site or other related online information about the party.
Example(s):
http://www.yourdomain.edu/~doe

userId	This element has no default value.

Content of this field:

Description of this field:

Attributes:	Use:	Default Value:
directory	required	

An identifier that links this party to a directory of personnel. Although specific contact information for a party might change, the underlying correspondence to a real individual does not. This identifier provides a pointer within a personnel directory that may contain further, and possibly more current, information about the party.
Example(s):
uid=jtown,o=NCEAS,dc=ecoinformatics,dc=org

salutation	This element has no default value.

Content of this field:

Description of this field:

The salutation field is used in addressing an individual with a particular title, such as Dr., Ms., Mrs., Mr., etc.

Type: res:NonEmptyStringType

Example(s):
Dr.

givenName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The given name field can be used for first name of the individual associated with the resource, or for any other names that are not intended to be alphabetized, (as appropriate). Note that while it is possible to include all given names in one field (as in the example below), it may be not be good practice to do so. For example, if an XSL transformation stylesheet were to abbreviate the content of a givenName to just the first initial, a givenName element that contained more than one name would not be transformed correctly.

Example(s):
Juan Luis
Jane

surName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The surname field is used for the last name of the individual associated with the resource. This is typically the family name of an individual, for example, the name by which s/he is referred to in citations.

Example(s):
San Gil
Curtis-Ainsworth
Tao

deliveryPoint	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The delivery point field is used for the physical address for postal communication. This field is used to accommodate the many different international conventions that are the equivalent to a U.S. 'street address'.

Example(s):
7209 Coast Drive, Building 44

city	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The city field is used for the city name of the contact associated with a particular resource.

Example(s):
San Francisco

administrativeArea	This element has no default value.

Content of this field:

Description of this field:

The administrative area field is the equivalent of a 'state' in the U.S., or Province in Canada. This field is intended to

Type: res:NonEmptyStringType	accommodate the many types of international administrative areas. **Example(s):** Colorado

postalCode	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The postal code is equivalent to a U.S. zip code, or the number used for routing to an international address. The U.S. postal code should include the 5 digit code plus the 4 digit extension. **Example(s):** 93106-2231

country	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The country field is used for the name of the contact's country. **Example(s):** U.S.A.

party	This element has no default value.
Content of this field: **Type:** ResponsibleParty	Description of this field: The responsible party contains multiple subfields that are used to describe a person, organization, or position within an organization. It is intended to be used to fully document contact information for many types of associations, such as owner, manager, steward, curator, etc.

Attribute Definitions:

phonetype	
Type: xs:string **Use:** optional **Default value:** voice	This attribute gives the type of phone to which this number applies. By default, this is assumed to be of type "voice", but other possibilities include "facsimile" and "tdd".

directory	
Type: xs:string **Use:** required	This attribute names the directory system to which this userId applies. This will generally be a URL that shows how to look up information, for example an LDAP url. However, it could also be a non-parsable description of the directory system if that is all that is available. **Example(s):** ldap:///ldap.ecoinformatics.org/dc=ecoinformatics,dc=org

id	
Type: res:IDType **Use:** optional	

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

ResponsibleParty

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
A choice of (		
individualName	required	
OR		
organizationName	required	
OR		
positionName	required	
)		
address	optional	unbounded
phone	optional	unbounded
electronicMailAddress	optional	unbounded
onlineUrl	optional	unbounded
userId	optional	unbounded
)		
OR		
res:ReferencesGroup		
)		

Description of this field:

The ResponsibleParty Type contains elements that are used to describe the person, organization or position within an organization that is associated in some way with the resource. It is intended to be used to fully document contact information for many types of associations, such as owner, manager, steward, curator, etc.

Note that the content model for a responsible party type allows a sequence of choices for the first element(s): <individualName>, <organizationName> and/or <positionName>. This means that a parent element (e.g., creator) may use combinations of the 3 sub-elements to make up a single logical party. For example, a creator with only the individualName of 'Joe Smith' is NOT the same as a creator with the individualName of 'Joe Smith' and the organizationName of 'NSF'. To include both a positionName and an organizationName as children of a <contact> implies that anyone currently occupying that positionName at that organizationName is an appropriate contact. The positionName should not be used in conjunction with individualName unless only that specific individual at that position would be considered appropriate for that designation.

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Example(s):
Please see the examples for the particular subfields.

Person

Content of this field:

Elements:	Use:	How many:
A sequence of (		
salutation	optional	unbounded
givenName	optional	unbounded
surName	required	
)		

Description of this field:

The person Type is used to enter the salutation, and two types of name parts for an individual associated with the resource. It uses these three subfields to help parse the person's entire name.

The two elements, <givenName> and <surName>, allow parsing of many types of names, even though distinct elements do not exist for concepts like "middle name" and "compound surname". <givenName> should be used for parts of the name that are often shortened to a first initial, or are not used for ordering, and typically includes first and middle names. The <surName> field is intended to be used for the part of the name that is generally displayed in its entirety and/or is alphabetized or otherwise ordered when appropriate. Note that only one <surName> is allowed, and is required, while <givenName>s are optional and unbounded.

The arrangement and content of the sub-elements is entirely up to the EML document's author, who presumably has first-hand knowledge of how the names are to be constructed. For example, if element position is important (e.g., the list of a book's authors), then EML authors should put the creators in that order. If it is appropriate for a resource to have its creators sorted alphabetically, then the EML author should construct the name parts so that the<surName> field may be used for this purpose. At this time, EML is not able to express cultural conventions so that authors may indicate the correct order for <givenName>s and <surName> when the whole name is expressed. However support for international names is under consideration for a future version of EML, along with other internationalization features.

Example(s):
Please see the examples within each subfield.

Address

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
deliveryPoint	optional	unbounded
city	optional	

Description of this field:

The address field is provides detailed information for communicating with a party contact via electronic mail or postal mail, including the physical delivery location.

Example(s):
Please see the examples for each subfield

administrativeArea	optional
postalCode	optional
country	optional

)

OR

res:ReferencesGroup

)

Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

Simple Type Definitions:

RoleType

The role code field provides information on how a person or organization is related to a resource. There may be many people associated, including an 'originator' of a dataset, an 'author', 'editor', or 'publisher' of a literature resource, or an organization that is a 'distributor'. the full list of choices is included in the example.

Example(s):

author, contentProvider, custodianSteward, distributor, editor, metadataProvider, originator, pointOfContact, principalInvestigator, processor, publisher, or user.

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-physical

Back to EML Contents

The eml-physical module - Physical file format

The eml-physical module describes the external and internal physical characteristics of a data object as well as the information required for its distribution. Examples of the external physical characteristics of a data object would be the filename, size, compression, encoding methods, and authentication of a file or byte stream. Internal physical characteristics describe the format of the data object being described. Both named binary or otherwise proprietary formats can be cited (e.g., Microsoft Access 2000), or text formats can be precisely described (e.g., ASCII text delimited with commas). For these text formats, it also includes the information needed to parse the data object to extract the entity and its attributes from the data object. Distribution information describes how to retrieve the data object. The retrieval information can be either online (e.g., a URL or other connection information) or offline (e.g., a data object residing on an archival tape).

The eml-physical module, like other modules, may be "referenced" via the <references> tag. This allows a physical document to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	Any data object that is being described by EML needs this information so the entities and attributes that reside with in the data object can be extracted.
Stand-alone:	yes
Imports:	eml-documentation, eml-literature, eml-resource, eml-access
Imported By:	
View an image of the schema:	**eml-physical image**

Element Definitions:

physical	This element has no default value.
Content of this field: **Type:** PhysicalType	Description of this field: The content model for physical is a CHOICE between "references" and all of the elements that let you describe the internal/external characteristics and distribution of a data object (e.g., dataObject, dataFormat, distribution.) A physical element can contain a reference to an physical element defined elsewhere. Using a reference means that the referenced physical is identical, not just in name but identical in its complete description.

objectName	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The name of the data object. This is possibly distinct from the entity name in that one physical object can contain multiple entities, even though that is not a recommended practice. The objectName often is the filename of a file in a file system or that is accessible on the network. **Example(s):** rainfall-sev-2002-10.txt

size	This element has no default value.
Content of this field: **Attributes:** **Use:** **Default Value:**	Description of this field: This element contains information of the physical size of the entity, by default represented in bytes unless the unit attribute is provided to change the units.

unit	optional	byte	**Example(s):** 134

authentication	**This element has no default value.**
Content of this field:	Description of this field:
Attributes: **Use:** **Default Value:** **method** optional	This element describes authentication procedures or techniques, typically by giving a checksum value for the object. The method used to compute the authentication value (e.g., MD5) is listed in the method attribute. **Example(s):** f5b2177ea03aea73de12da81f896fe40

compressionMethod	**This element has no default value.**
Content of this field: **Type:** **res:NonEmptyStringType**	Description of this field: This element lists a compression method used to compress the object, such as zip, compress, etc. Compression and encoding methods must be listed in the order in which they were applied, so that decompression and decoding should occur in the reverse order of the listing. For example, if a file is compressed using zip and then encoded using MIME base64, the compression method would be listed first and the encoding method second. **Example(s):** zip gzip compress

encodingMethod	**This element has no default value.**
Content of this field: **Type:** **res:NonEmptyStringType**	Description of this field: This element lists a encoding method used to encode the object, such as base64, BinHex, etc. Compression and encoding methods must be listed in the order in which they were applied, so that decompression and decoding should occur in the reverse order of the listing. For example, if a file is compressed using zip and then encoded using MIME base64, the compression method would be listed first and the encoding method second. **Example(s):** base64 uuencode binhex

characterEncoding	**This element has no default value.**
Content of this field: **Type:** **res:NonEmptyStringType**	Description of this field: This element contains the name of the character encoding. This is typically ASCII or UTF-8, or one of the other common encodings. **Example(s):** UTF-8

dataFormat	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
textFormat	required	
OR		
externallyDefinedFormat	required	
OR		
binaryRasterFormat	required	
)		

Description of this field:

This element is the parent which is a CHOICE between four possible internal physical formats which describe the internal physical characteristics of the data object. Using this information the user should be able parse physical object to extract the entity and its attributes. Note that this is the format of the physical object itself.

textFormat	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
numHeaderLines	optional	
numFooterLines	optional	
recordDelimiter	optional	unbounded
physicalLineDelimiter	optional	unbounded
numPhysicalLinesPerRecord	optional	
maxRecordLength	optional	
attributeOrientation	required	
A choice of (		
simpleDelimited	required	
OR		
complex	required	
)		
)		

Description of this field:

Description of a text formatted object. The description includes detailed parsing instructions for extracting attributes from the bytestream for simple delimited file formats (e.g., CSV), fixed format files that use fixed columns for attribute locations, and mixtures of the two. It also supports records that span multiple lines.

numHeaderLines	This element has no default value.

Content of this field:

Type: xs:int

Description of this field:

Number of header lines preceding data. Lines are determined by the physicalLineDelimiter, or if it is absent, by the recordDelimiter. This value indicated the number of header lines that should be skipped before starting to parse the data.

Example(s):

4

numFooterLines	This element has no default value.

Content of this field:

Type: xs:int

Description of this field:

Number of footer lines following data. Lines are determined by the physicalLineDelimiter, or if it is absent, by the recordDelimiter. This value indicated the number of footer lines that should be skipped after parsing the data. If this value is omitted, parsers should assume the

data continues to the end of the data stream.
Example(s):
4

recordDelimiter	This element has no default value.
Content of this field: **Type: xs:string**	Description of this field: This element specifies the record delimiter character when the format is text. The record delimiter is usually a linefeed (\n) on UNIX, a carriage return (\r) on MacOS, or both (\r\n) on Windows/DOS. Multiline records are usually delimited with two line ending characters, for example on UNIX it would be two linefeed characters (\n\n). As record delimiters are often non-printing characters, one can use either the special value "\n" to represent a linefeed (ASCII 0x0a) and "\r" to represent a carriage return (ASCII 0x0d). Alternatively, one can use the hex value to represent character values (e.g., 0x0a). **Example(s):** \n\r

physicalLineDelimiter	This element has no default value.
Content of this field: **Type: xs:string**	Description of this field: This element specifies the physical line delimiter character when the format is text. The line delimiter is usually a linefeed (\n) on UNIX, a carriage return (\r) on MacOS, or both (\r\n) on Windows/DOS. Multiline records are usually delimited with two line ending characters, for example on UNIX it would be two linefeed characters (\n\n). As line delimiters are often non-printing characters, one can use either the special value "\n" to represent a linefeed (ASCII 0x0a) and "\r" to represent a carriage return (ASCII 0x0d). Alternatively, one can use the hex value to represent character values (e.g., 0x0a). If this value is not provided, processors should assume that the physical line delimiter is the same as the record delimiter. **Example(s):** \n\r

numPhysicalLinesPerRecord	This element has no default value.
Content of this field: **Type: xs:unsignedInt**	Description of this field: A single logical data record may be written over several physical lines in a file, with no special marker to indicate the end of a record. In such cases, it is necessary to know the number of lines per record in order to correctly read them. If this value is not provided, processors should assume that records are wholly contained on one physical line. If the value is greater than 1, then processors should examine the lineNumber field for each attribute to determine which line of the record contains the information.

Example(s):
3

maxRecordLength	This element has no default value.

Content of this field:

Type: xs:unsignedLong

Description of this field:

The maximum number of characters in any record in the physical file. For delimited files, the record length varies and this is not particularly useful. However, for fixed format files that do not contain record delimiters, this field is critical to tell processors when one record stops and another begins.

Example(s):
597

attributeOrientation	This element has no default value.

Content of this field:

Derived from: xs:string (by xs:restriction)

Allowed values:

- column
- row

Description of this field:

Specifies whether the attributes described in the physical stream are found in columns or rows. The valid values are column or row. If set to 'column', then the attributes are in columns. If set to 'row', then the attributes are in rows. Row orientation is rare, but some systems such as SPlus and R utilize it. For example, some data with column orientation: DATE PLOT SPECIES 2002-01-15 hfr5 acer rubrum 2002-01-15 hfr5 acer xxxx The same data in a rowMajor table: DATE 2002-01-15 PLOT hfr5 SPECIES acer rubrum acer xxxx

Example(s):
column
row

simpleDelimited	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
fieldDelimiter	required	unbounded
collapseDelimiters	optional	
quoteCharacter	optional	unbounded
literalCharacter	optional	unbounded
)		

Description of this field:

A simple delimited format that uses one of a series of delimiters to indicate the ends of fields in the data stream. More complex formats such as fixed format or mixed delimited and fixed formats can be described using the "complex" element.

fieldDelimiter	This element has no default value.

Content of this field:

Description of this field:

This element specifies a character to be used in the object for indicating the ending column for an attribute.

Type: xs:string

The delimiter character itself is not part of the attribute value, but rather is present in the column following the last character of the value. Typical delimiter characters include commas, tabs, spaces, and semicolons. The only time the fieldDelimiter character is not interpreted as a delimiter is if it is contained in a quoted string (see quoteCharacter) or is immediately preceded by a literalCharacter. Non-printable quote characters can be provided as their hex values, and for tab characters by its ASCII string "\t". Processors should assume that the field starts in the column following the previous field if the previous field was fixed, or in the column following the delimiter from the previous field if the previous field was delimited.

Example(s):

,
\t
0x09
0x20

collapseDelimiters	This element has no default value.

Content of this field:

Description of this field:

The collapseDelimiters element specifies whether sequential delimiters should be treated as a single delimiter or multiple delimiters. An example is when a space delimiter is used; often there may be several repeated spaces that should be treated as a single delimiter, but not always. The valid values are yes or no. If it is set to yes, then consecutive delimiters will be collapsed to one. If set to no or absent, then consecutive delimiters will be treated as separate delimiters. Default behaviour is no; hence, consecutive delimiters will be treated as separate delimiters, by default.

Example(s):

yes
no

Derived from: xs:string (by xs:restriction)

Allowed values:

- yes
- no

quoteCharacter	This element has no default value.

Content of this field:

Description of this field:

This element specifies a character to be used in the object for quoting values so that field delimiters can be used within the value. This basically allows delimiter "escaping". The quoteChacter is typically a " or '. When a processor encounters a quote character, it should not interpret any following characters as a delimiter until a matching quote character has been encountered (i.e.,

Type: res:NonEmptyStringType

quotes come in pairs). It is an error to not provide a closing quote before the record ends. Non-printable quote characters can be provided as their hex values.
Example(s):
"
'

literalCharacter	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

This element specifies a character to be used for escaping special character values so that they are treated as literal values. This allows "escaping" for special characters like quotes, commas, and spaces when they are intended to be used in an attribute value rather than being intended as a delimiter. The literalCharacter is typically a \.
Example(s):
\

complex	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
textFixed	required	
OR		
textDelimited	required	
)		

Description of this field:

A complex text format that can describe delimited fields, fixed width fields, and mixtures of the two. This supports multiline records (where one record is distributed across multiple physical lines). When using the complex format, the number of textFixed and textDelimited elements should exactly equal the number of attributes that have been described for the entity, and the order of the textFixed and textDelimited elements should correspond to the order of the attributes as described in the entity. Thus, for a delimited file with fourteen attributes, one should provide exactly fourteen textDelimited elements.

textFixed	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
fieldWidth	required	
lineNumber	optional	
fieldStartColumn	optional	
)		

Description of this field:

Describes the physical format of data sequences that use a fixed number of characters in a specified position in the stream to locate attribute values. This method is common in sensor-derived data and in legacy database systems. To parse it, one must know the number of characters for each attribute and the starting column and line to begin reading the value.

fieldWidth	This element has no default value.

Content of this field:

Type: xs:unsignedLong

Description of this field:

Fixed width fields have a set length, thus the end of the field can always be determined by adding the fieldWidth to the starting column number.
Example(s):
7

lineNumber	

	This element has no default value.
Content of this field: **Type: xs:unsignedLong**	Description of this field: A single logical data record may be written over several physical lines in a file, with no special marker to indicate the end of a record. In such cases, the relative location of a data field must be indicated by both relative row and column number. The lineNumber should never greater that the number of physical lines per record. **Example(s):** 3

fieldStartColumn	This element has no default value.
Content of this field: **Type: xs:long**	Description of this field: Fixed width fields have a set length, thus the end of the field can always be determined by adding the fieldWidth to the starting column number. If the starting column is not provided, processors should assume that the field starts in the column following the previous field if the previous field was fixed, or in the column following the delimiter from the previous field if the previous field was delimited. **Example(s):** 58

textDelimited	This element has no default value.
Content of this field: **Elements:** **Use:** **How many:** A sequence of (**fieldDelimiter** required **collapseDelimiters** optional **lineNumber** optional **quoteCharacter** optional unbounded **literalCharacter** optional unbounded)	Description of this field: Describes the physical format of data sequences that use delimiters in the stream to locate attribute values. This method is common in data exported from spreadsheets and database systems, To parse it, one must know the character that indicates the end of each attribute and the line to begin reading the value.

fieldDelimiter	This element has no default value.
Content of this field: **Type: xs:string**	Description of this field: This element specifies a character to be used in the object for indicating the ending column for an attribute. The delimiter character itself is not part of the attribute value, but rather is present in the column following the last character of the value. Typical delimiter characters include commas, tabs, spaces, and semicolons. The only time the fieldDelimiter character is not interpreted as a delimiter is if it is contained in a quoted string (see quoteCharacter) or is immediately preceded by a literalCharacter. Non-printable quote characters can be provided as their hex values, and for tab characters by its ASCII string "\t". Processors should assume that the field starts in the column following the previous field if the

previous field was fixed, or in the column following the delimiter from the previous field if the previous field was delimited.

Example(s):

,
\t
0x09
0x20

collapseDelimiters	This element has no default value.

Content of this field:

Description of this field:

The collapseDelimiters element specifies whether sequential delimiters should be treated as a single delimiter or multiple delimiters. An example is when a space delimiter is used; often there may be several repeated spaces that should be treated as a single delimiter, but not always. The valid values are yes or no. If it is set to yes, then consecutive delimiters will be collapsed to one. If set to no or absent, then consecutive delimiters will be treated as separate delimiters. Default behaviour is no; hence, consecutive delimiters will be treated as separate delimiters, by default.

Example(s):

yes
no

Derived from: xs:string (by xs:restriction)

Allowed values:

- yes
- no

lineNumber	This element has no default value.

Content of this field:

Description of this field:

A single logical data record may be written over several physical lines in a file, with no special marker to indicate the end of a record. In such cases, the relative location of a data field must be indicated by both relative row and column number. The lineNumber should never be greater that the number of physical lines per record. When parsing the first field on a physical line as a delimited field, they should assume that the field data starts in the first column. Otherwise, follow the rules indicated under fieldDelimiter.

Type: xs:unsignedLong

Example(s):

3

quoteCharacter	This element has no default value.

Content of this field:

Description of this field:

This element specifies a character to be used in the object for quoting values so that field delimiters can be

used within the value. This basically allows delimiter "escaping". The quoteChacter is typically a " or '. When a processor encounters a quote character, it should not interpret any following characters as a delimiter until a matching quote character has been encountered (i.e., quotes come in pairs). It is an error to not provide a closing quote before the record ends. Non-printable quote characters can be provided as their hex values.

Type: res:NonEmptyStringType

Example(s):

"

'

literalCharacter	This element has no default value.

Content of this field:

Description of this field:

This element specifies a character to be used for escaping special character values so that they are treated as literal values. This allows "escaping" for special characters like quotes, commas, and spaces when they are intended to be used in an attribute value rather than being intended as a delimiter. The literalCharacter is typically a \.

Type: res:NonEmptyStringType

Example(s):

\

externallyDefinedFormat	This element has no default value.

Content of this field:

Description of this field:

Information about a non-text or proprietary formatted object. The description names the format explicitly, but assumes a processor implicitly knows how to parse that format to extract the data. A format version can be included. This is mainly used for proprietary formats, including binary files like Microsoft Excel and text formats like ESRI's ArcInfo export format. This is not a recommended way to permanently archive data because the software to parse the format is unlikely to be available over extended periods, but is included to allow for commonly used physical formats.

Elements:	Use:	How many:
A sequence of (		
formatName	required	
formatVersion	optional	
citation	optional	
)		

formatName	This element has no default value.

Content of this field:

Description of this field:

Name of the format of the data object

Type: res:NonEmptyStringType

Example(s):

Microsoft Excel

formatVersion	This element has no default value.

Content of this field:

Description of this field:

Version of the format of the data object

Type: res:NonEmptyStringType

Example(s):

2000 (9.0.2720)

citation	This element has no default value.

Content of this field:

Type: cit:CitationType

Description of this field:

Citation providing more detail about the physical format, including parsing information or information about the software required for reading the object.

binaryRasterFormat

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
rowColumnOrientation	required	
multiBand	optional	
nbits	required	
byteorder	required	
skipbytes	optional	
bandrowbytes	optional	
totalrowbytes	optional	
bandgapbytes	optional	
)		

Description of this field:

The binaryRasterInfo element is a container for various parameters used to described the contents of binary raster image files. In this case, it is based on a white paper on the ESRI site that describes the header information used for BIP and BIL files ("Extendable Image Formats for ArcView GIS 3.1 and 3.2").

rowColumnOrientation

This element has no default value.

Content of this field:

Derived from: xs:string (by xs:restriction)

Allowed values:

- column
- row

Description of this field:

Specifies whether the data should be read across rows or down columns. The valid values are column or row. If set to 'column', then the data are read down columns. If set to 'row', then the data are read across rows.

Example(s):

column

row

multiBand

This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
nbands	required	
layout	required	
)		

Description of this field:

Information needed to properly interpret a multiband image.

nbands

This element has no default value.

Content of this field:

Type: xs:int

Description of this field:

The number of spectral bands in the image. Must be greater than 1.

Example(s):

layout	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The organization of the bands in the image file. Acceptable values are bil - Band interleaved by line. bip - Band interleaved by pixel. bsq - Band sequential. **Example(s):** bil bip bsq

nbits	This element has no default value.
Content of this field: **Type: xs:int**	Description of this field: The number of bits per pixel per band. Acceptable values are typically 1, 4, 8, 16, and 32. The default value is eight bits per pixel per band. For a true color image with three bands (R, G, B) stored using eight bits for each pixel in each band, nbits equals eight and nbands equals three, for a total of twenty-four bits per pixel. **Example(s):** 8

byteorder	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The byte order in which values are stored. The byte order is important for sixteen-bit and higher images, that have two or more bytes per pixel. Acceptable values are little-endian (common on Intel systems like PCs) and big-endian (common on Motorola platforms). **Example(s):** little-endian big-endian

skipbytes	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The number of bytes of data in the image file to skip in order to reach the start of the image data. This keyword allows you to bypass any existing image header information in the file. The default value is zero bytes. **Example(s):** 0

bandrowbytes	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The number of bytes per band per row. This must be an integer. This keyword is used only with BIL files when there are extra bits at the end of each band within a row that must be skipped. **Example(s):**

totalrowbytes	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The total number of bytes of data per row. Use totalrowbytes when there are extra trailing bits at the end of each row. **Example(s):** 8

bandgapbytes	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The number of bytes between bands in a BSQ format image. The default is zero. **Example(s):** 1

distribution	This element has no default value.
Content of this field: **Type:** PhysicalDistributionType	Description of this field: This element provides information on how the resource is distributed. Connections to online systems can be described as URLs or as a list of connection parameters. Please see the Type definition for complete information.

online	This element has no default value.
Content of this field: **Type:** PhysicalOnlineType	Description of this field: Information for a resource that is distributed online. Please see the Type definition for complete information.

offline	This element has no default value.
Content of this field: **Type:** res:OfflineType	Description of this field: Information for a resource that is distributed offline. Please see the Type definition for complete information.

inline	This element has no default value.
Content of this field: **Type:** res:InlineType	Description of this field: Information for a resource that is distributed inline, i.e., along with the metadata. Please see the Type definition for complete information.

access	This element has no default value.
Content of this field: **Type:** acc:AccessType	Description of this field: When this element occurs in a distribution module, it controls access only to the resource being described by the same distribution parent. Please see the Type definition for complete information on constructing an access tree.

onlineDescription	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The onlineDescription element can hold a brief description of the content of the online element's online|offline|inline child. This description element could supply content for an html anchor tag.

url	This element has no default value.
Content of this field: **Type:** res:UrlType	Description of this field: The URL of the resource that is available online. Please see the Type definition for complete information.

connection	This element has no default value.
Content of this field: **Type:** res:ConnectionType	Description of this field: A connection to a resource that is available online. Please see the Type definition for complete information.

Attribute Definitions:

unit	
Use: optional **Default value:** byte	This element gives the unit of measurement for the size of the entity, and is by default a byte. **Example(s):** byte

method	
Type: xs:string **Use:** optional	This element names the method used to calculate and authentication checksum that can be used to validate a bytestream. Typical checksum methods include MD5 and CRC. **Example(s):** MD5

id
Type: res:IDType **Use:** optional

system
Type: res:SystemType **Use:** optional

scope
Type: res:ScopeType **Use:** optional **Default value:** document

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

PhysicalType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
objectName	required	
size	optional	
authentication	optional	unbounded
A choice of (		
compressionMethod	required	
OR		
encodingMethod	required	
)		
characterEncoding	optional	
dataFormat	required	
distribution	optional	unbounded
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

The eml-physical module describes the physical characteristics of a data object and the information required for its distribution. External physical characteristics include the filename, size, compression, encoding methods, and authentication of a file or byte stream. Internal physical characteristics describe the format of the data object. Proprietary formats can be cited (e.g., Microsoft Access 2000), or text formats can be precisely described (e.g., ASCII text delimited with commas). The module includes the information needed to parse the text data object to extract the entity and its attributes. Distribution information describes how to retrieve the data object, either as online (a URL or connection definition), offline (e.g., a data object residing on an archival tape), or inline (i.e., the data are included with the metadata).

Like many other EML elements, a physical Type can contain a reference to another physical element defined elsewhere in the document instead of a description of the resource. Using a reference means that the referenced physical is identical, not just in name but identical in its complete description.

PhysicalDistributionType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
A choice of (		

Description of this field:

The PhysicalDistributionType contains the information required for retrieving the resource.

It differs from the **res:DistributionType** :

Generally, the PhysicalDisribtutionType is intended

online	required	
OR		
offline	required	
OR		
inline	required	
)		
access	optional	
)		
OR		
res:ReferencesGroup		
)		
Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

for download whereas the Type at the resource level is intended primarily for information.

The phys:PhysicalDistributionType includes an optional access tree which can be used to override access rules applied at the resource level. Access for the documents included entities can then be managed individually.

Also see individual sub elements for more information.

PhysicalOnlineType

Content of this field:

Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
onlineDescription	optional	
A choice of (		
url	required	
OR		
connection	required	
)		
)		

Distribution information for accessing the resource online, represented either as a URL or as the series of named parameters needed to connect. The URL field can contain a simple web address or an entire query string. The connection element allows the components of a complex protocol to be described individually.

The PhysicalOnlineType differs from the **res:OnlineType** in that this type only allows a connectionDefinition to appear as the child of a connection. In other words, in a PhysicalOnlineType, the connectionDefinition cannot be abstracted, and must be included as part of an actual connection.

Simple Type Definitions:

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-project

Back to EML Contents

The eml-project module - Research context information for resources

The eml-project module describes the research context in which the dataset was created, including descriptions of over-all motivations and goals, funding, personnel, description of the study area etc. This is also the module to describe the design of the project: the scientific questions being asked, the architecture of the design, etc. This module is used to place the dataset that is being documented into its larger research context.

The eml-project module, like other modules, may be "referenced" via the <references> tag. This allows a research project to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	Use eml-project to document the research context of any dataset or project.
Stand-alone:	no
Imports:	eml-documentation, eml-resource, eml-party, eml-coverage, eml-literature, eml-text
Imported By:	
View an image of the schema:	**eml-project image**

Element Definitions:

researchProject	This element has no default value.
Content of this field: **Type:** ResearchProjectType	Description of this field: The root element of this module. This is used for testing or if you want to instantiate a stand-alone project file.

title	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: A descriptive title for the research project. **Example(s):** Species diversity in Tennessee riparian habitats.

personnel	This element has no default value.
Content of this field: **Derived from:** rp:ResponsibleParty (by xs:extension) **Elements:** **Use:** **How many:** A sequence of (**role** required)	Description of this field: The Personnel field extends ResponsibleParty with role information and is used to document people involved in a research project by providing contact information and their role in the project. A project must have at least one originator.

role	This element has no default value.
Content of this field:	Description of this field: The role field contains information about role a person plays in a research project. There are a number of suggested roles, however, it is possible to add a role if the suggested roles are not adequate. **Example(s):** author contentProvider

Type: rp:RoleType

custodianSteward
distributor
editor
metadataProvider
originator
owner
pointOfContact
principalInvestigator
processor
publisher
user
fieldStationManager
informationManager

abstract	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

Descriptive abstract that summarizes information about the research project.

funding	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

The funding field is used to provide information about funding sources for the project such as: grant and contract numbers; names and addresses of funding sources. Other funding-related information may also be included.

studyAreaDescription	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
descriptor	required	
OR		
citation	optional	
OR		
coverage	optional	
)		

Description of this field:

The studyAreaDescription field documents the physical area associated with the research project. It can include descriptions of the geographic, temporal, and taxonomic coverage of the research location and descriptions of domains (themes) of interest such as climate, geology, soils or disturbances or reference to citable biological or geophysical classification systems such as the Bailey Ecoregions or the Holdridge Life Zones.

descriptor	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
descriptorValue	required	unbounded
citation	optional	unbounded
)		

Attributes:	Use:	Default Value:
name	required	

Description of this field:

The descriptor field is used to document domains (themes) of interest such as climate, geology, soils or disturbances or references to citable biological or geophysical classification systems such as the Bailey Ecoregions or the Holdridge Life Zones.

citableClassificationSystem required

descriptorValue	This element has no default value.

Content of this field:

Attributes: **Use:** **Default Value:**

name_or_id optional

Description of this field:

The descriptorValue field contains the value of a descriptor, describing some aspect of the study area. This may either be a general description in textual form or the value part of a "name/value" pair where the name is entered in the attribute "name_or_id". For example, if the value of the "name" attribute" of the element "descriptor" is "climate", and the value of the attribute "name_or_id" of the element "descriptorValue" is "Annual Precipitation" then the value of this element could be "12.5 inches".

Example(s):

12.5 inches

tundra-forest

citation	This element has no default value.

Content of this field:

Type: **cit:CitationType**

Description of this field:

A citation for this descriptor.

citation	This element has no default value.

Content of this field:

Type: **cit:CitationType**

Description of this field:

The citation for this descriptor.

coverage	This element has no default value.

Content of this field:

Type: **cov:Coverage**

Description of this field:

The coverage of this descriptor.

designDescription	This element has no default value.

Content of this field:

Elements: **Use:** **How many:**

A choice of (

description required

OR

citation optional

)

Description of this field:

The field designDescription contains general textual descriptions of research design. It can include detailed accounts of goals, motivations, theory, hypotheses, strategy, statistical design, and actual work. Literature citations may also be used to describe the research design.

description	This element has no default value.

Content of this field:

Type: **txt:TextType**

Description of this field:

The field designDescription contains general textual descriptions of research design. It can include detailed accounts of goals, motivations, theory, hypotheses, strategy, statistical design, and actual work.

citation	This element has no default value.

Content of this field:

Description of this field:

The citation field is a citation to literature that describes

Type: cit:CitationType

elements of the research design, such as goals, motivations, theory, hypotheses, strategy, statistical design, and actual work.

relatedProject	This element has no default value.

Content of this field:

Type: ResearchProjectType

Description of this field:

This field is a recursive link to another project. This allows projects to be nested under one another for the case where one project spawns another.

Attribute Definitions:

name_or_id

Type: xs:string

Use: optional

The name_or_id field is the name part of a name/value pair of a descriptor; or ID portion of a classification, if applicable. The values of biogeophysical classification systems, e.g. Bailey-Ecoregions, often take the form of an ID or Code along with a text representation. For example, the ID/Code M131 refers to the phrase "Open Woodland - Tundra". M131 is an unambiguous reference to a more detailed description. If one is using a published classification system then there should be a corresponding citation to the source, e.g., Bailey,R.G., 1996 "Ecosystem Geography".

Example(s):

M131

Average Annual Rainfall

name

Type: DescriptorType

Use: required

The name of the descriptor system. The name can be either a theme such as climate or hydrology, or the name of a citable classification system.

citableClassificationSystem

Type: xs:boolean

Use: required

This boolean attribute defines whether this descriptor comes from a citable classification system or not.

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

ResearchProjectType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
title	required	unbounded
personnel	required	unbounded
abstract	optional	
funding	optional	
studyAreaDescription	optional	
designDescription	optional	
relatedProject	optional	unbounded
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

The researchProject complex type describes the structure for documenting the research context of a dataset or another project. It can include research goals, motivations, theory, hypotheses, etc., as well as a description of research efforts that form the basis for other work. (To document methods specific to a dataset use eml-methods.) This field can be associated with a dataset using the project field of eml-dataset, and can be associated with another project using the relatedProject field of eml-project (this module).

Simple Type Definitions:

DescriptorType

The DescriptorType is used to represent either the name of a citable classification system/controlled vocabulary such as the Bailey classification of ecoregions or a domain of physical descriptors such as climate or disturbances.

Example(s):

climate
soils
hydrology
"bailey"
biome
disturbance
geology

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-protocol

Back to EML Contents

The eml-protocol module - Research protocol specific information

The EML Protocol Module is used to define abstract, prescriptive procedures for generating or processing data. Conceptually, a protocol is a standardized method.

Eml-protocol resembles eml-methods; however, eml-methods is descriptive (often written in the declarative mood: "I took five subsamples...") whereas eml-protocol is prescriptive (often written in the imperative mood: "Take five subsamples..."). A protocol may have versions, whereas methods (as used in eml-methods) should not.

Module details

Recommended Usage:	Use eml-protocol to describe prescriptive procedures that can be associated with other descriptive or prescriptive procedures.
Stand-alone:	yes
Imports:	eml-resource, eml-methods, eml-documentation, eml-access
Imported By:	
View an image of the schema:	**eml-protocol image**

Element Definitions:

proceduralStep	**This element has no default value.**
Content of this field:	Description of this field:
Type: md:ProcedureStepType	

protocol	**This element has no default value.**
Content of this field:	Description of this field:
Type: ProtocolType	The protocol field provides a container for other related fields, such as proceduralStep and ResourceGroup.

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

ProtocolType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
res:ResourceGroup		
proceduralStep	optional	unbounded
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

The ProtocolType container defines a number of reusable fields that can be referenced from multiple EML modules. It represents well-defined, prescriptive procedures that can be used to document other prescriptive procedures or descriptive procedures such as methods.

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-resource

Back to EML Contents

The eml-resource module - Base information for all resources

The eml-resource module contains general information that describes dataset resources, literature resources, protocol resources, and software resources. Each of the above four types of resources share a common set of information, but also have information that is unique to that particular resource type. Each resource type uses the eml-resource module to document the information common to all resources, but then extend eml-resource with modules that are specific to that particular resource type. For instance, all resources have creators, titles, and perhaps keywords, but only the dataset resource would have a "data table" within it. Likewise, a literature resource may have an "ISBN" number associated with it, whereas the other resource types would not.

The eml-resource module is exclusively used by other modules, and is therefore not a stand-alone module.

Module details

Recommended Usage:	all datasets
Stand-alone:	no
Imports:	eml-documentation, eml-party, eml-coverage, eml-text
Imported By:	
View an image of the schema:	**eml-resource image**

Element Definitions:

alternateIdentifier	This element has no default value.
Content of this field: **Attributes:** **Use:** **Default Value:** **system** optional	Description of this field: An additional, secondary identifier for this entity. The primary identifier belongs in the "id" attribute, but additional identifiers that are used to label this entity, possibly from different data management systems, can be listed here. **Example(s):** VCR3465

shortName	This element has no default value.
Content of this field: **Type:** **NonEmptyStringType**	Description of this field: The 'shortName' field provides a concise name that describes the resource that is being documented. It is the appropriate place to store a filename associated with other storage systems. **Example(s):** vernal-data-1999

title	This element has no default value.
Content of this field: **Type:** **NonEmptyStringType**	Description of this field: The 'title' field provides a description of the resource that is being documented that is long enough to differentiate it from other similar resources. Multiple titles may be provided, particularly when trying to express the title in more than one language (use the "xml:lang" attribute to indicate the language if not English/en). **Example(s):** Vernal pool amphibian density data, Isla Vista, 1990-1996.

creator	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The 'creator' element provides the full name of the person, organization, or position who created the resource. The list of creators for a resource represent the people and organizations who should be cited for the resource. **Example(s):** For a book, the creators are its authors.

metadataProvider	This element has no default value.
Content of this field: **Type:** rp:ResponsibleParty	Description of this field: The 'metadataProvider' element provides the full name of the person, organization, or position who created documentation for the resource. **Example(s):** The scientist who collected the data, sometimes a data technician, or other individual.

associatedParty	This element has no default value.
Content of this field: **Derived from:** rp:ResponsibleParty (by xs:extension) **Elements:** **Use:** **How many:** A sequence of (role required)	Description of this field: The 'associatedParty' element provides the full name of other people, organizations, or positions who should be associated with the resource. These parties might play various roles in the creation or maintenance of the resource, and these roles should be indicated in the "role" element. **Example(s):** The technician who collected the data.

role	This element has no default value.
Content of this field: **Type:** rp:RoleType	Description of this field: Use this field to describe the role the party played with respect to the resource. Some potential roles include technician, reviewer, principal investigator, and many others. **Example(s):** principalInvestigator

pubDate	This element has no default value.
Content of this field: **Type:** yearDate	Description of this field: The 'pubDate' field represents the date that the resource was published. The format should be represented as: CCYY, which represents a 4 digit year, or as CCYY-MM-DD, which denotes the full year, month, and day. Note that month and day are optional components. Formats must conform to ISO 8601. **Example(s):** 1999-10-26

language	This element has no default value.
Content of this field:	Description of this field: The language in which the resource is written. This can be a

Type: NonEmptyStringType

well-known language name, or one of the ISO language codes to be more precise.
Example(s):
English

series	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

This field describes the series of resources that include the resource being described. For example, a volume of a journal may be part of a series of the journal for a particular year.
Example(s):
Volume 20

abstract	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

A brief overview of the resource that is being documented. The abstract should include basic information that summarizes the resource.

keywordSet	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
keyword	required	unbounded
keywordThesaurus	optional	
)		

Description of this field:

The 'keywordSet' element is a container for the 'keyword' and 'keywordThesaurus' fields. Each keywordSet field can contain one or more keywords and a name of a thesaurus for the set of keywords. Each keyword field should contain one and only one keyword (i.e., keywords should not be separated by commas or other delimiters).
Example(s):
Please see the examples for the subfields contained within this field.

keyword	This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
keywordType	optional	

Description of this field:

This field names a keyword or key phrase that concisely describes the resource or is related to the resource. Each keyword field should contain one and only one keyword (i.e., keywords should not be separated by commas or other delimiters).
Example(s):
biodiversity

keywordThesaurus	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

This field provides the name of the official keyword thesaurus from which keyword was derived. The keyword thesauri are usually discipline specific.
Example(s):
IRIS keyword thesaurus

additionalInfo	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

This field provides any information that is not characterized by the other resource metadata fields.

Example(s):

Copyright 2001, Robert Warner

intellectualRights	This element has no default value.

Content of this field:

Type: txt:TextType

Description of this field:

Typically, an intellectual Rights element will contain a rights management statement for the resource, or reference a service providing such information. Rights information encompasses Intellectual Property Rights (IPR), Copyright, and various Property Rights. In the case of a data set, rights might include requirements for use, requirements for attribution, or other requirements the owner would like to impose.

Example(s):

Copyright 2001 Regents of the University of California Santa Barbara. Free for use by all individuals provided that the owners are acknowledged in any use or publication.

distribution	This element has no default value.

Content of this field:

Type: DistributionType

Description of this field:

This element provides information on how the resource is distributed. When used at the resource level, this element can provide only general information, but elements for describing connections to online systems are provided. See the Type for specific recommendations and examples.

coverage	This element has no default value.

Content of this field:

Type: cov:Coverage

Description of this field:

This element describes the extent of the coverage of the resource in terms of its spatial extent, temporal extent, and taxonomic extent. For data sets, this is useful to specify the entire extent to which all of the data might apply.

Example(s):

See the coverage module for examples.

references	This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
system	optional	document

Description of this field:

The id of another element in this EML document to be used to here in this context. This is used instead of duplicating information when an identical piece of information needs to be used multiple times in an EML document. For example, if the same person is the creator, metadataProvider, and contact for a dataset, their name and address can be provided once as part of the "creator" element, and then their "id" can be used in the "references" element of metadataProvider and contact. This reduces the likelihood of error by reducing redundancy, and allows one to specify that

two pieces of information are identical. To be a valid EML document, the content of every "references" element MUST be defined in the document as the value of an "id" attribute on some element within the document. Other critical rules about the use of IDs and references in EML are provided in the text of the EML specification.

Example(s):

knb.45.3

online	This element has no default value.

Content of this field:

Type: OnlineType

Description of this field:

This element contains information for accessing the resource online, represented either as a URL a connection, or a connectionDefinition which may be referenced in other parts of the EML document. See the Type definition for more information.

offline	This element has no default value.

Content of this field:

Type: OfflineType

Description of this field:

This element is for data which are distributed offline, generally by request. See the Type definition for more information.

inline	This element has no default value.

Content of this field:

Type: InlineType

Description of this field:

The data are distributed inline, with the metadata. See the Type definition for more information.

schemeName	This element has no default value.

Content of this field:

Attributes:	Use:	Default Value:
system	optional	

Description of this field:

The name of the scheme used to identify this connection. The scheme name is qualified by its system attribute. The scheme name implies a particular protocol for accessing information from the connection. Applications must have a knowledge of the scheme or be able to deduce the protocol from the scheme description in order to effectively access data over the connection. Many schemes will be unknown to client applications. At some later point in time a registry for connection schemes may be established in order to promote application interoperability, and we may expand this portion of EML to adopt a more comprehensive standard such as WSDL, but for now this simpler description is provided.

Example(s):

metacat

description	This element has no default value.

Content of this field:

Description of this field:

The description of the scheme used to identify this connection. The scheme name implies a particular protocol for accessing information from the connection. Applications

must have a knowledge of the scheme or be able to deduce the protocol from the scheme description in order to effectively access data over the connection.

Example(s):

The metacat application protocol. Applications must first log into metacat by sending an HTTP POST request in http-url-encoded format with the parameters action, username, and password. Action must be set to "login". If authentication is successful, the metacat server will respond with a session cookie. All future requests should include the session cookie in the HTTP header. To retrieve an object, the client then would send an HTTP POST in http-url-encoded format, with an action parameter set to "get" and the docid parameter set to the identifier for the desired object. The response will either be an XML document or a multipart-form-encoded response containing data.

Type: txt:TextType

parameterDefinition	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
name	required	
definition	required	
defaultValue	optional	
)		

Description of this field:

The definition of a parameter that is needed to properly use this connection scheme. Each parameter has a name and a definition that are used by applications to assess the type of information needed for the request. Parameters may also set default values that are used if a connection does not provide a value for a parameter.

name	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

The name of a parameter that is needed to properly use this connection scheme.

Example(s):

hostname

definition	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

The definition of a parameter that is needed to properly use this connection scheme. The definition is used by applications to assess the type of information needed for the request.

Example(s):

The fully qualified name of the internet host that is providing the metacat service, as would be returned by a Domain Name System (DNS) query.

defaultValue	This element has no default value.

Content of this field:

Description of this field:

The default value for a parameter that is needed to properly use this connection scheme. If a default value is set, then it should be used for connections that do not override the default with a connection-specific value. This allows a definition to be established that declares common

Type: NonEmptyStringType	information that might be shared by several connections as default values. Parameter values provided in the connection always override any default values provided in the connection definition. **Example(s):** metacat.nceas.ucsb.edu

mediumName	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: Name of the medium on which this resource is distributed. Can be various digital media such as tapes and disks, or printed media which can collectively be termed 'hardcopy'. **Example(s):** Tape, 3.5 inch Floppy Disk, hardcopy

mediumDensity	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: the density of the digital medium if this is relevant. Used mainly for floppy disks or tape. **Example(s):** High Density (HD), Double Density (DD)

mediumDensityUnits	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: if a density is given numerically, the units should be given here. **Example(s):** B/cm

mediumVolume	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: the total volume of the storage medium on which this resource is shipped. **Example(s):** 650 MB

mediumFormat	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: the file system format of the medium on which the resource is shipped **Example(s):** NTFS, FAT32, EXT2, QIK80

mediumNote	This element has no default value.
Content of this field: **Type:** NonEmptyStringType	Description of this field: any additional pertinent information about the media

onlineDescription	This element has no default value.
Content of this field:	Description of this field:

Type: NonEmptyStringType	This element can hold a brief description of the content of the online element's online|offline|inline child. This description element could supply content for an html anchor tag.

url	This element has no default value.
Content of this field:	Description of this field:
Type: UrlType	A URL (Uniform Resource Locator) from which this additional information can be obtained, or from which the resource can be downloaded directly. In the resource module, the distribution URL is generally meant for informational purposes, and the "function" attribute should be set to "information". However, if the URL returns the data stream itself, then the "function" attribute should be set to "download". See the Type Definition for more information.

connection	This element has no default value.
Content of this field:	Description of this field:
Type: ConnectionType	A description of the information needed to make an application connection to a data service. The connection contains a connectionDefinition and optional parameters for overriding defaults. See the Type Definition for more information.

connectionDefinition	This element has no default value.
Content of this field:	Description of this field:
Type: ConnectionDefinitionType	The definition of a type of connection that will be used in another location in the EML document. The connectionDefinition element only provides the definition of the protocol and its parameters, but not the actual values to be used to make the connection (instead, see the connection element). This connectionDefinition may be used by multiple connections (e.g., to download different files from the same database), but each connection must provide or reference a valid connection definition. The definition has a "scheme" which identifies the protocol by name, with a detailed description and its required parameters. A connectionDefinition lists all of the parameters needed for the connection and possible default values for each.

connectionDefinition	This element has no default value.
Content of this field:	Description of this field:
Type: ConnectionDefinitionType	In a ConnectionType, the connectionDefinition element provides the definition of the protocol and its parameters. The definition has a "scheme" which identifies the protocol by name, with a detailed description and its required parameters. A connectionDefinition lists all of the parameters needed for the connection and possible default values for each.

parameter	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
name	required	
value	required	
)		

Description of this field:

A parameter to be used to make this connection. This value overrides any default value that may have been provided in the connection definition.

name	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

The name of the parameter to be used to make this connection.
Example(s):
hostname

value	This element has no default value.

Content of this field:

Type: NonEmptyStringType

Description of this field:

The value of the parameter to be used to make this connection. This value overrides any default value that may have been provided in the connection definition.
Example(s):
nceas.ucsb.edu

Attribute Definitions:

system

Type: SystemType

Use: optional

keywordType

Type: KeyTypeCode

Use: optional

This field classifies the keyword that has been provided from a list of pre-determined categories. The possible types are listed in the example.
Example(s):
"place","stratum","temporal","theme", or "taxonomic"

system

Type: SystemType

Use: optional

Default value: document

id

Type: IDType

Use: optional

system

Type: SystemType

Use: optional

scope

Type: ScopeType

Use: optional

Default value: document

system

Type: SystemType

Use: optional

The computing system within which this scheme name has relevance. This attribute qualifies the scheme name in order to decrease the likelihood of scheme name collisions when more that one EML user defines a scheme name with the same name but different semantics.

Example(s):

http://knb.ecoinformatics.org/knb/

id

Type: IDType

Use: optional

system

Type: SystemType

Use: optional

scope

Type: ScopeType

Use: optional

Default value: document

function

Type: FunctionType

Use: optional

Default value: download

id

Type: IDType

Use: optional

system

Type: SystemType

Use: optional

scope

Type: ScopeType

Use: optional

Default value: document

Complex Type Definitions:

DistributionType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A choice of (		
online	required	
OR		
offline	required	
OR		
inline	required	
)		
ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

Distribution information for accessing the resource by one of three methods: online, offline or inline. Generally, the Type at the resource level is intended to be informational, although elements are included to describe a complex connection protocol.

For more information, see the sub-elements. Also compare to **phys:PhysicalDistributionType** .

ConnectionDefinitionType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
schemeName	required	
description	required	
parameterDefinition	required	unbounded
)		
OR		
ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

Definition of the connection protocol. The definition has a "scheme" which identifies the protocol by name, with a detailed description and its required parameters. A connectionDefinition lists all of the parameters needed for the connection and possible default values for each. A definition provided in this element may be used in other parts of the EML document.

InlineType

Content of this field:

Derived from: xs:anyType (by xs:restriction)

Derived from: xs:anyType (by xs:restriction)

Elements:	Use:	How many:
A sequence of (		

Description of this field:

Object data distributed inline in the metadata. Users have the option of including the data right inline in the metadata by providing it inside of the "inline" element. For many text formats, the data can be simply included directly in the element. However, certain character sequences are invalid in an XML document (e.g., <), so care will need to be taken to either 1) wrap the data in a CDATA section if needed, or 2) encode the data using a text encoding algorithm such as

)

base64, and then include that in a CDATA section. The latter will be necessary for binary formats.

OfflineType

Content of this field:

Elements:	Use:	How many:
A sequence of (		
mediumName	required	
mediumDensity	optional	
mediumDensityUnits	optional	
mediumVolume	optional	
mediumFormat	optional	unbounded
mediumNote	optional	
)		

Description of this field:

the medium on which this resource is distributed digitally, such as 3.5" floppy disk, or various tape media types, or 'hardcopy'

Example(s):

CD-ROM, 3.5 in. floppy disk, Zip disk

OnlineType

Content of this field:

Elements:	Use:	How many:
A sequence of (		
onlineDescription	optional	
A choice of (		
url	required	
OR		
connection	required	
OR		
connectionDefinition	required	
)		
)		

Description of this field:

Distribution information for accessing the resource online, represented either as a URL or as the series of named parameters needed to connect. The URL field can contain a simple web address or an entire query string. The connection element allows the components of a complex protocol to be described individually. The connectionDefinition element can also be appear outside of the connection elements, so that it can be defined once and referenced by several connections.

Also see the **phys:PhysicalOnlineType**, which may be more appropriate for describing the online connections to a specific entity described by this metadata document.

UrlType

Content of this field:

Derived from: xs:anyURI (by xs:extension)

Elements:	Use:	How many:
function	optional	download

Description of this field:

A URL (Uniform Resource Locator) from which this resource can be downloaded or additional information can be obtained. If the URL provides further information about downloading the object but does not directly return the data stream, then the "function" attribute should be set to "information". If accessing the URL would directly return the data stream, then the "function" attribute should be set to "download". If the "function" attribute is omitted, then "download" is implied for the URL function. In more complex cases where a non-standard connection must be established that complies with application specific procedures beyond what can be described in the simple URL, then the "connection" element should be used instead of the URL element.

Example(s):

http://data.org/getdata?id=98332

ConnectionType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
connectionDefinition	required	
parameter	optional	unbounded
)		
OR		
ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

A description of the information needed to make an application connection to a data service. The connection starts with a connectionDefinition which lists all of the parameters needed for the connection and possible default values for each. It may also include a list of parameter values (one for each), that override the defaults for this particular connection. One parameter element should exist for every parameterDefinition that is present in the connectionDefinition, although parameters that were defined with a defaultValue in their parameterDefinition can be omitted. All information about how to use the parameters to establish a session and extract data is present in the connectionDefinition, possibly implicitly by naming a connection schemeName that is well-known. See descriptions of the child element types for further information.

Simple Type Definitions:

KeyTypeCode

Derived from: xs:string (by xs:restriction)

Allowed values:

- place
- stratum
- temporal
- theme
- taxonomic

This field provides a restricted list of categories that a keyword may fall under.

Example(s):

place

yearDate

This type is the union of the built-in types for year and date.

Example(s):

1999, or 2001-03-15

IDType

A unique identifier for this additional metadata that can be used to reference it elsewhere. This is a formal field in that it is an error to provide a value for the id attribute that is not unique within the document's set of id attributes. This is designed to allow other portions of the metadata to reference this section formally.

Example(s):

knb.343.22

SystemType

The data management system within which an identifier is in scope and therefore unique. This is typically a URL (Uniform Resource Locator) that indicates a data management system. All identifiers that share a system must be unique. In other words, if the same identifier is used in two locations with identical systems, then by definition the objects at which they point are in fact the same object.
Example(s):
http://metacat.somewhere.org/svc/mc/

ScopeType

Derived from: xs:string (by xs:restriction)

Allowed values:

- system
- document

The scope of the identifier. Scope is generally set to either "system", meaning that it is scoped according to the "system" attribute, or "document" if it is only to be in scope within this single document instance. In this particular use of scope, it is FIXED to be "system" because the packageId is required and always has the scope of the required "system".
Example(s):
system

FunctionType

Derived from: xs:string (by xs:restriction)

Allowed values:

- download
- information

NonEmptyStringType

Derived from: xs:string (by xs:restriction)

Allowed values:

This type specifies a content pattern for all elements that are required by EML to ensure that there is actual content (i.e., not just whitespace). The pattern described can be interpreted as "at least one non-whitespace character, followed by any number of whitespace plus not-whitespace characters. " Leading and/or trailing whitespace is allowed, and whitespace may include carriage returns and newlines.

Group Definitions:

ResourceGroup

Content of this field:

Elements:	Use:	How many:
A sequence of (		
alternateIdentifier	optional	unbounded
shortName	optional	
title	required	unbounded
creator	required	unbounded
metadataProvider	optional	unbounded

Description of this field:

The 'ResourceBase' complexType contains information that is inherited by each resource type that is being documented. The sub-elements with the resource base are common to all resources.
Example(s):
Please see the individual sub-fields for specific examples.

associatedParty	optional	unbounded
pubDate	optional	
language	optional	
series	optional	
abstract	optional	
keywordSet	optional	unbounded
additionalInfo	optional	unbounded
intellectualRights	optional	
distribution	optional	unbounded
coverage	optional	

)

ReferencesGroup	
Content of this field:	Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		
references	required	
)		

A group containing the "references" element that is used throughout EML.

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-software

Back to EML Contents

The eml-software module - Software specific information

The eml-software module contains general information that describes software resources. This module is intended to fully document software that is needed in order to view a resource (such as a dataset) or to process a dataset. The software module is also imported into the eml-methods module in order to document what software was used to process or perform quality control procedures on a dataset.

The eml-software module, like other modules, may be "referenced" via the <references> tag. This allows a software resource to be described once, and then used as a reference in other locations within the EML document via its ID.

Module details

Recommended Usage:	All datasets where software was used in the analysis or creation of the dataset.
Stand-alone:	yes
Imports:	eml-documentation, eml-resource, eml-access, eml-project, eml-physical
Imported By:	
View an image of the schema:	**eml-software image**

Element Definitions:

software	**This element has no default value.**
Content of this field: **Type: SoftwareType**	Description of this field: The software element contains general information about a software resource that is being documented. This field is intended to give information for software tools that are needed to interpret a dataset, software that was written to process a resource, or software as a resource in itself. It is based on eml-resource and Open Software Description (OSD) a W3C submission. There can be multiple implementations within a software package because a physical software package can run on multiple hardware and/or operating systems. See implementation element documentation for a more thorough explanation.

implementation	**This element has no default value.**
Content of this field: **Elements:** **Use:** **How many:** A sequence of (**distribution** required unbounded **size** optional **language** optional unbounded **operatingSystem** optional unbounded **machineProcessor** optional unbounded **virtualMachine** optional **diskUsage** optional **runtimeMemoryUsage** optional **programmingLanguage** optional unbounded **checksum** optional **dependency** optional unbounded	Description of this field: Implementation describes the hardware, operating system resources a package runs on. Note, a package can have multiple implementations. So for example, a package may be written in java and the package may run on numerous hardware and/or operating systems like Pentium/Linux, Pentium/NT and so on. Hardware and Software descriptions that have different requirements can be placed here. **Example(s):** Please see the examples for each sub-element of the implementation type.

)

distribution	This element has no default value.
Content of this field: **Type:** phys:PhysicalDistributionType	Description of this field: This field provides information on how the resource is distributed online and offline. Connections to online systems can be described as URLs and as a list of relevant connection parameters.

size	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The physical size of an implementation on disk. **Example(s):** 100 Megabytes

language	This element has no default value.
Content of this field: **Elements:** **Use:** **How many:** A sequence of (LanguageValue required LanguageCodeStandard optional)	Description of this field: The International Language of the software implementation.

LanguageValue	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The actual value for the language or a code for the language. **Example(s):** english eng

LanguageCodeStandard	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The International Language Code being used in the field languageValue. See http://www.loc.gov/standards/iso639-2/ **Example(s):** ISO639-2

operatingSystem	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The operating system(s) an implementation runs on. **Example(s):** Linux Windows 95 Windows NT4 Windows XP Sun Solaris 2.8 Mac OS X

machineProcessor	This element has no default value.

Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The Machine Processor required for executing the implementation. **Example(s):** Pentium II Intel 486 SUN Sparc Motorola

virtualMachine	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The virtual machine that the implementation requires. **Example(s):** Java Virtual Machine 1.2

diskUsage	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The minimum amount of Disk Space required to install this implementation. **Example(s):** 220 Megabytes 15 MB 100 kB

runtimeMemoryUsage	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The minimum amount of memory required to run an implementation. **Example(s):** 32 Megabytes 128 MB

programmingLanguage	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The computer programming language the software package was programmed in. **Example(s):** C++ Java C C# Perl Cobol Fortran Lisp Visual Basic VBA Bourne Shell Script

checksum	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The generated checksum value of a software package that is a self-contained module.

Example(s):

$sum software.jar 27021 22660

licenseURL	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

URL where the license can be found

license	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Text of the license

version	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

String value corresponding to the major, minor, custom, and build version.

project	This element has no default value.

Content of this field:

Type: proj:ResearchProjectType

Description of this field:

This field is a description of the project with which this software product is related. Please see the eml-project module for more information.

dependency	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
action	required	
software	required	
)		

Description of this field:

The dependency element is recursive. It is a sub-element of the software Element but it also has as a sub-element its parent element Software Package. Dependency has been made optional because to make it mandatory does not allow the recursion to end. Dependency has also been made a sub-element of implementation because there can be both implementation and package level dependencies within a package.

action	This element has no default value.

Content of this field:

Type: Action

Description of this field:

This element and its enumerations of assert and install can be used as commands by a software application to carry out these actions on software package dependencies. This is a change from how we have used all previous elements within eml. Up until now all other elements have been simply metadata designed to describe data, literature citations, etc... with the Action element we can use this module as a command to carry out the action.

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

SoftwareType

Content of this field:

Description of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
res:ResourceGroup		
implementation	required	unbounded
dependency	optional	unbounded
A choice of (		
licenseURL	required	
OR		
license	required	
)		
version	required	
project	optional	
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Simple Type Definitions:

Action

Derived from: xs:string (by xs:restriction)

Allowed values:

- install
- assert

This element and its enumerations of assert and install can be used as commands by a software application to carry out these actions on software package dependencies. This is a change from how we have used all previous elements within eml. Up until now all other elements have been simply metadata designed to describe data, literature citations, etc... with the Action element we can use this module as a command to carry out the action.

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-spatialRaster

Back to EML Contents

The eml-spatialRaster module - Logical information about regularly gridded geospatial image data

The eml-spatialRaster module allows for the description of entities composed of rectangular grids of data values that are usually georeferenced to a portion of the earth's surface. Specific attributes of a spatial raster can be documented here including the spatial organization of the raster cells, the cell data values, and if derived via imaging sensors, characteristics about the image and its individual bands.

Module details

Recommended Usage:	all spatial datasets that use spatial gridded data
Stand-alone:	yes
Imports:	eml-documentation, eml-spatialReference, eml-coverage, eml-entity, eml-resource, eml-attribute, eml-constraint
Imported By:	
View an image of the schema:	**eml-spatialRaster image**

Element Definitions:

spatialRaster	**This element has no default value.**
Content of this field: **Type:** **SpatialRasterType**	Description of this field: Description of a GIS layer composed of raster pixels comprising a regular-pattern grid. Generally, a raster object may be thought of as a pattern of closely spaced rows of dots that collectively form an image. Raster spatial objects are used to locate zero-, two-, or three-dimensional locations in the dataset **Example(s):** An interpolated grid of irregularly spaced data is an example of this element.

attributeList	**This element has no default value.**
Content of this field: **Type:** **att:AttributeListType**	Description of this field: The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	**This element has no default value.**
Content of this field: **Type:** **con:ConstraintType**	Description of this field: Description of any relational constraints on this entity. For more information see the eml-constraint module.

	This element has no default

spatialReference	value.

Content of this field:

Type: spref:SpatialReferenceType

Description of this field:

Spatial Referencing systems define the coordinates used to describe horizontal and vertical locations. These are typically either geographic, projected planar, or arbitrary planar.

georeferenceInfo	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A choice of (		
cornerPoint	required	4
OR		
controlPoint	required	unbounded
OR		
bilinearFit	required	
)		

Description of this field:

Information on how to position the grid within the coordinate system defined in the spatial reference module.

cornerPoint	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
xCoordinate	required	
yCoordinate	required	
pointInPixel	required	
corner	required	
)		

Description of this field:

Location of a corner on the coordinate system defined in the spatial reference module. Use this element when the rows and columns of the grid are aligned with the coordinate system. One or more corner points are needed to locate a rectified grid.

xCoordinate	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Location of the georeferencing point expressed in units of the coordinate system defined in the spatial Reference module.
Example(s):
455000

yCoordinate	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Y Location of the georeferencing point expressed in units of the coordinate system defined in the spatial Reference module.
Example(s):

3455000

pointInPixel	This element has no default value.

Content of this field:

Description of this field:

Location within the pixel of the georeferencing point.
Example(s):
upperLeft

Derived from: xs:string (by xs:restriction)

Allowed values:

- upperLeft
- upperRight
- lowerRight
- lowerLeft
- center

corner	This element has no default value.

Content of this field:

Type: rasterOriginType

Description of this field:

Identification of the corner in the grid corresponding to the coordinates provided.
Example(s):
Upper Left

controlPoint	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
column	required	
row	required	
xCoordinate	required	
yCoordinate	required	
pointInPixel	required	
)		

Description of this field:

If the grid is rectified to the coordinate system, then a single point may be used to position the grid. Otherwise, a series of points is necessary to fit the grid to the coordinate system.

column	This element has no default value.

Content of this field:

Type: xs:int

Description of this field:

Column location of the georeferencing point indicated as the nth column counting the cell indicated in rasterOrigin as column 1.
Example(s):
1

row	This element has no default value.

Content of this field:

Type: xs:int

Description of this field:

Row location of the georeferencing point indicated as the nth row counting the cell indicated in rasterOrigin as row 1.

Example(s):

1

xCoordinate	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Location of the georeferencing point expressed in units of the coordinate system defined in the spatial Reference module.

Example(s):

455000

yCoordinate	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Y Location of the georeferencing point expressed in units of the coordinate system defined in the spatial Reference module.

Example(s):

3455000

pointInPixel	This element has no default value.

Content of this field:

Description of this field:

Location within the pixel of the georeferencing point.

Example(s):

upperLeft

Derived from: xs:string (by xs:restriction)

Allowed values:

- upperLeft
- upperRight
- lowerRight
- lowerLeft
- center

bilinearFit	This element has no default value.

Content of this field:

Description of this field:

Intercept and slope terms to describe the orientation and position of the grid to the

Elements:	Use:	How many:
A sequence of (		

xIntercept	required	coordinate system based on corner point identified in the rasterOrigin element. defined in the spatial Reference module.
xSlope	required	
yIntercept	required	
ySlope	required	

)

xIntercept	**This element has no default value.**

Content of this field:

Type: xs:float

Description of this field:

X Intercept of the rasterOrigin point within the coordinate system.

Example(s):

3455000

xSlope	**This element has no default value.**

Content of this field:

Type: xs:float

Description of this field:

Slope describing transformation of grid cell distances into x coordinates.

Example(s):

5.0123

yIntercept	**This element has no default value.**

Content of this field:

Type: xs:float

Description of this field:

Location of the rasterOrigin point on the y axis of the the coordinate system.

Example(s):

455000

ySlope	**This element has no default value.**

Content of this field:

Type: xs:float

Description of this field:

Slope describing transformation of grid cell distances into y axis coordinates.

Example(s):

5.0123

horizontalAccuracy	**This element has no default value.**

Content of this field:

Type: DataQuality

Description of this field:

Horizontal accuracy may be reported either as a quantitative estimate expressed in the units of the coordinate system or as a text assessment.

verticalAccuracy	**This element has no default value.**

Content of this field:

Type: DataQuality

Description of this field:

Vertical accuracy may be reported either as a quantitative estimate expressed in the units of the height or depth measurement system or as a text assessment.

cellSizeXDirection	This element has no default value.

Content of this field:

Description of this field:

Cell sizes should be expressed in the units declared in the spatialReference module.
Example(s):
28.5

cellSizeYDirection	This element has no default value.

Content of this field:

Description of this field:

Cell sizes should be expressed in the units declared in the spatialReference module.
Example(s):
28.5

numberOfBands	This element has no default value.

Content of this field:

Description of this field:

Image data may have more than one sensor band represented for each pixel.
Example(s):
7

rasterOrigin	This element has no default value.

Content of this field:

Type: rasterOriginType

Description of this field:

Identification the corner of the grid where the first values for both the x and y axes begin in the file.
Example(s):
Upper Left

rows	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Maximum number of raster objects along the ordinate (y) axis
Example(s):
455

columns	This element has no default

	value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Maximum number of raster objects along the abscissa (x) axis **Example(s):** 455

verticals	This element has no default value.
Content of this field:	Description of this field: Maximum number of raster objects along the vertical (z) axis.

cellGeometry	This element has no default value.
Content of this field: **Type:** CellGeometryType	Description of this field: Indication of whether the cell value is representative of a single point(matrix) within the cell or the entire cell (pixel) **Example(s):** pixel

toneGradation	This element has no default value.
Content of this field: **Type:** xs:integer	Description of this field: Number of colors present in the image. **Example(s):** 255

scaleFactor	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The scale factor is used for raster-rescaling operations, where the following operation is applied to each pixel in the data in the source: rescaled=(source*scaleFactor)+offset. **Example(s):** 2

offset	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The offset is used for raster-rescaling operations, where the following operation is applied to each pixel in the data in the source: rescaled=(source*scaleFactor)+offset.

Example(s):
20

imageDescription	This element has no default value.

Content of this field:

Description of this field:

Provides information about the image's suitability for use, sensor properties, and individual band descriptions.

Elements:	Use:	How many:
A sequence of (		
illuminationElevationAngle	optional	
illuminationAzimuthAngle	optional	
imageOrientationAngle	optional	
imagingCondition	optional	
imageQualityCode	optional	
cloudCoverPercentage	optional	
preProcessingTypeCode	optional	
compressionGenerationQuality	optional	
triangulationIndicator	optional	
radiometricDataAvailability	optional	
cameraCalibrationInformationAvailability	optional	
filmDistortionInformationAvailability	optional	
lensDistortionInformationAvailability	optional	
bandDescription	optional	unbounded
)		

illuminationElevationAngle	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Illumination elevation measured in degrees clockwise from the target plane at intersection of the optical line of sight with the earth's surface.

Example(s):
45.5

illuminationAzimuthAngle	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Illumination azimuth measured in degrees clockwise from true north at the time the image is taken.

Example(s):
45.5

imageOrientationAngle	This element has no default value.

Content of this field:

Description of this field:

Type: xs:float

Angle from the first row of the image to true north in degrees, clockwise.
Example(s):
45.5

imagingCondition	This element has no default value.

Content of this field:

Type: ImagingConditionCode

Description of this field:

Code which indicates conditions which affect the quality of the image.
Example(s):
cloud

imageQualityCode	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Specifies the image quality.
Example(s):
Excellent

cloudCoverPercentage	This element has no default value.

Content of this field:

Type: xs:float

Description of this field:

Area of the dataset obscured by clouds, expressed as a percentage of the spatial extent.
Example(s):
12

preProcessingTypeCode	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Image distributor's code that identifies the level of radiometric and geometric processing applied against the image.
Example(s):
LEVELI1A, SPOTVIEWORTHO

compressionGenerationQuality	This element has no default value.

Content of this field:

Type: xs:integer

Description of this field:

Counts the number of lossy compression cycles performed on the image.
Example(s):
2

triangulationIndicator	This element has no default value.

Content of this field:

Type: xs:boolean

Description of this field:

Code which indicates whether or not triangulation has been performed upon the image.

Example(s):

false

radiometricDataAvailability	This element has no default value.

Content of this field:

Type: xs:boolean

Description of this field:

Code which indicates whether or not Standard Radiometric Product data is available.

Example(s):

false

cameraCalibrationInformationAvailability	This element has no default value.

Content of this field:

Type: xs:boolean

Description of this field:

Code which indicates whether or not constants are available which allow for camera calibration corrections.

Example(s):

false

filmDistortionInformationAvailability	This element has no default value.

Content of this field:

Type: xs:boolean

Description of this field:

Code which indicates whether or not Calibration Reseau information is available

Example(s):

true

lensDistortionInformationAvailability	This element has no default value.

Content of this field:

Type: xs:boolean

Description of this field:

Code which indicates whether or not lens aberration correction information is available

Example(s):

true

bandDescription	This element has no default value.

Content of this field:

Type: BandType

Description of this field:

Description of the wavelength domain that the sensor operates in

	This element has no default

sequenceIdentifier	value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: Number that uniquely identifies instances of bands of wavelengths on which a sensor operates. **Example(s):** 3

highWavelength	This element has no default value.
Content of this field: **Type: xs:float**	Description of this field: Highest wavelength that the sensor is capable of collecting within a designated band. **Example(s):** 2.456

lowWaveLength	This element has no default value.
Content of this field: **Type: xs:float**	Description of this field: Lowest wavelength that the sensor is capable of collecting within a designated band. **Example(s):** 0.1234

waveLengthUnits	This element has no default value.
Content of this field: **Type:** spref:lengthUnits	Description of this field: Units of measure for the wavelength at which the sensor collected the data. **Example(s):** microns

peakResponse	This element has no default value.
Content of this field:	Description of this field:

accuracyReport	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: A text statement of the data quality, included the means by which it was determined.

quantitativeAccuracyReport	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
quantitativeAccuracyValue	required	
quantitativeAccuracyMethod	required	
)		

Description of this field:

A quantitative assessment of the data quality expressed as a value and the method of its determination.

quantitativeAccuracyValue

This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

The value resulting from the accuracy test. Typically, this will be expressed in units corresponding to those declared for the parameter being assessed.

Example(s):

4.5

quantitativeAccuracyMethod

This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Identification and explanation of the method used to calculate the quantitative accuracy assessment.

Example(s):

Error expressed as root mean square of 5 control points.

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

SpatialRasterType

Content of this field:

Elements:	Use:	How many:

Description of this field:

A choice of (

A sequence of (

ent:EntityGroup		
attributeList	required	
constraint	optional	unbounded
spatialReference	required	
georeferenceInfo	optional	
horizontalAccuracy	required	
verticalAccuracy	required	
cellSizeXDirection	required	
cellSizeYDirection	required	
numberOfBands	required	
rasterOrigin	required	
rows	required	
columns	required	
verticals	required	
cellGeometry	required	
toneGradation	optional	
scaleFactor	optional	
offset	optional	
imageDescription	optional	

)

OR

res:ReferencesGroup

)

Attributes:	**Use:**	**Default Value:**
id	optional	
system	optional	
scope	optional	document

BandType

Content of this field:

Description of this field:

Elements:	**Use:**	**How many:**

A sequence of (

sequenceIdentifier	optional	
highWavelength	optional	
lowWaveLength	optional	
waveLengthUnits	optional	
peakResponse	optional	

)

DataQuality

Content of this field:

Description of this field:

Elements:	**Use:**	**How many:**

A sequence of (

accuracyReport	required	
quantitativeAccuracyReport	optional	unbounded

)

Simple Type Definitions:

CellValueType

Derived from: xs:string (by xs:restriction)

Allowed values:

- Values
- Coded
- RGB
- Codes
- HIS
- HLS
- tekHVC

ImagingConditionCode

Derived from: xs:string (by xs:restriction)

Allowed values:

- blurredimage
- cloud
- degradingObliquity
- fog
- heavySmokeorDust
- night
- rain
- semiDarkness
- shadow
- snow
- terrainMasking

rasterOriginType

Derived from: xs:string (by xs:restriction)

Allowed values:

- Upper Left
- Lower Left
- Upper Right
- Lower Right

CellGeometryType

Derived from: xs:string (by xs:restriction)

Allowed values:

- pixel
- matrix

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-spatialReference

Back to EML Contents

Schema for validating spatial referencing descriptions.

This module defines both projected and unprojected coordinate systems for referencing the spatial coordinates of a dataset to the earth. The schema is based on that used by Environmental Systems Research Inc (ESRI) for its .prj file format. EML provides a library of pre-defined coordinate systems that may be referred to by name in the horizCoordSysName element. A custom projection may be defined using this schema for any projection that does not appear in this dictionary.

Module details

Recommended Usage:	all spatial entities
Stand-alone:	yes
Imports:	eml-documentation, eml-resource
Imported By:	
View an image of the schema:	**eml-spatialReference image**

Element Definitions:

horizCoordSysName	**This element has no default value.**
Content of this field:	Description of this field: The name of a coordinate system for which a definition has been provided in the eml-spatialReferenceDictionary.xsl file. **Example(s):** NAD_1927_StatePlane_Arizona_Central_FIPS_0202

Derived from: xs:string (by xs:restriction)

Allowed values:

- GCS_Abidjan_1987
- GCS_Accra
- GCS_Adindan
- GCS_Afgooye
- GCS_Agadez
- GCS_Ain_el_Abd_1970
- GCS_Arc_1950
- GCS_Arc_1960
- GCS_Ayabelle
- GCS_Beduaram
- GCS_Bissau
- GCS_Camacupa
- GCS_Cape
- GCS_Carthage_Degree
- GCS_Carthage_Paris
- GCS_Carthage
- GCS_Conakry_1905
- GCS_Cote_d_Ivoire
- GCS_Dabola
- GCS_Douala
- GCS_Egypt_1907
- GCS_European_1950
- GCS_European_Libyan_Datum_1979
- GCS_Garoua
- GCS_Hartebeesthoek_1994
- GCS_Kuwait_Oil_Company
- GCS_KUDAMS
- GCS_Leigon

Many pages comprising list of allowable content for horizCoordSysName
removed.

- World_Miller_Cylindrical
- World_Mollweide
- World_Plate_Carree
- World_Polyconic
- World_Quartic_Authalic
- World_Robinson
- World_Sinusoidal
- Sphere_Aitoff
- Sphere_Behrmann
- Sphere_Bonne
- Sphere_Craster_Parabolic
- Sphere_Cylindrical_Equal_Area
- Sphere_Eckert_I
- Sphere_Eckert_II
- Sphere_Eckert_III
- Sphere_Eckert_IV
- Sphere_Eckert_V
- Sphere_Eckert_VI
- Sphere_Equidistant_Conic
- Sphere_Equidistant_Cylindrical
- Sphere_Flat_Polar_Quartic
- Sphere_Gall_Stereographic
- Sphere_Hammer_Aitoff
- Sphere_Loximuthal
- Sphere_Mercator
- Sphere_Miller_Cylindrical
- Sphere_Mollweide
- Sphere_Plate_Carree
- Sphere_Polyconic
- Sphere_Quartic_Authalic
- Sphere_Robinson
- Sphere_Sinusoidal
- Sphere_Times
- Sphere_Van_der_Grinten_I
- Sphere_Vertical_Perspective
- Sphere_Winkel_I
- Sphere_Winkel_II
- Sphere_Winkel_Tripel_NGS
- The_World_From_Space
- World_Times
- World_Van_der_Grinten_I
- World_Vertical_Perspective
- World_Winkel_I
- World_Winkel_II
- World_Winkel_Tripel_NGS

horizCoordSysDef	**This element has no default value.**
Content of this field: **Type:** **horizCoordSysType**	Description of this field: Terms and parameters necessary to define a geographic or projected coordinate system for horizonal distances.

vertCoordSys	**This element has no default value.**
Content of this field:	Description of this field:

Elements:	**Use:**	**How many:**
A sequence of (		

altitudeSysDef optional
depthSysDef optional
)

altitudeSysDef	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
altitudeDatumName	required	
altitudeResolution	required	unbounded
altitudeDistanceUnits	required	
altitudeEncodingMethod	required	
)		

Description of this field:

The term "altitude"' is used instead of the common term "elevation" to conform to the terminology in Federal Information Processing Standards 70-1 and 173.

altitudeDatumName	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Example(s):
WGS_datum

altitudeResolution	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Example(s):
1

altitudeDistanceUnits	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

Example(s):
1

altitudeEncodingMethod	This element has no default value.

Content of this field:

Type: res:NonEmptyStringType

Description of this field:

depthSysDef	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
depthDatumName	required	
depthResolution	required	unbounded
depthDistanceUnits	required	
depthEncodingMethod	required	
)		

Description of this field:

depthDatumName	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

depthResolution	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

depthDistanceUnits	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

depthEncodingMethod	This element has no default value.

Content of this field:

Description of this field:

Type: res:NonEmptyStringType

datum	This element has no default value.

Content of this field:

Description of this field:

Elements:	Use:	How many:
Attributes:	Use:	Default Value:
name		

Example(s):
WGS_1984

spheroid	This element has no default value.

Content of this field:

Description of this field:

An ellipse is used to define a idealized surface that ignores variation in elevation.

Elements:	Use:	How many:
Attributes:	Use:	Default Value:
name		
semiAxisMajor		
denomFlatRatio		

primeMeridian	This element has no default value.

Content of this field:

Description of this field:

Elements:	Use:	How many:
Attributes:	Use:	Default Value:
name		
longitude	required	

unit	This element has no default value.

Content of this field:

Description of this field:

Elements:	Use:	How many:
Attributes:	Use:	Default Value:
name	required	

Example(s):
degrees

geogCoordSys	This element has no default

	value.

Content of this field:

Description of this field:

Type: geogCoordSysType

projCoordSys	**This element has no default value.**

Content of this field:

Description of this field:

Elements:	Use:	How many:
A sequence of (		
geogCoordSys	required	
projection	required	
)		

geogCoordSys	**This element has no default value.**

Content of this field:

Description of this field:

Type: geogCoordSysType

projection	**This element has no default value.**

Content of this field:

Description of this field:

Elements:	Use:	How many:
A sequence of (		
parameter	required	unbounded
unit	required	
)		

Attributes:	Use:	Default Value:
name		

parameter	**This element has no default value.**

Content of this field:

Description of this field:

Most projections require one or more parameters to control the orientation and positon of the projected coordinate plane.

Elements:	Use:	How many:

Attributes:	Use:	Default Value:
name	required	
description	optional	
value	required	

unit	**This element has no default value.**

Content of this field:

Description of this field:

Elements:	Use:	How many:

Attributes:	Use:	Default Value:
name	required	

projectionList	**This element has no default value.**

Content of this field:

Description of this field:

Elements:	Use:	How many:
A sequence of (		
horizCoordSysDef	required	unbounded

)

horizCoordSysDef	This element has no default value.

Content of this field:

Type: horizCoordSysType

Description of this field:

Terms and parameters necessary to define a geographic or projected coordinate system for horizonal distances.

spatialReference	This element has no default value.

Content of this field:

Type: SpatialReferenceType

Description of this field:

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

name

Type: xs:string

name

Type: xs:string

Example(s):
Bessel 1866

semiAxisMajor

Type: xs:float

Example(s):
Meter

denomFlatRatio

Type: xs:float

name

Type: xs:string

Example(s):
Greenwich

longitude

Use: required

Derived from: xs:float (by xs:restriction)

Allowed values:

- **Minimum:** -180
- **Maximum:** 180

Example(s):
0.0

name

Type: angleUnits
Use: required

Example(s):
Degrees

name

Type: xs:string

name

Type: xs:string
Use: required

If the paramter corresponds to one that has been defined in the existing eml-SpatialReferenceDictionary, then that name should be used to enhance recogonizability.
Example(s):
False_Easting

description

Type: xs:string
Use: optional

Example(s):
An offset by which x origin has been adjusted, expressed in projection units

value

Type: xs:anySimpleType
Use: required

Example(s):
500000

name

Type: lengthUnits
Use: required

Example(s):
Meter

name

Type: xs:string

EML does not attempt to document the details of the particular projection algorithm. It is incumbent on metadata provider to ensure that the meaning of the names used to identify the method and its parameters will be as clear as possible.
Example(s):
Transverse_Mercator

name

Type: xs:string
Use: required

Complex Type Definitions:

SpatialReferenceType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
A choice of (		
horizCoordSysName	required	
OR		
horizCoordSysDef	required	
)		
vertCoordSys	optional	
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

Description of this field:

A spatial reference description provides the information for relating the positional information in a spatial dataset to real-world locations. A description typically includes identification of a reference datum, a spheroid definition, and projection algorithm. It also provides any constants required by that algorithm.

geogCoordSysType

Content of this field:

Elements:	Use:	How many:
A sequence of (		
datum	required	
spheroid	required	
primeMeridian	required	
unit	required	
)		

Attributes:	Use:	Default Value:
name		

Description of this field:

horizCoordSysType

Content of this field:

Elements:	Use:	How many:
A choice of (		
geogCoordSys	required	
OR		
projCoordSys	required	
)		

Attributes:	Use:	Default Value:
name	required	

Description of this field:

Simple Type Definitions:

lengthUnits

Derived from: xs:string (by xs:restriction)

Allowed values:

- meter

- nanometer
- micrometer
- micron
- millimeter
- centimeter
- decimeter
- dekameter
- hectometer
- kilometer
- megameter
- angstrom
- inch
- Foot_US
- foot
- Foot_Gold_Coast
- fathom
- nauticalMile
- yard
- Yard_Indian
- Link_Clarke
- Yard_Sears
- mile

angleUnits

Derived from: xs:string (by xs:restriction)

Allowed values:

- radian
- degree
- grad
- degree
- grad

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-spatialVector

Back to EML Contents

The eml-spatialVector module - Logical information about non-gridded geospatial image data

The eml-spatialVector module allows for the description of spatial objects in a GIS system that are not defined in a regularly gridded pattern. These geometries include points and vectors and the relationships among them. Specific attributes of a spatial vector can be documented here including the vector's geometry type, count and topology level.

Module details

Recommended Usage:	all spatial datasets that contain spatial data entities represented as vector features.
Stand-alone:	yes
Imports:	eml-documentation, eml-spatialReference, eml-entity, eml-resource, eml-attribute, eml-constraint
Imported By:	
View an image of the schema:	**eml-spatialVector image**

Element Definitions:

spatialVector	**This element has no default value.**
Content of this field:	Description of this field:
Type: SpatialVectorType	Description of a spatial data entity based on features represented by vectors

attributeList	**This element has no default value.**
Content of this field:	Description of this field:
Type: att:AttributeListType	The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	**This element has no default value.**
Content of this field:	Description of this field:
Type: con:ConstraintType	Description of any relational constraints on this entity. For more information see the eml-constraint module.

geometry	**This element has no default value.**
Content of this field:	Description of this field:
Type: GeometryType	Geometric feature classification is based on an enumeration of simple feature classes defined by the OpenGIS consortium and implemented in Geographic Markup Language as simple feature types. **Example(s):** polygon

geometricObjectCount	**This element has no default value.**
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	Total number of the geometric objects occurring in the dataset. **Example(s):** 24

topologyLevel	This element has no default value.
Content of this field: **Type:** TopologyLevel	Description of this field: This element describes the relative complexity of the geometric information in the data file. **Example(s):** geometryOnly

spatialReference	This element has no default value.
Content of this field: **Type:** spref:SpatialReferenceType	Description of this field:

horizontalAccuracy	This element has no default value.
Content of this field: **Type:** DataQuality	Description of this field: Horizontal positional accuracy of the data expressed as either text or quantitative assessment.

verticalAccuracy	This element has no default value.
Content of this field: **Type:** DataQuality	Description of this field: Vertical positional accuracy of the data expressed as either text or quantitative assessment.

accuracyReport	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: A text statement of the data quality, included the means by which it was determined.

quantitativeAccuracyReport	This element has no default value.
Content of this field: **Elements:** **Use:** **How many:** A sequence of (quantitativeAccuracyValue required quantitativeAccuracyMethod required)	Description of this field: A quantitative assessment of the data quality expressed as a value and the method of its determination.

quantitativeAccuracyValue	This element has no default value.
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The value resulting from the accuracy test. Typically, this will be expressed in units corresponding to those declared for the parameter being assessed. **Example(s):** 4.5

quantitativeAccuracyMethod	This element has no default value.
Content of this field:	Description of this field: Identification and explanation of the method used to

Type: res:NonEmptyStringType

calculate the quantitative accuracy assessment.
Example(s):
Error expressed as root mean square of 5 control points.

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

SpatialVectorType

Content of this field:

Description of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
ent:EntityGroup		
attributeList	required	
constraint	optional	unbounded
geometry	required	unbounded
geometricObjectCount	optional	
topologyLevel	optional	
spatialReference	optional	
horizontalAccuracy	optional	
verticalAccuracy	optional	
)		
OR		
res:ReferencesGroup		
)		

Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

DataQuality

Content of this field: Description of this field:

Elements:	Use:	How many:
A sequence of (		
accuracyReport	required	
quantitativeAccuracyReport	optional	unbounded
)		

Simple Type Definitions:

GeometryType

Derived from: xs:string (by xs:restriction)

Allowed values:

- Point
- LineString
- LinearRing
- Polygon
- MultiPoint
- MultiLineString
- MultiPolygon
- MultiGeometry

TopologyLevel

Derived from: xs:string (by xs:restriction)

Allowed values:

- geometryOnly
- nonPlanarGraph
- planarLineGraph
- fullPlanarGraph
- surfaceGraph
- fullTopology3D

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-storedProcedure

Back to EML Contents

The eml-storedProcedure module - Data tables resulting from procedures stored in a database

The storedProcedure module is meant to capture information on procedures that produce data output in the form of a data table. In an RDBMS one can code complex queries and transactions into stored procedures and then invoke them directly from front-end applications. It allows the optional description of any parameters that are expected to be passed to the procedure when it is called.

Module details

Recommended Usage:	Use the storedProcedure module to document datasets that use storedProcedures to retrieve archived data.
Stand-alone:	yes
Imports:	eml-entity, eml-documentation, eml-attribute, eml-protocol, eml-physical, eml-coverage, eml-resource, eml-constraint
Imported By:	
View an image of the schema:	**eml-storedProcedure image**

Element Definitions:

storedProcedure	This element has no default value.
Content of this field: **Type:** StoredProcedureType	Description of this field: The storedProcedure element is meant to capture information on procedures that produce data output in the form of a data table. In an RDBMS one can code complex queries and transactions into stored procedures and then invoke them directly from front-end applications. This element allows the optional description of any parameters that are expected to be passed to the procedure when it is called. A common use of a stored procedure is to rotate a data table from attributes in columns to attributes in rows for statistical analysis.

attributeList	This element has no default value.
Content of this field: **Type:** att:AttributeListType	Description of this field: The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	This element has no default value.
Content of this field: **Type:** con:ConstraintType	Description of this field: Description of any relational constraints on ' this entity. For more information see the eml-constraint module.

parameter	This element has no default value.
Content of this field: **Type:** ParameterType	Description of this field: The parameter elements defines the fields that may be required to invoke a stored procedure.

name	This element has no default value.

Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The value of the name field is the name of the parameter. **Example(s):** SiteID

domainDescription	This element has no default value.
Content of this field:	Description of this field:
Type: res:NonEmptyStringType	The domainDescription field is used to describe the valid entries for the parameter being described. A stored procedure will work only when the parameter's value corresponds to an actual value in a database. **Example(s):** The database has SiteId values that range from cap1 to cap10, however the allowable SiteId values for this stored procedure are cap1,cap2,cap4 or cap7.

required	This element has no default value.
Content of this field:	Description of this field:
Type: xs:boolean	The required field is used to document whether the parameter being described is or is not required when invoking the stored procedure. **Example(s):** true

repeats	This element has no default value.
Content of this field:	Description of this field:
Type: xs:boolean	The repeats field is used to document whether or not the parameter being described can be repeated when invoking the stored procedure. **Example(s):** true

Attribute Definitions:

id

Type: res:IDType

Use: optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

StoredProcedureType			
Content of this field:			Description of this field:
Elements:	**Use:**	**How many:**	The StoredProcedureType complex type defines the structure for documenting a stored procedure.
A choice of (			
A sequence of (			
ent:EntityGroup			
attributeList	required		
constraint	optional	unbounded	
parameter	optional	unbounded	
)			
OR			
res:ReferencesGroup			
)			
Attributes:	**Use:**	**Default Value:**	
id	optional		
system	optional		
scope	optional	document	

ParameterType			
Content of this field:			Description of this field:
Elements:	**Use:**	**How many:**	The parameter complex type defines the structure for documenting the parameters that may be required to invoke a stored procedure.
A sequence of (			
name	required		
domainDescription	required		
required	required		
repeats	required		
)			

Group Definitions:

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-text

Back to EML Contents

The eml-text module - Text field formatting

The eml-text module is a wrapper container that allows general text descriptions to be used within the various modules of eml. It can include either structured or unstructured text blocks. It isn't really appropriate to use this module outside of the context of a parent module, because the parent module determines the appropriate context to which this text description applies. The eml-text module allows one to provide structure to a text description in order to convey concepts such as sections (paragraphs), hierarchy (ordered and unordered lists), emphasis (bold, superscript, subscript) etc. The structured elements are a subset of **DocBook** so the predefined DocBook stylesheets can be used to style EML fields that implement this module.

Module details

Recommended Usage:	any module
Stand-alone:	no
Imports:	eml-documentation
Imported By:	
View an image of the schema:	**eml-text image**

Element Definitions:

text	**This element has no default value.**
Content of this field: **Type: TextType**	Description of this field: The "text" element allows for both formatted and unformatted text blocks to be included in EML. It can contain a number of relevant subsections that allow the use of titles, sections, and paragraphs in the text block. This markup is a subset of DocBook.

section	**This element has no default value.**
Content of this field: **Type: SectionType**	Description of this field: The "section" element allows for grouping related paragraphs of text together, with an optional title. This markup is a subset of DocBook.

para	**This element has no default value.**
Content of this field: **Type: ParagraphType**	Description of this field: The "paragraph" element allows for both formatted and unformatted text blocks to be included in EML. It can be plain text or text with a limited set of markup tags, including emphasis, subscript, superscript, and lists. This markup is a subset of DocBook.

itemizedlist	**This element has no default value.**
Content of this field: **Type: ListType**	Description of this field: A list of items in a text paragraph. The list is generally displayed as a bulleted list. This markup is a subset of DocBook.

orderedlist	**This element has no default value.**

Content of this field:

Type: ListType

Description of this field:

An ordered list of items in a text paragraph. The list is generally displayed as a numbered list. This markup is a subset of DocBook.

emphasis	This element has no default value.

Content of this field:

Type: xs:string

Description of this field:

A span of emphasized text in a paragraph. Emphasized text is generally rendered as boldfaced or otherwise distinct from the surrounding text. This markup is a subset of DocBook.

subscript	This element has no default value.

Content of this field:

Type: SubSuperScriptType

Description of this field:

A subscript in a text paragraph. This markup is a subset of DocBook.

superscript	This element has no default value.

Content of this field:

Type: SubSuperScriptType

Description of this field:

A superscript in a text paragraph. This markup is a subset of DocBook.

literalLayout	This element has no default value.

Content of this field:

Type: xs:string

Description of this field:

This element specifies that the structure of the text within the tag, specifically the whitespace, should not be altered.

ulink	This element has no default value.

Content of this field:

Elements:	Use:	How many:
A sequence of (		
citetitle	required	
)		
Attributes:	**Use:**	**Default Value:**
url	optional	

Description of this field:

this element and its children allow paragraphs to contain urls and titles for anchor tags. This markup is a subset of DocBook.

citetitle	This element has no default value.

Content of this field:

Description of this field:

the citetitle element contains a text title for the url. It can be displayed in an anchor tag. This markup is a subset of DocBook.

Example(s):

The Dublin Core Metadata Initiative

title	This element has no default value.

Content of this field:

Type: xs:string

Description of this field:

The optional title for a section. This markup is a subset of DocBook.

para	This element has no default value.
Content of this field: **Type:** ParagraphType	Description of this field: The "paragraph" element allows for both formatted and unformatted text blocks to be included in EML. It can be plain text or text with a limited set of markup tags, including emphasis, subscript, superscript, and lists. This markup is a subset of DocBook.

section	This element has no default value.
Content of this field: **Type:** SectionType	Description of this field: The "section" element allows for grouping related paragraphs of text together, with an optional title. This markup is a subset of DocBook.

listitem	This element has no default value.
Content of this field: **Elements:** **Use:** **How many:** A choice of (**para** required OR **itemizedlist** required OR **orderedlist** required)	Description of this field: An item in a list of items. Each list item is formatted as a bulleted or numbered item depending on the list type in which it resides. List items contain paragraphs which in turn can be plain text or text with a limited set of markup tags, including emphasis, subscript, superscript, and lists. This markup is a subset of DocBook.

para	This element has no default value.
Content of this field: **Type:** ParagraphType	Description of this field: The "paragraph" element allows for both formatted and unformatted text blocks to be included in EML. It can be plain text or text with a limited set of markup tags, including emphasis, subscript, superscript, and lists. This markup is a subset of DocBook.

itemizedlist	This element has no default value.
Content of this field: **Type:** ListType	Description of this field: A list of items in a text paragraph. The list is generally displayed as a bulleted list. This markup is a subset of DocBook.

orderedlist	This element has no default value.
Content of this field: **Type:** ListType	Description of this field: An ordered list of items in a text paragraph. The list is generally displayed as a numbered list. This markup is a subset of DocBook.

subscript	This element has no default value.
Content of this field:	Description of this field:

Type: SubSuperScriptType	A subscript in a text paragraph. This markup is a subset of DocBook.

superscript	This element has no default value.
Content of this field:	Description of this field:
Type: SubSuperScriptType	A superscript in a text paragraph. This markup is a subset of DocBook.

Attribute Definitions:

url	
Use: optional	the url attribute contains the location of the work for a link. This markup is a subset of DocBook. **Example(s):** url="http://dublincore.org/documents/usageguide/"

Complex Type Definitions:

TextType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A choice of (		
section	optional	unbounded
OR		
para	optional	unbounded
)		

The "text" element allows for both formatted and unformatted text blocks to be included in EML. It can contain a number of relevant subsections that allow the use of titles, sections, and paragraphs in the text block. This markup is a subset of DocBook.

ParagraphType	
Content of this field:	Description of this field:

Elements:	Use:	How many:
A choice of (		
itemizedlist	required	
OR		
orderedlist	required	
OR		
emphasis	required	
OR		
subscript	required	
OR		
superscript	required	
OR		
literalLayout	required	
OR		
ulink	optional	
)		

The "paragraph" element allows for both formatted and unformatted text blocks to be included in EML. It can be plain text or text with a limited set of markup tags, including emphasis, subscript, superscript, lists and links. This markup is a subset of DocBook.

SectionType

Content of this field:

Elements:	Use:	How many:
A sequence of (		
title	optional	
A choice of (		
para	required	
OR		
section	required	
)		
)		

Description of this field:

The "section" element allows for grouping related paragraphs (or other sections) of text together, with an optional title. This markup is a subset of DocBook.

ListType

Content of this field:

Elements:	Use:	How many:
A sequence of (		
listitem	required	unbounded
)		

Description of this field:

A list of items in a text paragraph. The ListType is used by both orderedlist elements and itemizedlist elements. This markup is a subset of DocBook.

SubSuperScriptType

Content of this field:

Elements:	Use:	How many:
A choice of (		
subscript	required	
OR		
superscript	required	
)		

Description of this field:

A subscript or a superscript in a text paragraph. This type is used by both subscript and superscript elements to define their recursive content. This markup is a subset of DocBook.

Web Contact: ***jones@nceas.ucsb.edu***

Back to EML Contents

Module Documentation: eml-unitTypeDefinitions

EML Unit Type Definitions

Philosophy of Units

The concept of "unit" represents one of the most fundamental categories of metadata. The classic example of data entropy is the case in which a reported numeric value loses meaning due to lack of associated units. Much of Ecology is driven by measurement, and most measurements are inherently comparative. Good data description requires a representation of the basis for comparison, i.e., the unit. In modeling units, the authors of EML drew inspiration from the **NIST Reference on Constants, Units, and Uncertainty**. This document defines a unit as "a particular physical quantity, defined and adopted by convention, with which other particular quantities of the same kind are compared to express their value." The authors of the EML 2 specification (hereafter "the authors") decided to make the unit element required, wherever possible.

The units are defined in the **STMML language** in a document that is shipped with each release of EML. See the accompanying STMML file, eml-unitDictionary.xml, for precise, quantitative definitions of each of these units and their relationships to base SI units

which modules use these types (and which could but dont yet?)

anything else?

Module details

Recommended Usage:	any module that needs units
Stand-alone:	no
Imports:	eml-documentation
Imported By:	

Simple Type Definitions:

StandardUnitDictionary

The unitDictionary is the standard set of units included with the EML distribution, mainly from the SI standard. These unit names should be used in the standardUnit field to describe an attribute. See the accompanying STMML file, eml-unitDictionary.xml, for precise, quantitative definitions of each of these units and their relationships to base SI units.

The standard Unit Dictionary is built from a union of simpleTypes. This construct allows unit types to be used individually as appropriate in EML content (e.g., LengthUnitType for distances)

LengthUnitType

Derived from: xs:string (by xs:restriction)

Allowed values:

- meter
- nanometer
- micrometer
- micron
- millimeter

The LengthUnitType is the enumerated list of units which are of length type, or have a parentSI of meter. These unit names can be used where ever content should be restricted to a length, such as a distance or altitude. The units are defined in the STMML language in a document that is shipped with each release of EML called eml-unitDictionary.xml. See this file for precise, quantitative definitions of each of these units and their relationships to base SI units.

- centimeter
- decimeter
- dekameter
- hectometer
- kilometer
- megameter
- angstrom
- inch
- Foot_US
- foot
- Foot_Gold_Coast
- fathom
- nauticalMile
- yard
- Yard_Indian
- Link_Clarke
- Yard_Sears
- mile

MassUnitType

Derived from: xs:string (by xs:restriction)

Allowed values:

- kilogram
- nanogram
- microgram
- milligram
- centigram
- decigram
- gram
- dekagram
- hectogram
- megagram
- tonne
- pound
- ton

The MassUnitType is the enumerated list of units which are of mass type, or have a parentSI of kilogram. These unit names can be used where ever content should be restricted to a mass, such as an amount. The units are defined in the STMML language in a document that is shipped with each release of EML called eml-unitDictionary.xml. See this file for precise, quantitative definitions of each of these units and their relationships to base SI units.

otherUnitType

Derived from: xs:string (by xs:restriction)

Allowed values:

- dimensionless
- second
- kelvin
- coulomb
- ampere
- mole
- candela
- number

The unitDictionary is the standard set of units included with the EML distribution, mainly from the SI standard. These unit names can be used in the standardUnit field to describe an attribute. The units are defined in the STMML language in a document that is shipped with each release of EML. See the accompanying STMML file eml-unitDictionary.xml for precise, quantitative definitions of each of these units and their relationships to base SI units.

The standard Unit Dictionary is built from a union of simpleTypes. This Type enumerates the units which are not in other Type definitions, but are to be included as standard.

- radian
- degree
- grad
- cubicMeter
- nominalMinute
- nominalHour
- nominalDay
- nominalWeek
- nominalYear
- nominalLeapYear
- celsius
- fahrenheit
- nanosecond
- microsecond
- millisecond
- centisecond
- decisecond
- dekasecond
- hectosecond
- kilosecond
- megasecond
- minute
- hour
- kiloliter
- microliter
- milliliter
- liter
- gallon
- quart
- bushel
- cubicInch
- pint
- megahertz
- kilohertz
- hertz
- millihertz
- newton
- joule
- calorie
- britishThermalUnit
- footPound
- lumen
- lux
- becquerel
- gray
- sievert
- katal
- henry
- megawatt
- kilowatt
- watt
- milliwatt
- megavolt
- kilovolt

- volt
- millivolt
- farad
- ohm
- ohmMeter
- siemen
- weber
- tesla
- pascal
- megapascal
- kilopascal
- atmosphere
- bar
- millibar
- kilogramsPerSquareMeter
- gramsPerSquareMeter
- milligramsPerSquareMeter
- kilogramsPerHectare
- tonnePerHectare
- poundsPerSquareInch
- kilogramPerCubicMeter
- milliGramsPerMilliLiter
- gramsPerLiter
- milligramsPerCubicMeter
- microgramsPerLiter
- milligramsPerLiter
- gramsPerCubicCentimeter
- gramsPerMilliliter
- gramsPerLiterPerDay
- litersPerSecond
- cubicMetersPerSecond
- cubicFeetPerSecond
- squareMeter
- are
- hectare
- squareKilometers
- squareMillimeters
- squareCentimeters
- acre
- squareFoot
- squareYard
- squareMile
- litersPerSquareMeter
- bushelsPerAcre
- litersPerHectare
- squareMeterPerKilogram
- metersPerSecond
- metersPerDay
- feetPerDay
- feetPerSecond
- feetPerHour
- yardsPerSecond
- milesPerHour
- milesPerSecond

- milesPerMinute
- centimetersPerSecond
- millimetersPerSecond
- centimeterPerYear
- knots
- kilometersPerHour
- metersPerSecondSquared
- waveNumber
- cubicMeterPerKilogram
- cubicMicrometersPerGram
- amperePerSquareMeter
- amperePerMeter
- molePerCubicMeter
- molarity
- molality
- candelaPerSquareMeter
- metersSquaredPerSecond
- metersSquaredPerDay
- feetSquaredPerDay
- kilogramsPerMeterSquaredPerSecond
- gramsPerCentimeterSquaredPerSecond
- gramsPerMeterSquaredPerYear
- gramsPerHectarePerDay
- kilogramsPerHectarePerYear
- kilogramsPerMeterSquaredPerYear
- molesPerKilogram
- molesPerGram
- millimolesPerGram
- molesPerKilogramPerSecond
- nanomolesPerGramPerSecond
- kilogramsPerSecond
- tonnesPerYear
- gramsPerYear
- numberPerMeterSquared
- numberPerKilometerSquared
- numberPerMeterCubed
- numberPerLiter
- numberPerMilliliter
- metersPerGram
- numberPerGram
- gramsPerGram
- microgramsPerGram
- cubicCentimetersPerCubicCentimeters

angleUnitType

Derived from: xs:string (by xs:restriction)

Allowed values:

- radian
- degree
- grad

The AngleUnitType is the enumerated list of angle units. For example, plane angle (radian, rad) and solid angle (steradian, sr) are actually dimensionless, and their symbols used as appropriate (e.g, sr in photometry). These unit names could be used where ever content should be restricted.

Web Contact: ***jones@nceas.ucsb.edu***

Module Documentation: eml-view

Back to EML Contents

The eml-view module - Data tables resulting from a database query

The eml-view module describes a view from a database management system. A view is a query statement that is stored as a database object and executed each time the view is called.

Module details

Recommended Usage:	all datasets that contain one or more views
Stand-alone:	yes
Imports:	eml-entity, eml-documentation, eml-attribute, eml-protocol, eml-physical, eml-coverage, eml-resource, eml-constraint
Imported By:	
View an image of the schema:	**eml-view image**

Element Definitions:

view	**This element has no default value.**
Content of this field: **Type:** ViewType	Description of this field: The View element is a container for documenting a view. The structure of the view element is defined by the ViewType.

attributeList	**This element has no default value.**
Content of this field: **Type:** att:AttributeListType	Description of this field: The list of attributes associated with this entity. For more information see the eml-attribute module.

constraint	**This element has no default value.**
Content of this field: **Type:** con:ConstraintType	Description of this field: Description of any relational constraints on ' this entity. For more information see the eml-constraint module.

queryStatement	**This element has no default value.**
Content of this field: **Type:** res:NonEmptyStringType	Description of this field: The value of a queryStatement field is the actual query statement stored in the database is entered here. The query statement generates the entity being documented. **Example(s):** Select site as SiteID,common_name as CommonName, count as CountOfIndividuals from samples inner join taxonlist on samples.speciesid=taxonlist.speciesid

Attribute Definitions:

id
Type: res:IDType **Use:** optional

system

Type: res:SystemType

Use: optional

scope

Type: res:ScopeType

Use: optional

Default value: document

Complex Type Definitions:

ViewType

Content of this field:

Elements:	Use:	How many:
A choice of (		
A sequence of (		
ent:EntityGroup		
attributeList	required	
constraint	optional	unbounded
queryStatement	required	
)		
OR		
res:ReferencesGroup		
)		

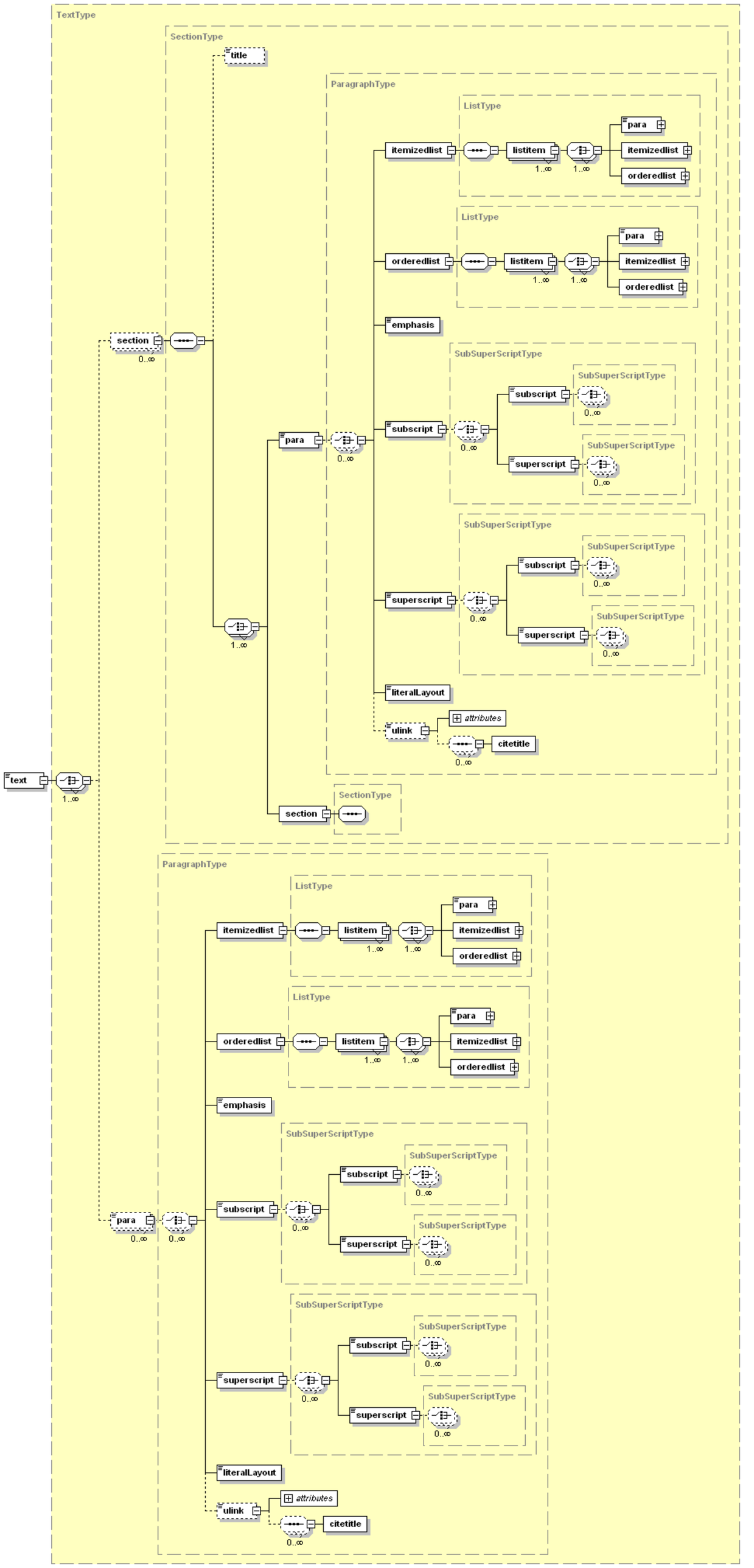
Attributes:	Use:	Default Value:
id	optional	
system	optional	
scope	optional	document

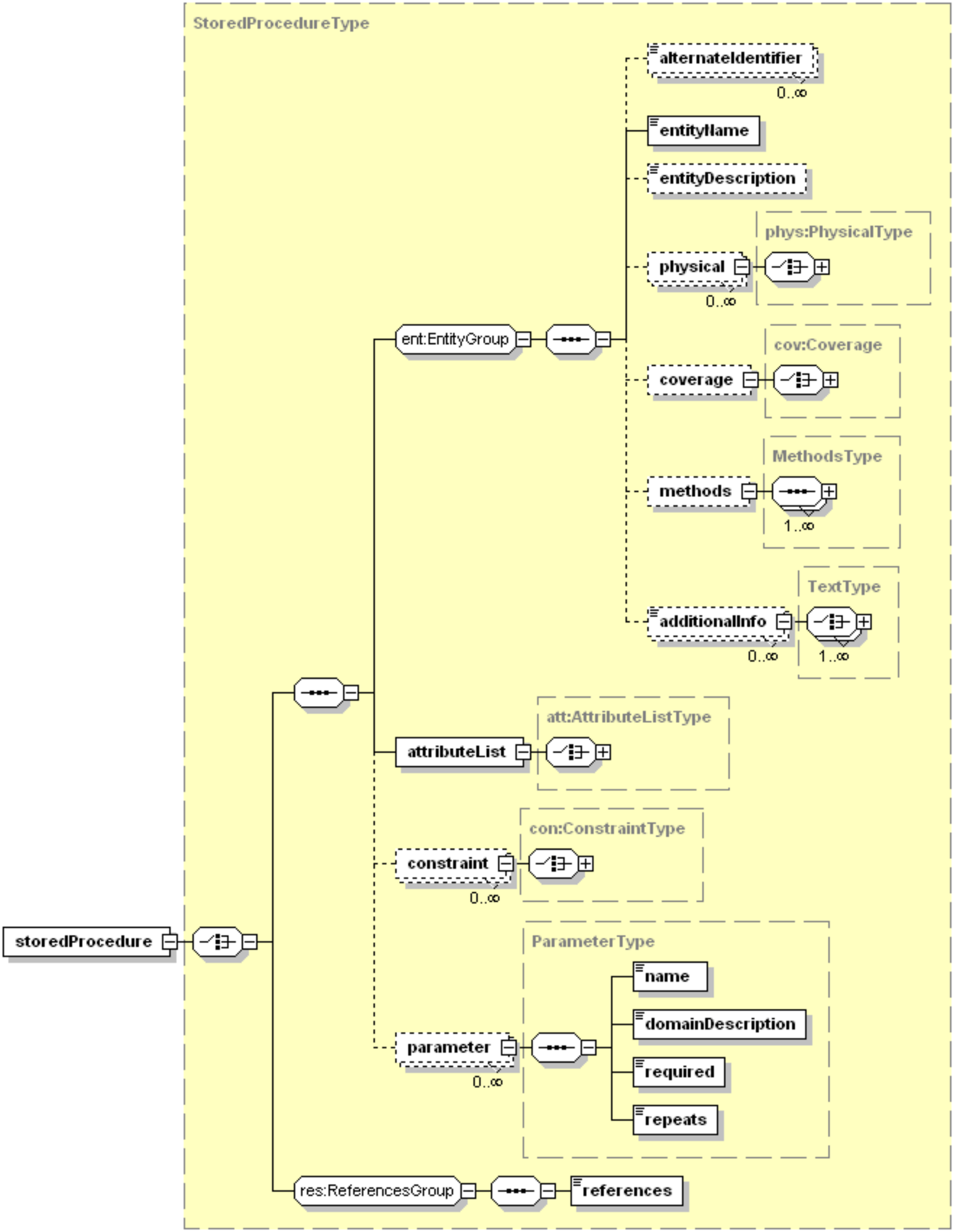
Description of this field:

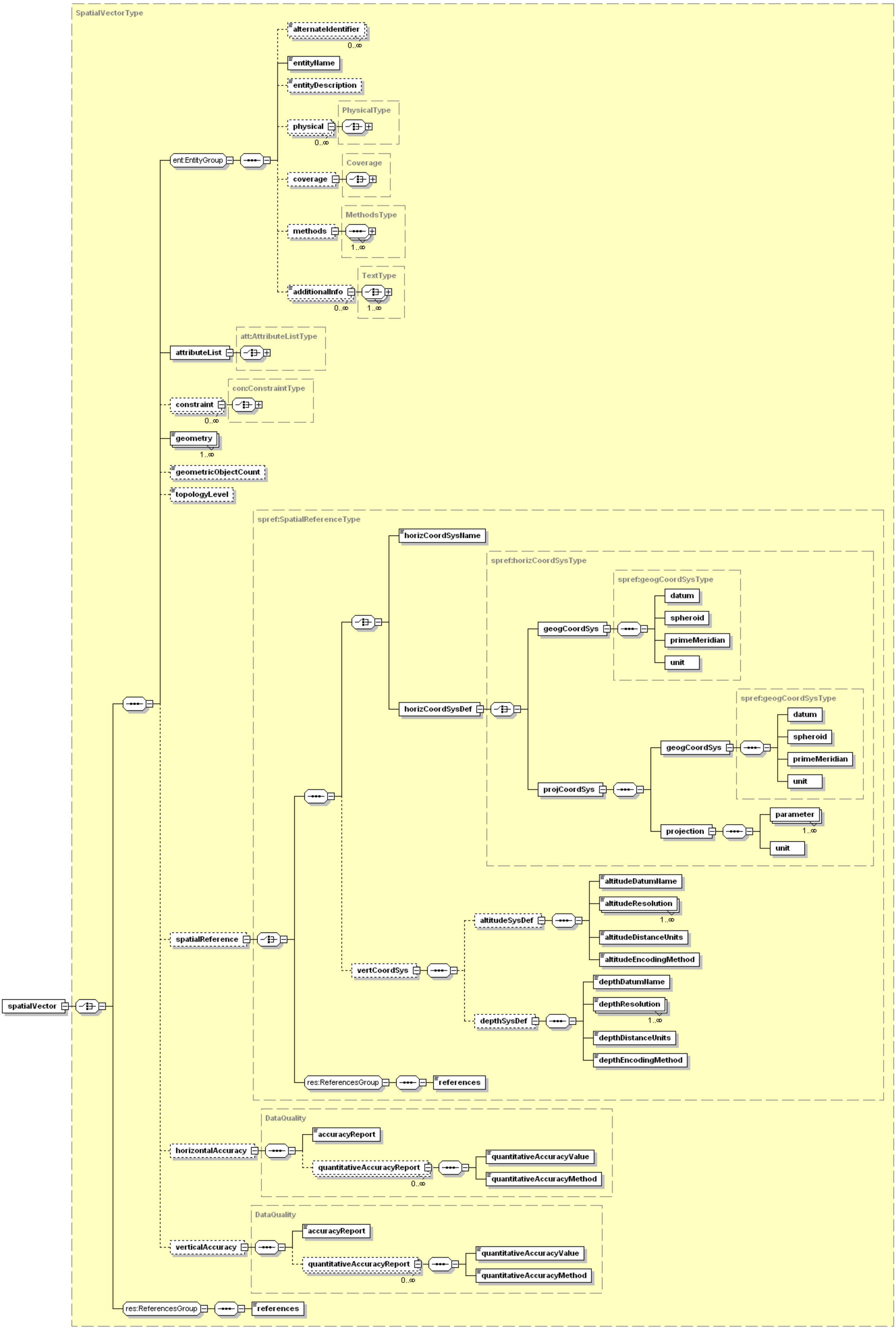
The ViewType complex type defines the structure for documenting a view. This type extends the EntityGroup with a queryStatement.

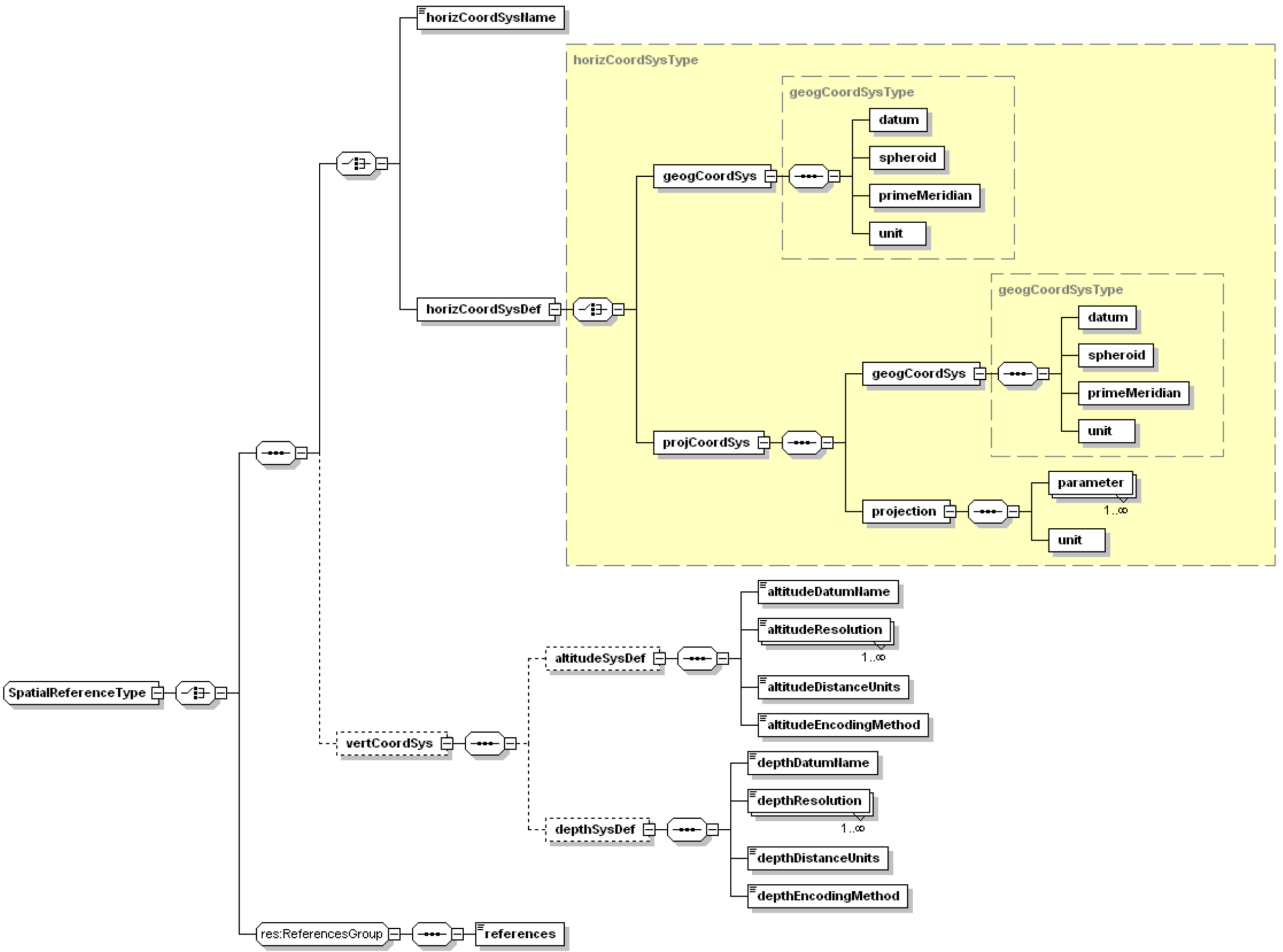
Group Definitions:

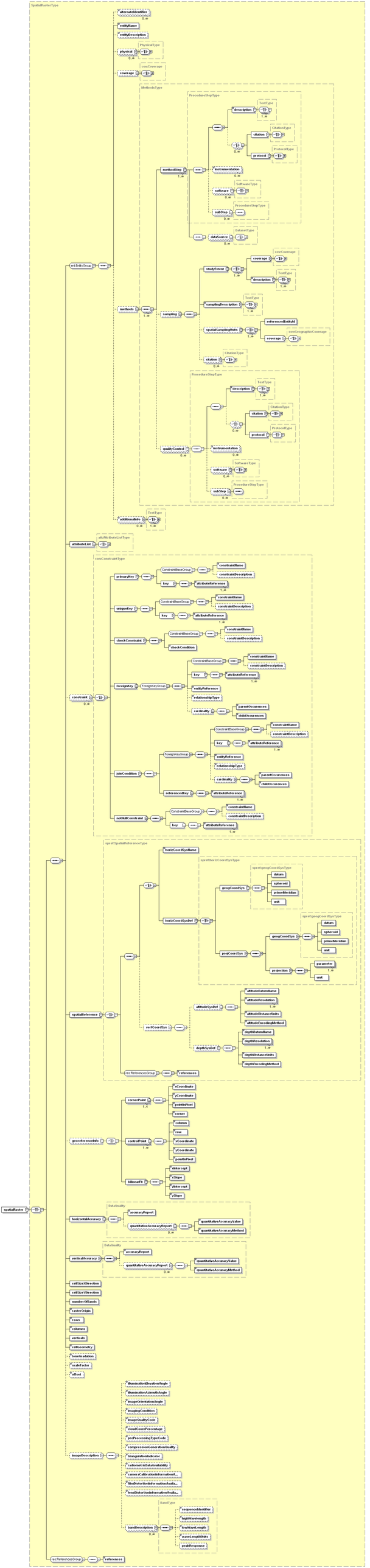
Web Contact: ***jones@nceas.ucsb.edu***


TextType
SectionType
title
ParagraphType
ListType
itemizedlist
listitem
1..∞
para
itemizedlist
orderedlist
ListType
orderedlist
listitem
1..∞
para
itemizedlist
orderedlist
emphasis
SubSuperScriptType
subscript
SubSuperScriptType
superscript
SubSuperScriptType
literalLayout
ulink
attributes
citetitle
0..∞
section
0..∞
para
1..∞
SectionType
text
ParagraphType
ListType
itemizedlist
listitem
1..∞
para
itemizedlist
orderedlist
ListType
orderedlist
listitem
1..∞
para
itemizedlist
orderedlist
emphasis
SubSuperScriptType
subscript
SubSuperScriptType
superscript
SubSuperScriptType
literalLayout
ulink
attributes
citetitle
0..∞
para
0..∞


StoredProcedureType
alternateIdentifier
0..∞
entityName
entityDescription
phys:PhysicalType
physical
0..∞
ent:EntityGroup
cov:Coverage
coverage
MethodsType
methods
1..∞
TextType
additionalInfo
0..∞
1..∞
att:AttributeListType
attributeList
con:ConstraintType
constraint
0..∞
ParameterType
name
domainDescription
required
repeats
parameter
0..∞
storedProcedure
res:ReferencesGroup
references


SpatialVectorType
alternateIdentifier
0..∞
entityName
entityDescription
PhysicalType
physical
0..∞
ent:EntityGroup
Coverage
coverage
MethodsType
methods
1..∞
TextType
additionalInfo
0..∞
1..∞
att:AttributeListType
attributeList
con:ConstraintType
constraint
0..∞
geometry
1..∞
geometricObjectCount
topologyLevel
spref:SpatialReferenceType
horizCoordSysName
spref:horizCoordSysType
spref:geogCoordSysType
datum
spheroid
primeMeridian
unit
geogCoordSys
horizCoordSysDef
projCoordSys
spref:geogCoordSysType
datum
spheroid
primeMeridian
unit
geogCoordSys
projection
parameter
1..∞
unit
spatialReference
vertCoordSys
altitudeSysDef
altitudeDatumName
altitudeResolution
1..∞
altitudeDistanceUnits
altitudeEncodingMethod
depthSysDef
depthDatumName
depthResolution
1..∞
depthDistanceUnits
depthEncodingMethod
res:ReferencesGroup
references
spatialVector
DataQuality
horizontalAccuracy
accuracyReport
quantitativeAccuracyReport
0..∞
quantitativeAccuracyValue
quantitativeAccuracyMethod
DataQuality
verticalAccuracy
accuracyReport
quantitativeAccuracyReport
0..∞
quantitativeAccuracyValue
quantitativeAccuracyMethod
res:ReferencesGroup
references


SpatialReferenceType
horizCoordSysName
horizCoordSysDef
horizCoordSysType
geogCoordSysType
datum
spheroid
primeMeridian
unit
geogCoordSys
projCoordSys
projection
parameter
1..∞
vertCoordSys
altitudeSysDef
altitudeDatumName
altitudeResolution
altitudeDistanceUnits
altitudeEncodingMethod
depthSysDef
depthDatumName
depthResolution
depthDistanceUnits
depthEncodingMethod
res:ReferencesGroup
references


SpatialRasterType
alternateIdentifier
0..∞
entityName
entityDescription
PhysicalType
physical
0..∞
coverage
coneCoverage
MethodsType
ProcedureStepType
TextType
description
1..∞
CitationType
citation
ProtocolType
protocol
methodStep
1..∞
instrumentation
0..∞
SoftwareType
software
0..∞
ProcedureStepType
subStep
0..∞
DataseriesType
dataSource
0..∞
entityGroup
coneCoverage
coverage
studyExtent
1..∞
samplingDescription
spatialSamplingUnits
referenceEntityId
coneGraphicCoverage
methods
1..∞
samplingStep
qualityControl
0..∞
additionalInfo
attributeList
attributeListType
coneConstraintType
primaryKey
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞
uniqueKey
checkConstraint
checkCondition
foreignKey
ForeignKeyGroup
entityReference
relationshipType
cardinality
parentOccurences
childOccurences
constraint
0..∞
joinCondition
referenceKey
notNullConstraint
spatialReference
spatialReferenceType
horizCoordSysName
horizCoordSysDef
spatialGeoCoordSysType
geogCoordSys
datum
spheroid
primeMeridian
unit
projCoordSys
projection
parameter
vertCoordSys
altitudeSysDef
altitudeDatumName
altitudeResolution
altitudeDistanceUnits
altitudeEncodingMethod
depthSysDef
depthDatumName
depthResolution
depthDistanceUnits
depthEncodingMethod
resReferencesGroup
references
georeferenceInfo
cornerPoint
xCoordinate
yCoordinate
pointInPixel
corner
controlPoint
column
row
bilinearFit
xIntercept
xSlope
yIntercept
ySlope
spatialRaster
DataQuality
horizontalAccuracy
accuracyReport
quantitativeAccuracyReport
quantitativeAccuracyValue
quantitativeAccuracyMethod
verticalAccuracy
cellGeometryDirection
cellSizeXDirection
cellSizeYDirection
numberOfBands
rasterOrigin
rows
columns
verticals
cellGeometry
toneGradation
scaleFactor
offset
imageDescription
illuminationElevationAngle
illuminationAzimuthAngle
imageOrientationAngle
imagingCondition
imageQualityCode
cloudCoverPercentage
preProcessingTypeCode
compressionGenerationQuality
triangulationIndicator
radiometricDataAvailability
cameraCalibrationInformationA...
filmDistortionInformationAvaila...
lensDistortionInformationAvaila...
BandType
bandDescription
sequenceIdentifier
bitsPerElement
lowWaveLength
highWaveLength
waveLengthUnits
peakResponse
resReferencesGroup
references

SoftwareType

software

alternateIdentifier 0..∞

shortName

title 1..∞

creator 1..∞ — ResponsibleParty

metadataProvider 0..∞ — ResponsibleParty

associatedParty 0..∞ — ResponsibleParty

role

pubDate

language

series

res:ResourceGroup

abstract 1..∞ — TextType

keywordSet 0..∞

keyword 1..∞

keywordThesaurus

additionalInfo 0..∞ — TextType 1..∞

intellectualRights — TextType 1..∞

distribution 0..∞ — res:DistributionType

coverage — Coverage

distribution 1..∞ — phys:PhysicalDistributionType

size

language 0..∞

LanguageValue

LanguageCodeStandard

operatingSystem 0..∞

machineProcessor 0..∞

virtualMachine

diskUsage

runtimeMemoryUsage

programmingLanguage 0..∞

checksum

dependency 0..∞

action

software

implementation 1..∞

dependency 0..∞

action

software

licenseURL

license

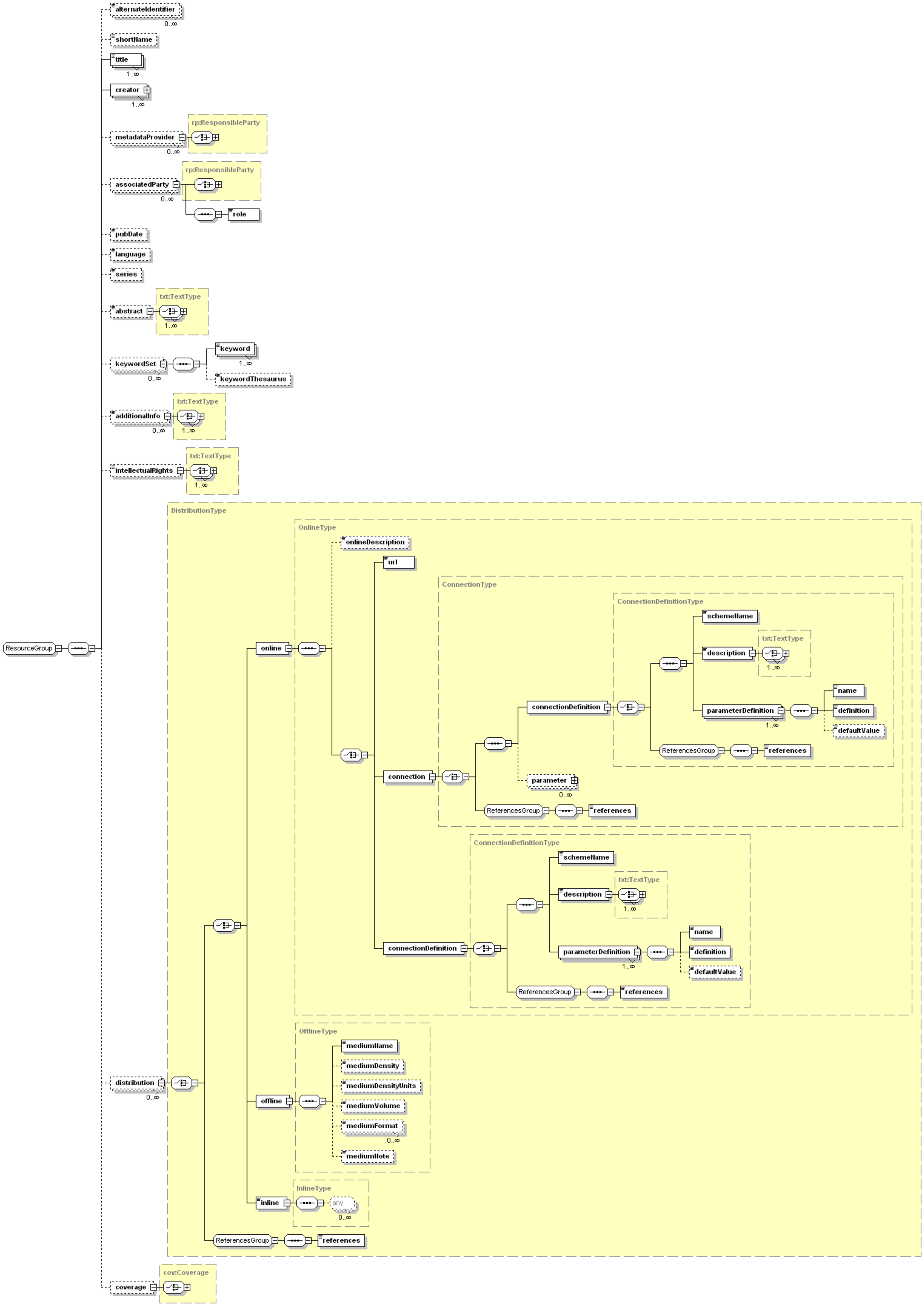
0..∞

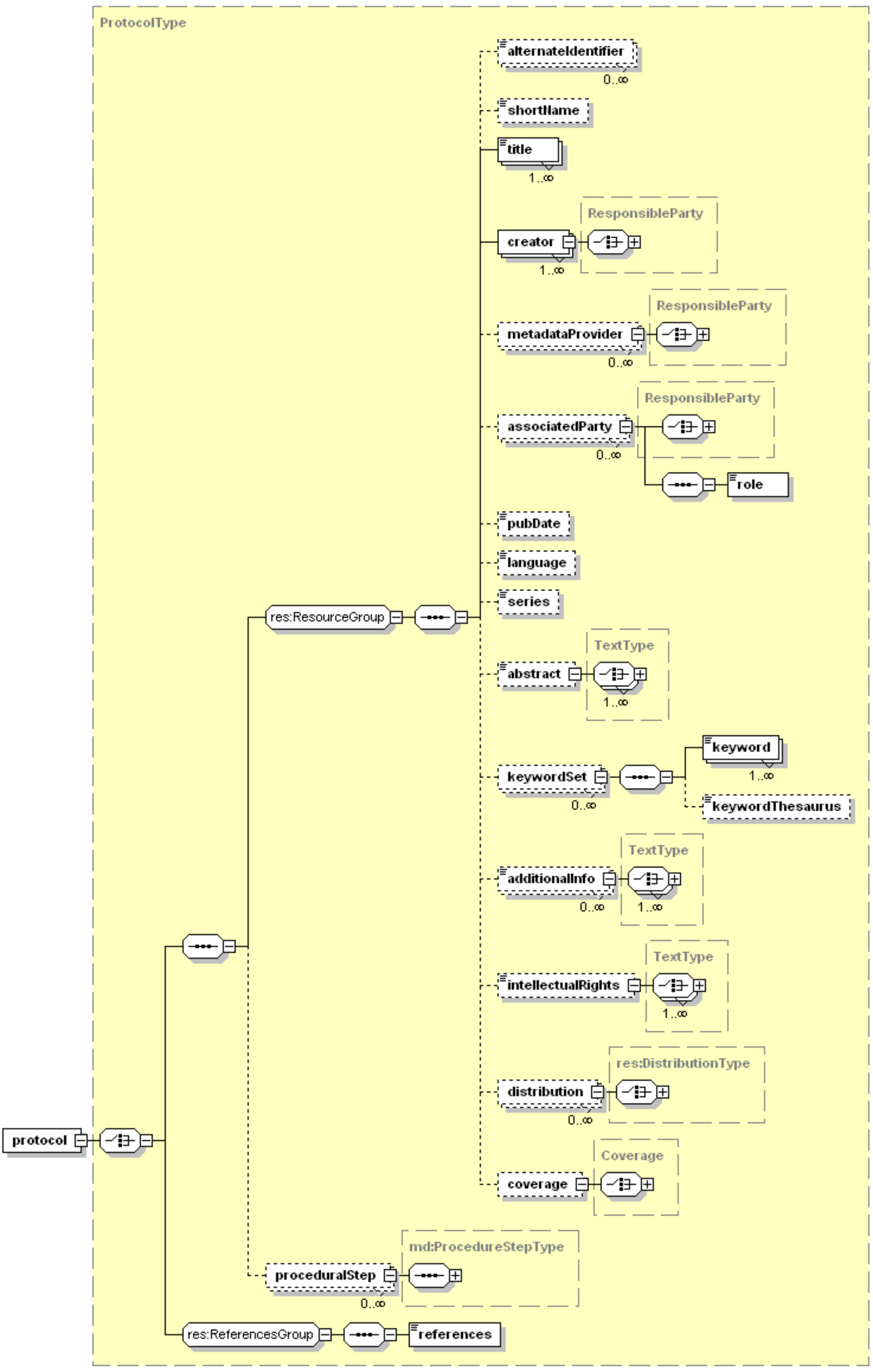
version

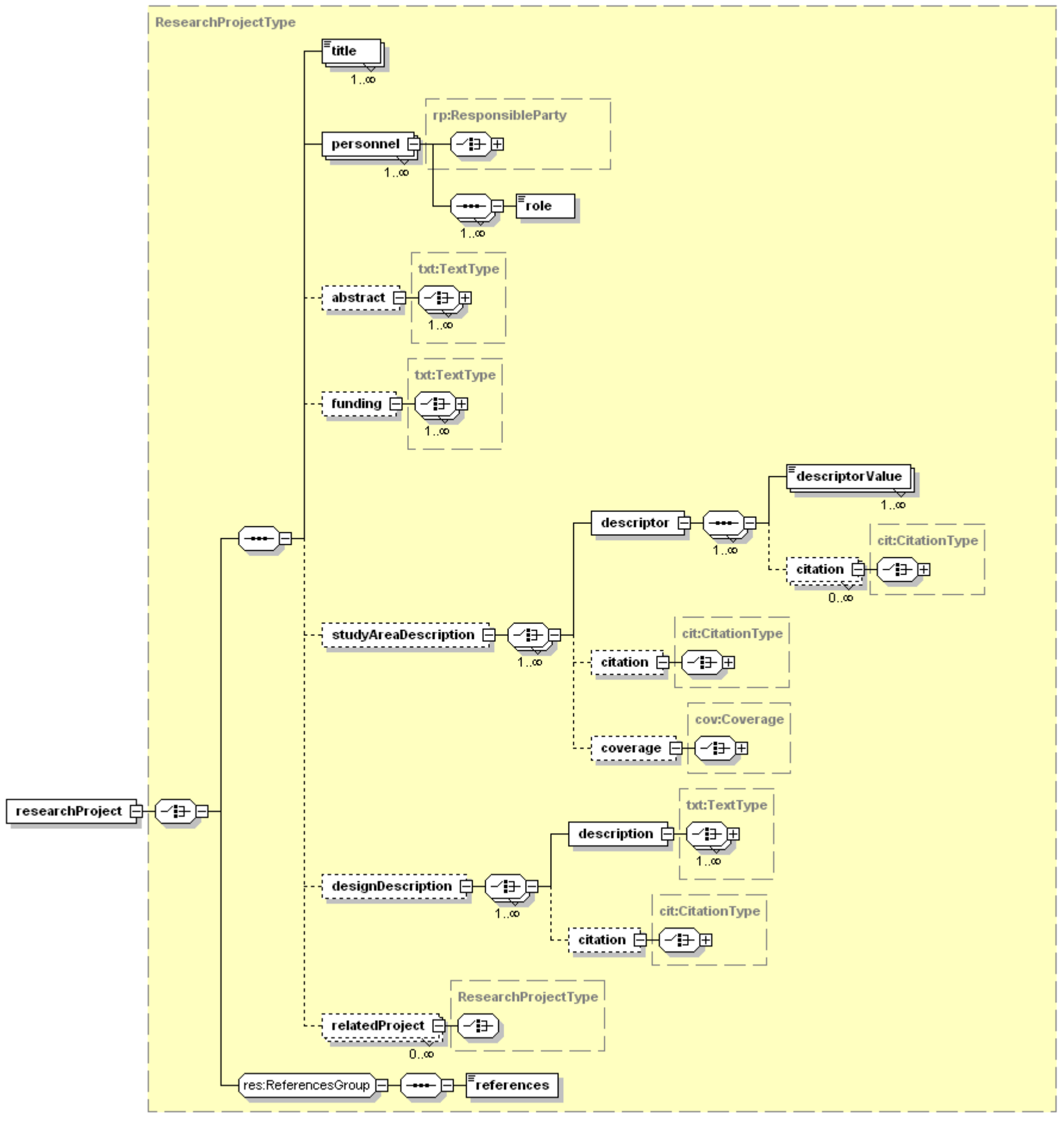
project — proj:ResearchProjectType

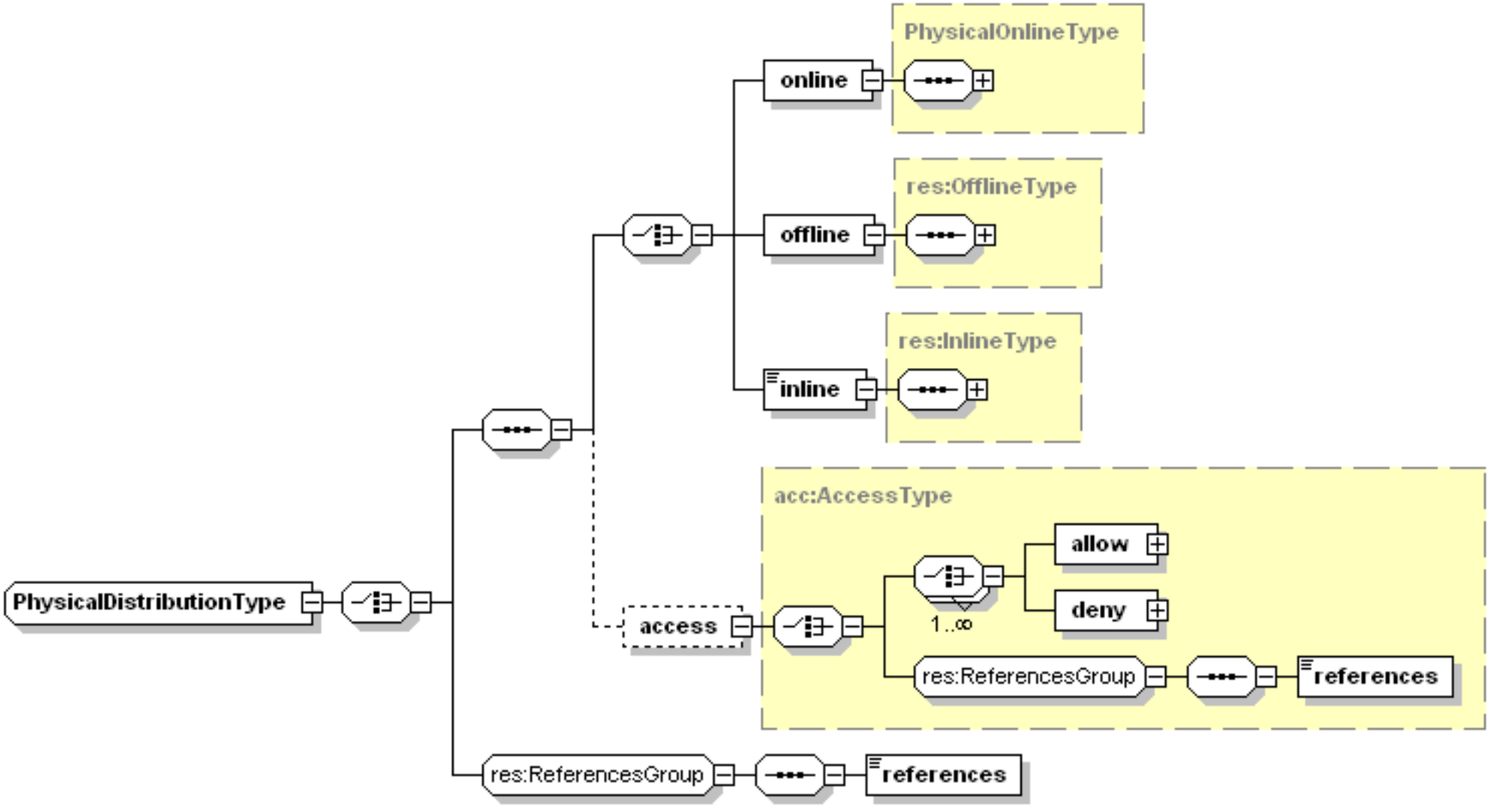
res:ReferencesGroup

references


alternateIdentifier
0..∞
shortName
title
1..∞
creator
1..∞
metadataProvider
rp:ResponsibleParty
0..∞
associatedParty
rp:ResponsibleParty
0..∞
role
pubDate
language
series
abstract
txt:TextType
1..∞
keywordSet
0..∞
keyword
1..∞
keywordThesaurus
additionalInfo
0..∞
txt:TextType
1..∞
intellectualRights
txt:TextType
1..∞
ResourceGroup
DistributionType
OnlineType
onlineDescription
url
online
connection
connectionDefinition
ConnectionDefinitionType
schemeName
description
txt:TextType
1..∞
parameterDefinition
1..∞
name
definition
defaultValue
ReferencesGroup
references
parameter
0..∞
ReferencesGroup
references
connectionDefinition
ConnectionDefinitionType
schemeName
description
txt:TextType
1..∞
parameterDefinition
1..∞
name
definition
defaultValue
ReferencesGroup
references
distribution
0..∞
offline
OfflineType
mediumName
mediumDensity
mediumDensityUnits
mediumVolume
mediumFormat
0..∞
mediumNote
inline
InlineType
any
0..∞
ReferencesGroup
references
coverage
cov:Coverage


ProtocolType
alternateIdentifier
0..∞
shortName
title
1..∞
ResponsibleParty
creator
1..∞
ResponsibleParty
metadataProvider
0..∞
ResponsibleParty
associatedParty
0..∞
role
pubDate
language
series
res:ResourceGroup
TextType
abstract
1..∞
keyword
1..∞
keywordSet
0..∞
keywordThesaurus
TextType
additionalInfo
0..∞
1..∞
TextType
intellectualRights
1..∞
res:DistributionType
distribution
0..∞
Coverage
coverage
protocol
md:ProcedureStepType
proceduralStep
0..∞
res:ReferencesGroup
references


ResearchProjectType
title
1..∞
rp:ResponsibleParty
personnel
1..∞
role
1..∞
txt:TextType
abstract
1..∞
txt:TextType
funding
1..∞
descriptorValue
1..∞
descriptor
1..∞
cit:CitationType
citation
0..∞
studyAreaDescription
1..∞
cit:CitationType
citation
cov:Coverage
coverage
researchProject
txt:TextType
description
1..∞
designDescription
1..∞
cit:CitationType
citation
ResearchProjectType
relatedProject
0..∞
res:ReferencesGroup
references


PhysicalOnlineType
online
res:OfflineType
offline
res:InlineType
inline
acc:AccessType
allow
deny
1..∞
res:ReferencesGroup
references
PhysicalDistributionType
access
res:ReferencesGroup
references

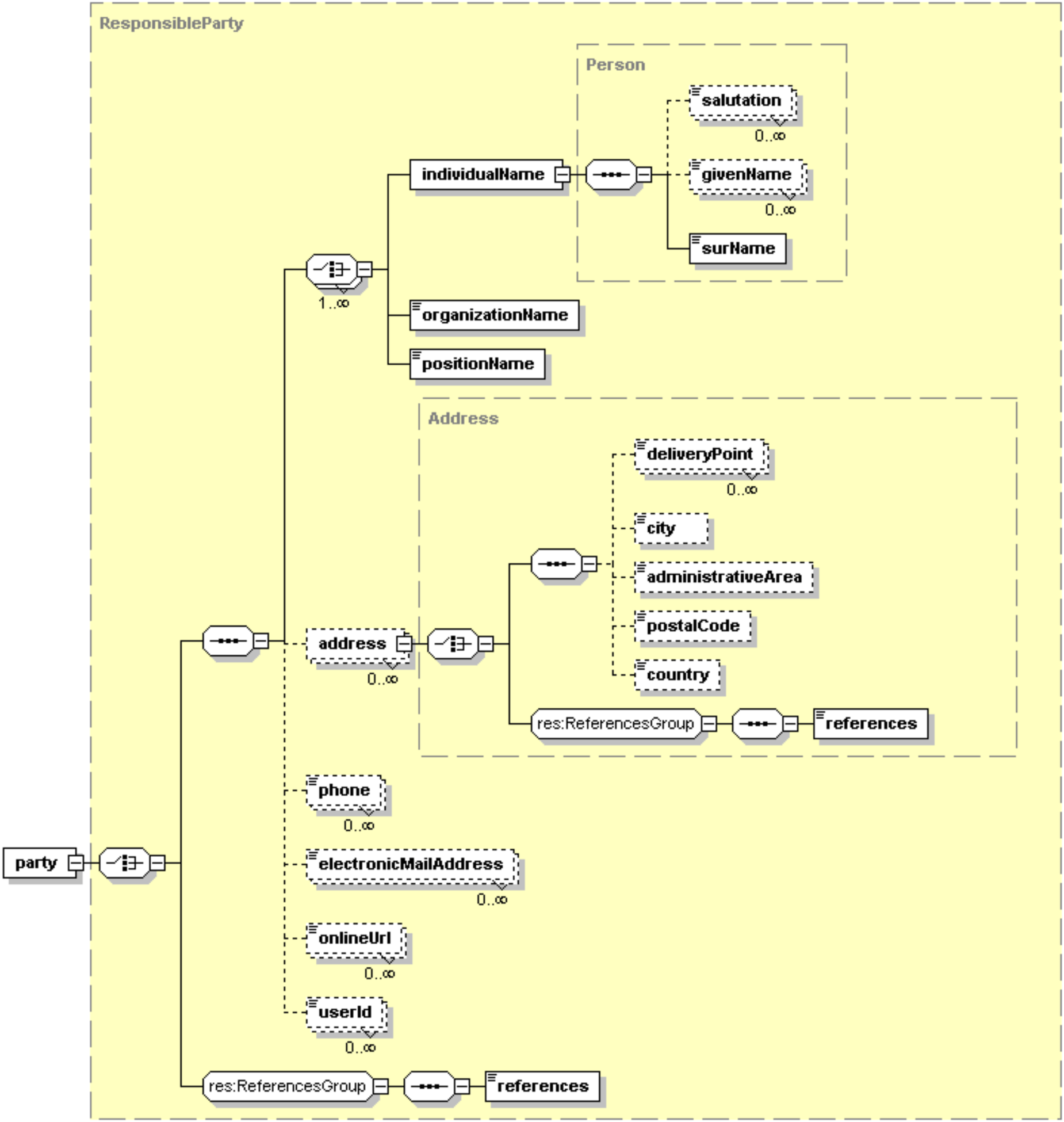
PhysicalType

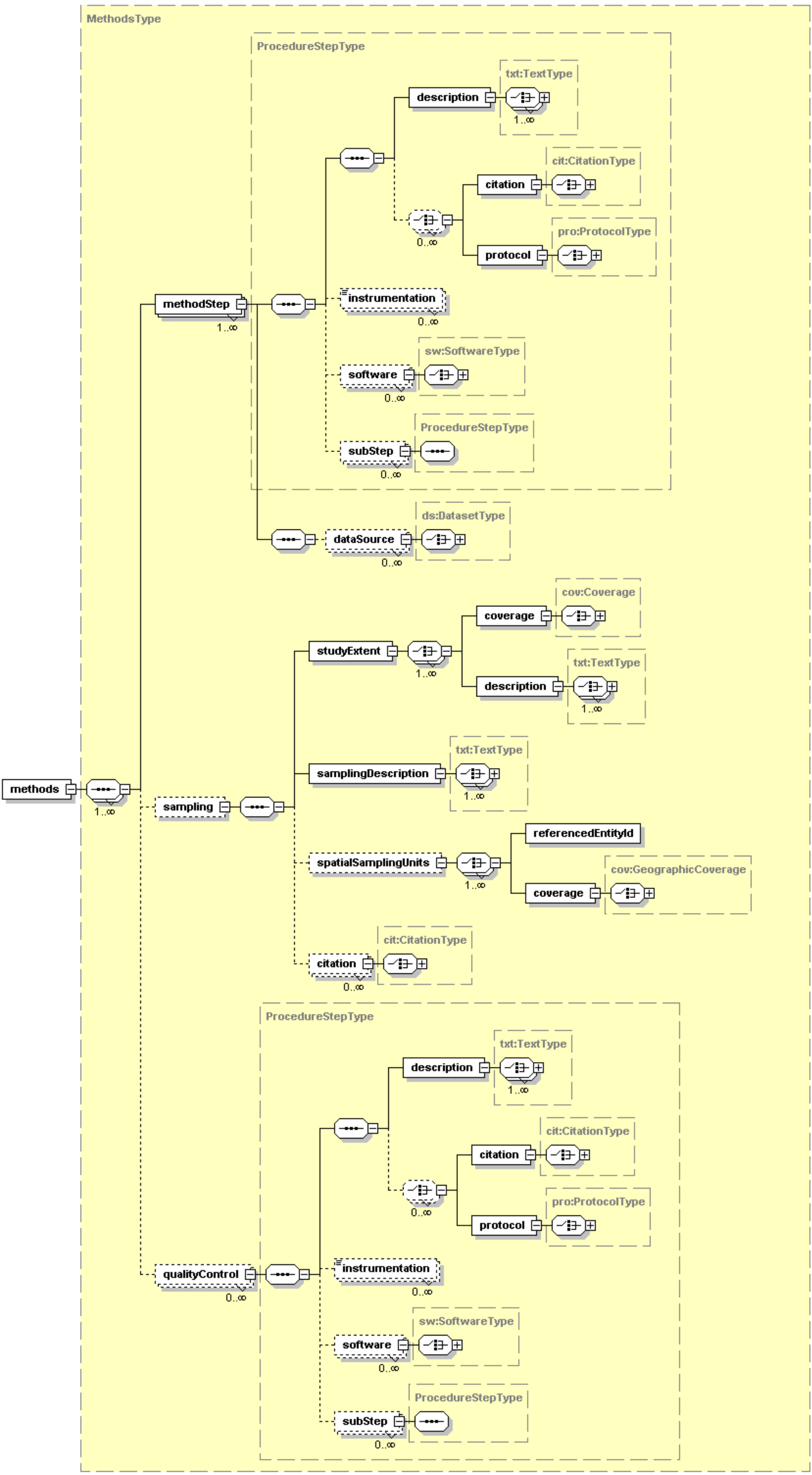
objectName
size
authentication
0..∞
compressionMethod
encodingMethod
0..∞
characterEncoding
numHeaderLines
numFooterLines
recordDelimiter
0..∞
physicalLineDelimiter
0..∞
numPhysicalLinesPerRecord
maxRecordLength
attributeOrientation
textFormat
fieldDelimiter
1..∞
collapseDelimiters
quoteCharacter
0..∞
literalCharacter
0..∞
simpleDelimited
complex
1..∞
fieldWidth
lineNumber
fieldStartColumn
textFixed
fieldDelimiter
collapseDelimiters
lineNumber
quoteCharacter
0..∞
literalCharacter
0..∞
textDelimited
dataFormat
formatName
formatVersion
citation
cit:CitationType
externallyDefinedFormat
rowColumnOrientation
nbands
layout
multiBand
nbits
byteorder
skipbytes
bandrowbytes
totalrowbytes
bandgapbytes
binaryRasterFormat

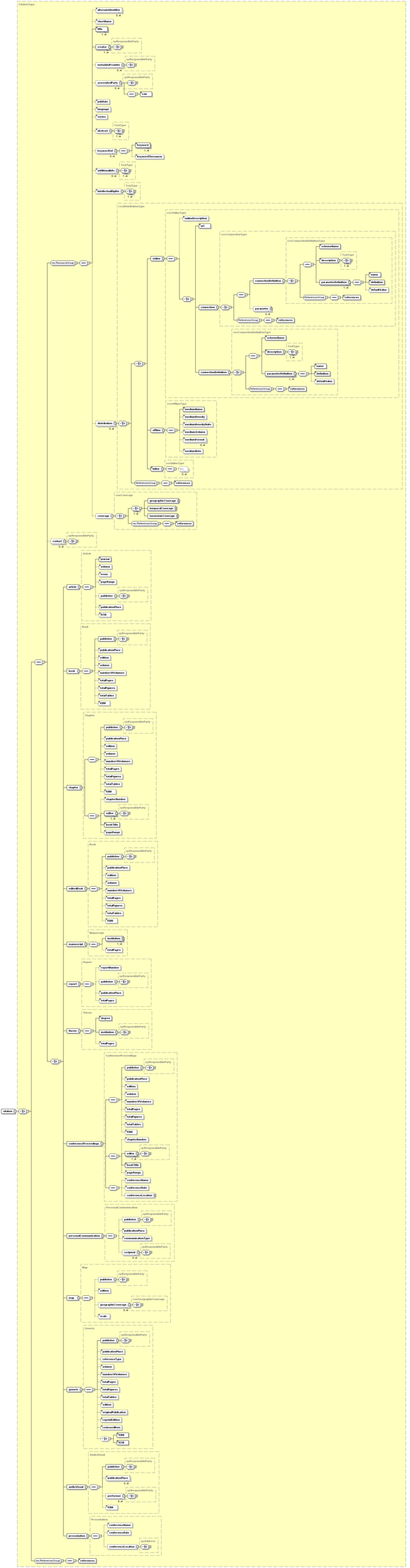
PhysicalDistributionType

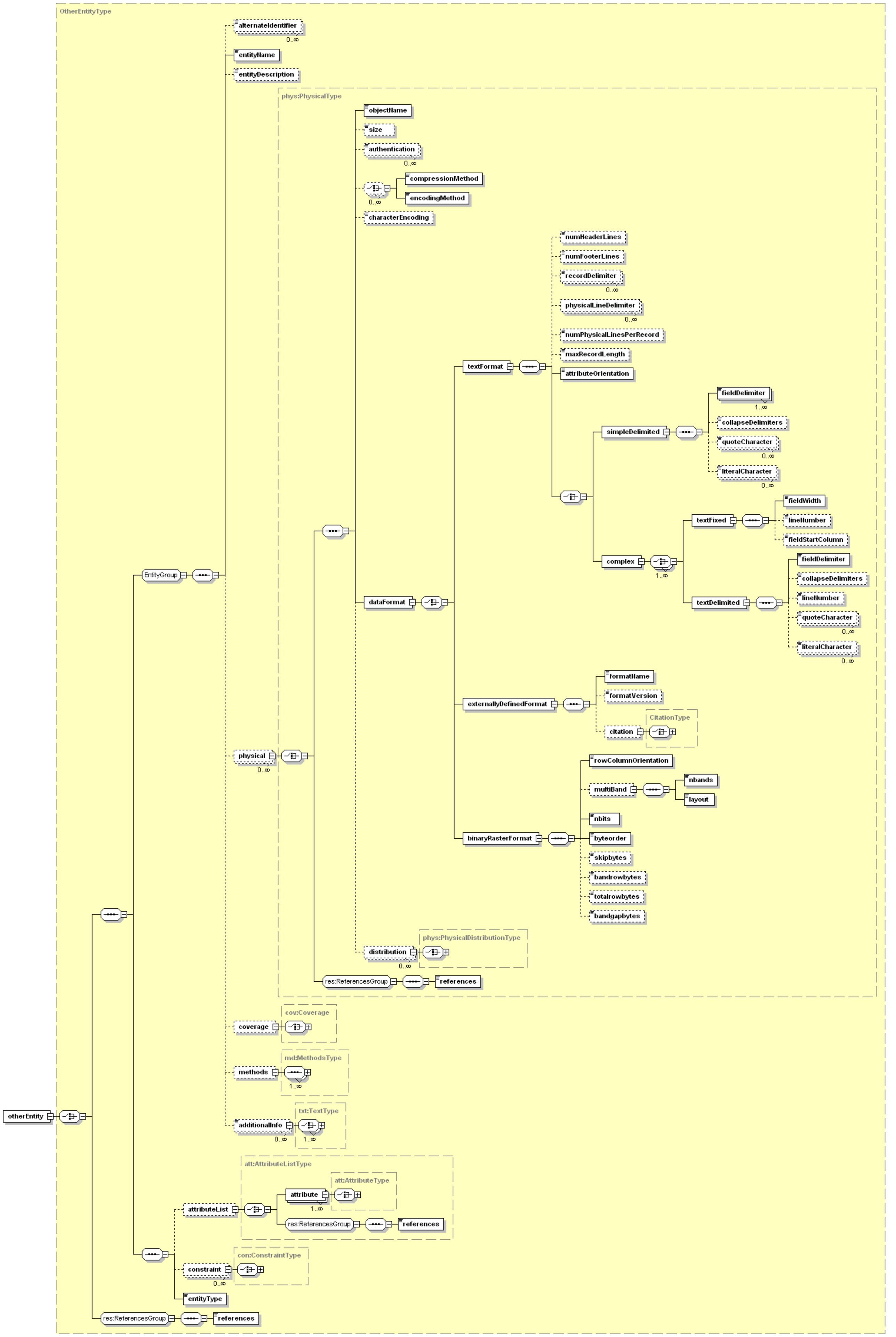
PhysicalOnlineType

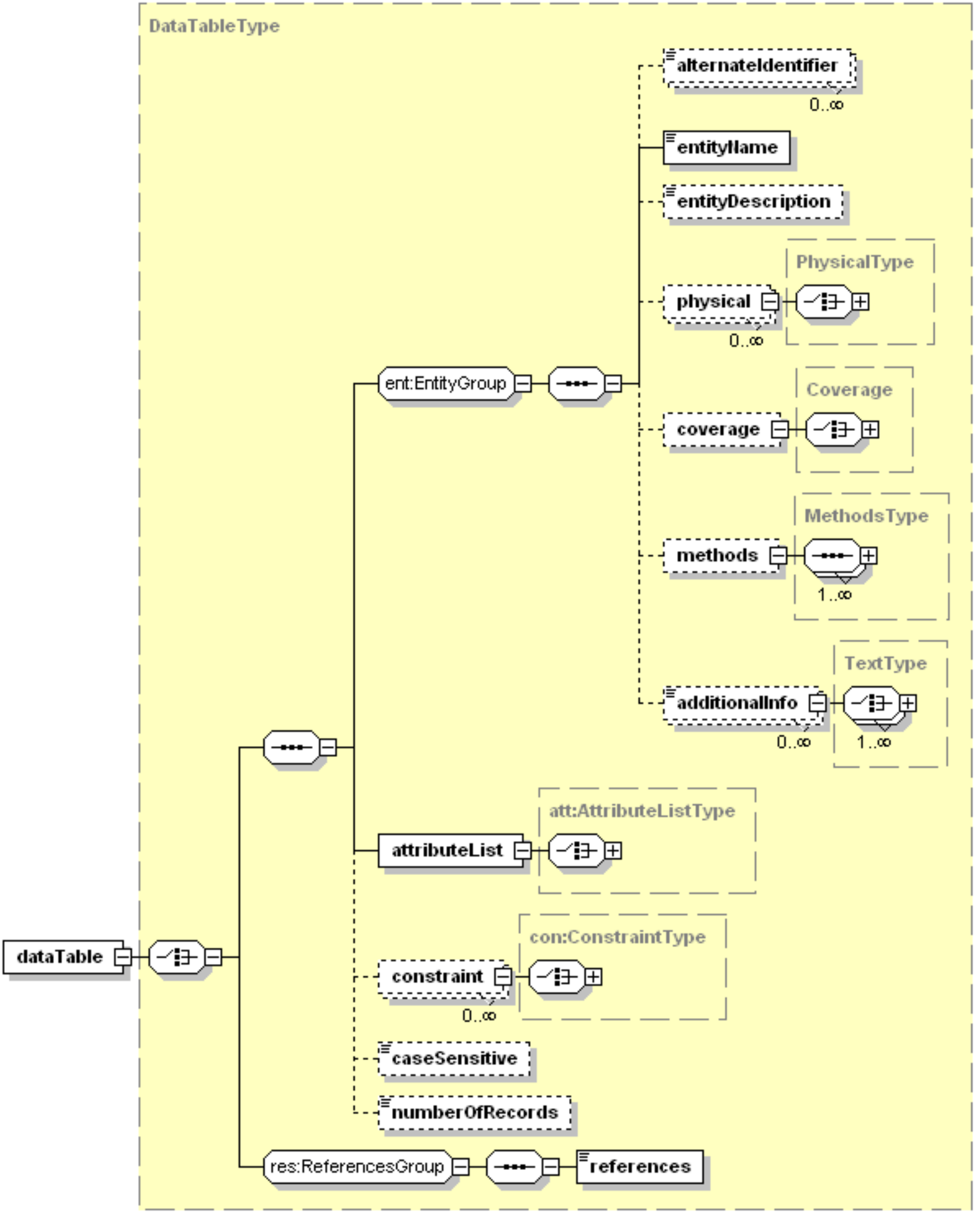
onlineDescription
url
res:ConnectionType
res:ConnectionDefinitionType
schemeName
TextType
description
1..∞
name
definition
defaultValue
parameterDefinition
1..∞
ReferencesGroup
references
connectionDefinition
name
value
parameter
0..∞
ReferencesGroup
references
connection
online
res:OfflineType
mediumName
mediumDensity
mediumDensityUnits
mediumVolume
mediumFormat
0..∞
mediumNote
offline
res:InlineType
any
0..∞
inline
distribution
0..∞
acc:AccessType
allow
deny
1..∞
res:ReferencesGroup
references
access
res:ReferencesGroup
references
physical
res:ReferencesGroup
references


ResponsibleParty
Person
salutation
0..∞
individualName
givenName
0..∞
surName
1..∞
organizationName
positionName
Address
deliveryPoint
0..∞
city
administrativeArea
postalCode
country
address
0..∞
res:ReferencesGroup
references
phone
0..∞
party
electronicMailAddress
0..∞
onlineUrl
0..∞
userId
0..∞
res:ReferencesGroup
references


MethodsType
ProcedureStepType
txt:TextType
description
1..∞
cit:CitationType
citation
0..∞
pro:ProtocolType
protocol
methodStep
1..∞
instrumentation
0..∞
sw:SoftwareType
software
0..∞
ProcedureStepType
subStep
0..∞
ds:DatasetType
dataSource
0..∞
cov:Coverage
coverage
studyExtent
1..∞
txt:TextType
description
1..∞
methods
1..∞
sampling
samplingDescription
txt:TextType
1..∞
referencedEntityId
spatialSamplingUnits
1..∞
cov:GeographicCoverage
coverage
cit:CitationType
citation
0..∞
ProcedureStepType
txt:TextType
description
1..∞
cit:CitationType
citation
0..∞
pro:ProtocolType
protocol
qualityControl
0..∞
instrumentation
0..∞
sw:SoftwareType
software
0..∞
ProcedureStepType
subStep
0..∞


CitationType
alternateIdentifier
0..∞
shortName
title
1..∞
rpt:ResponsibleParty
creator
1..∞
contributorProvider
0..∞
associatedParty
0..∞
role
publDate
language
series
TextType
abstract
1..∞
keywordSet
keyword
1..∞
keywordThesaurus
additionalInfo
0..∞
datesIncludingRights
1..∞
resResourceGroup
resDistributionType
online
offline
inBand
distribution
1..∞
resOnlineType
onlineDescription
url
resConnectionType
resConnectionDefinitionType
schemaName
description
parameterDefinition
name
definition
defaultValue
connection
connectionDefinition
parameter
ReferencesGroup
references
resOfflineType
mediumName
mediumNote
mediumDensity
mediumDensityUnits
mediumVolumes
mediumFormat
resInBandType
coverage
resCoverage
geographicCoverage
temporalCoverage
taxonomicCoverage
resReferencesGroup
content
Article
journal
volume
issue
pageRange
publisher
publicationPlace
ISSN
article
Book
numberOfVolumes
totalPages
totalFigures
totalTables
EISBN
book
Chapter
chapterNumber
editor
bookTitle
chapter
editedBook
Manuscript
institution
totalPages
manuscript
Report
reportNumber
report
Thesis
degree
thesis
ConferenceProceedings
pageRange
conferenceName
conferenceDate
conferenceLocation
conferenceProceedings
PersonalCommunication
communicationType
recipient
personalCommunication
Map
scale
map
Generic
referenceType
generic
AudioVisual
performer
audioVisual
Presentation
presentation
rpt:Address
resReferencesGroup
references


OtherEntityType
alternateIdentifier
0..∞
entityName
entityDescription
phys:PhysicalType
objectName
size
authentication
0..∞
compressionMethod
encodingMethod
0..∞
characterEncoding
numHeaderLines
numFooterLines
recordDelimiter
0..∞
physicalLineDelimiter
0..∞
numPhysicalLinesPerRecord
maxRecordLength
attributeOrientation
textFormat
fieldDelimiter
1..∞
collapseDelimiters
quoteCharacter
0..∞
literalCharacter
0..∞
simpleDelimited
fieldWidth
lineNumber
fieldStartColumn
textFixed
fieldDelimiter
collapseDelimiters
lineNumber
quoteCharacter
0..∞
literalCharacter
0..∞
textDelimited
complex
1..∞
EntityGroup
dataFormat
formatName
formatVersion
CitationType
citation
externallyDefinedFormat
rowColumnOrientation
nbands
multiBand
layout
nbits
byteorder
skipbytes
bandrowbytes
totalrowbytes
bandgapbytes
binaryRasterFormat
physical
0..∞
phys:PhysicalDistributionType
distribution
0..∞
res:ReferencesGroup
references
cov:Coverage
coverage
md:MethodsType
methods
1..∞
txt:TextType
additionalInfo
0..∞
1..∞
att:AttributeListType
att:AttributeType
attribute
1..∞
attributeList
res:ReferencesGroup
references
con:ConstraintType
constraint
0..∞
entityType
otherEntity
res:ReferencesGroup
references


DataTableType
alternateIdentifier
0..∞
entityName
entityDescription
PhysicalType
physical
0..∞
ent:EntityGroup
Coverage
coverage
MethodsType
methods
1..∞
TextType
additionalInfo
0..∞
1..∞
att:AttributeListType
attributeList
dataTable
con:ConstraintType
constraint
0..∞
caseSensitive
numberOfRecords
res:ReferencesGroup
references

DatasetType

alternateIdentifier 0..∞

shortName

title 1..∞

creator 1..∞ — rp:ResponsibleParty

metadataProvider 0..∞ — rp:ResponsibleParty

associatedParty 0..∞ — rp:ResponsibleParty

role

pubDate

language

series

res:ResourceGroup

abstract 1..∞ — txt:TextType

keywordSet 0..∞

keyword 1..∞

keywordThesaurus

additionalInfo 0..∞ — txt:TextType 1..∞

intellectualRights 1..∞ — txt:TextType

distribution 0..∞ — res:DistributionType

coverage — Coverage

purpose 1..∞ — txt:TextType

maintenance — MaintenanceType

description 1..∞ — txt:TextType

maintenanceUpdateFrequency

changeHistory 0..∞

changeScope

oldValue

changeDate

comment

contact 1..∞ — rp:ResponsibleParty

publisher — rp:ResponsibleParty

pubPlace

methods 1..∞ — md:MethodsType

project — proj:ResearchProjectType

dataset

dataTable — dat:DataTableType

spatialRaster — sr:SpatialRasterType

spatialVector — sv:SpatialVectorType

0..∞

storedProcedure — sp:StoredProcedureType

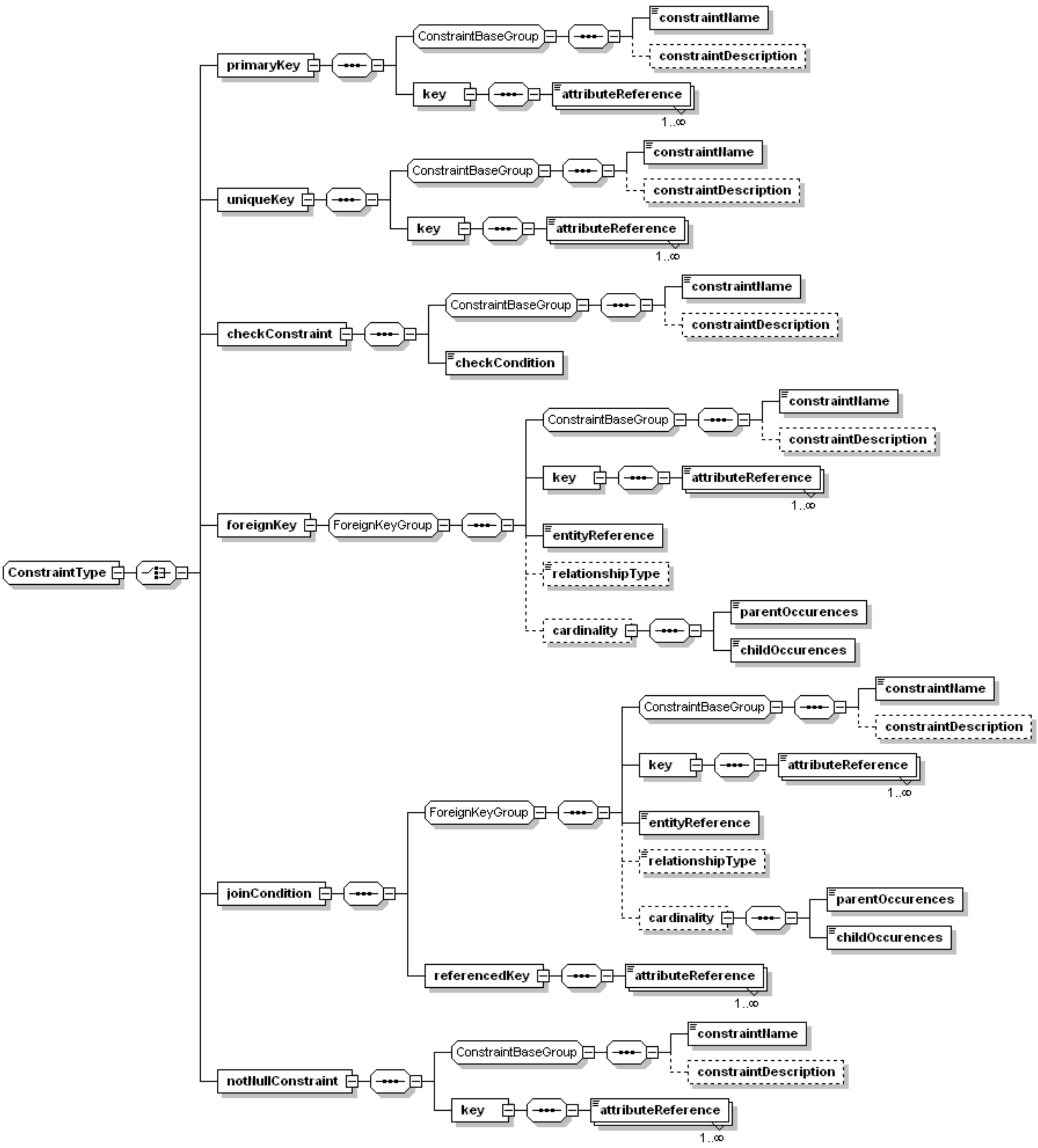
view — v:ViewType

otherEntity — ent:OtherEntityType

res:ReferencesGroup

references

GeographicCoverage
geographicDescription
boundingCoordinates
westBoundingCoordinate
eastBoundingCoordinate
northBoundingCoordinate
southBoundingCoordinate
boundingAltitudes
datasetGPolygon
0..∞
datasetGPolygonOuterGRing
gRingPoint
3..∞
GRingPointType
gRingLatitude
gRingLongitude
gRing
datasetGPolygonExclusionGRing
1..∞
res:ReferencesGroup
references
geographicCoverage
TemporalCoverage
SingleDateTimeType
singleDateTime
calendarDate
time
alternativeTimeScale
timeScaleName
timeScaleAgeEstimate
timeScaleAgeUncertainty
timeScaleAgeExplanation
timeScaleCitation
cit:CitationType
rangeOfDates
beginDate
endDate
temporalCoverage
Coverage
TaxonomicCoverage
classificationSystem
classificationSystemCitation
classificationSystemModificatio...
identificationReference
identifierName
rp:ResponsibleParty
taxonomicSystem
taxonomicProcedures
taxonomicCompleteness
vouchers
specimen
repository
originator
generalTaxonomicCoverage
TaxonomicClassificationType
taxonRankName
taxonRankValue
commonName
taxonomicClassification
taxonomicCoverage


ConstraintType
primaryKey
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞
uniqueKey
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞
checkConstraint
ConstraintBaseGroup
constraintName
constraintDescription
checkCondition
foreignKey
ForeignKeyGroup
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞
entityReference
relationshipType
cardinality
parentOccurences
childOccurences
joinCondition
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞
ForeignKeyGroup
entityReference
relationshipType
cardinality
parentOccurences
childOccurences
referencedKey
attributeReference
1..∞
notNullConstraint
ConstraintBaseGroup
constraintName
constraintDescription
key
attributeReference
1..∞

AttributeType

attributeName

attributeLabel

0..∞

attributeDefinition

storageType

0..∞

attribute

measurementScale

nominal

nonNumericDomain

NonNumericDomainType

enumeratedDomain

codeDefinition

1..∞

code

definition

source

externalCodeSet

codesetName

citation

cit:CitationType

codesetURL

entityCodeList

entityReference

valueAttributeReference

definitionAttributeReference

orderAttributeReference

textDomain

definition

pattern

0..∞

source

res:ReferencesGroup

references

ordinal

nonNumericDomain

NonNumericDomainType

enumeratedDomain

codeDefinition

1..∞

code

definition

source

externalCodeSet

codesetName

citation

cit:CitationType

codesetURL

entityCodeList

entityReference

valueAttributeReference

definitionAttributeReference

orderAttributeReference

textDomain

definition

pattern

0..∞

source

res:ReferencesGroup

references

interval

unit

UnitType

standardUnit

customUnit

precision

numericDomain

NumericDomainType

numberType

BoundsGroup

bounds

0..∞

minimum

maximum

res:ReferencesGroup

references

ratio

unit

UnitType

standardUnit

customUnit

precision

numericDomain

NumericDomainType

numberType

BoundsGroup

bounds

0..∞

minimum

maximum

res:ReferencesGroup

references

dateTime

formatString

dateTimePrecision

dateTimeDomain

DateTimeDomainType

BoundsDateGroup

bounds

0..∞

minimum

maximum

res:ReferencesGroup

references

missingValueCode

0..∞

code

codeExplanation

accuracy

Accuracy

attributeAccuracyReport

quantitativeAttributeAccuracyA...

0..∞

attributeAccuracyValue

attributeAccuracyExplanation

coverage

cov:Coverage

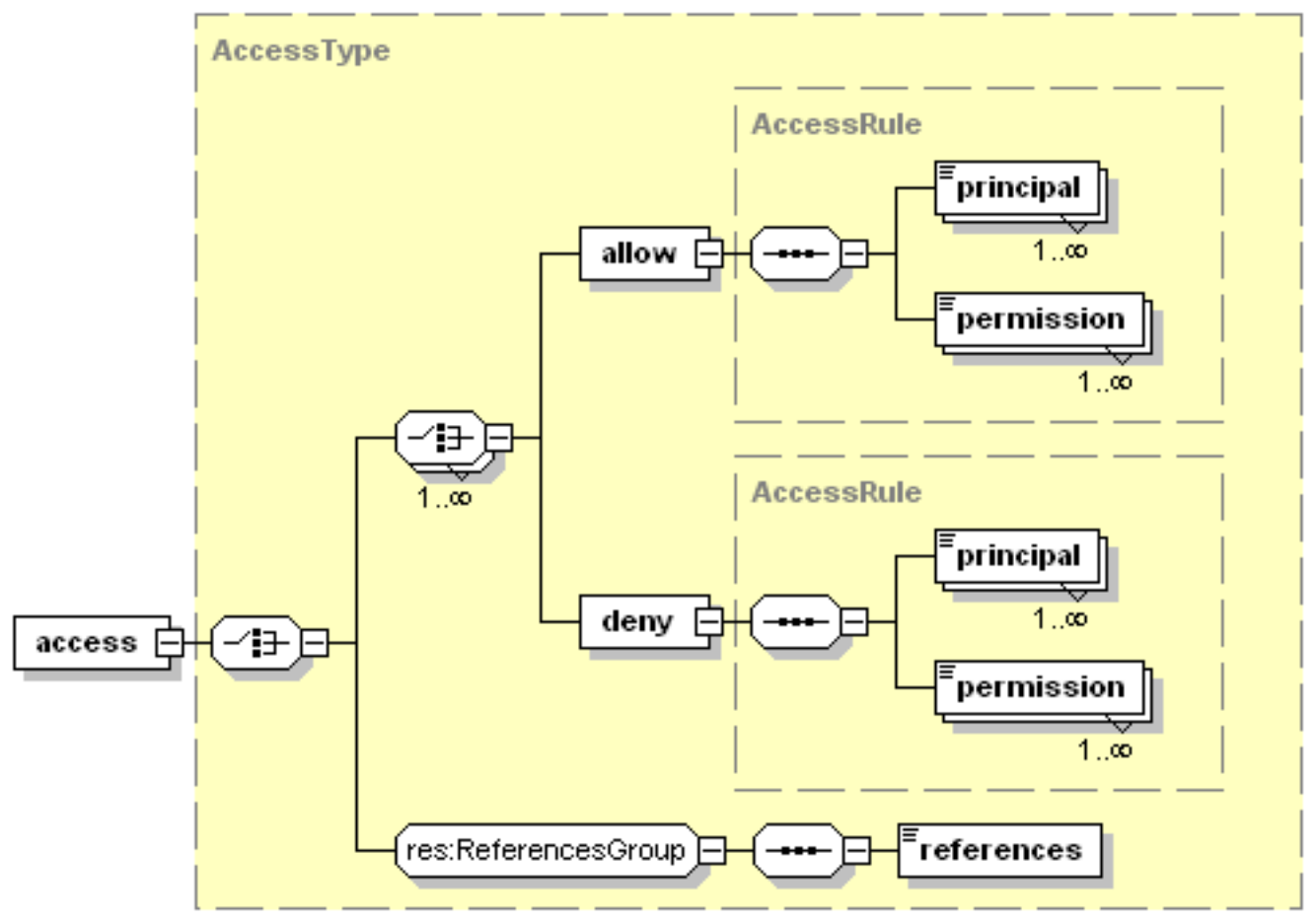
methods

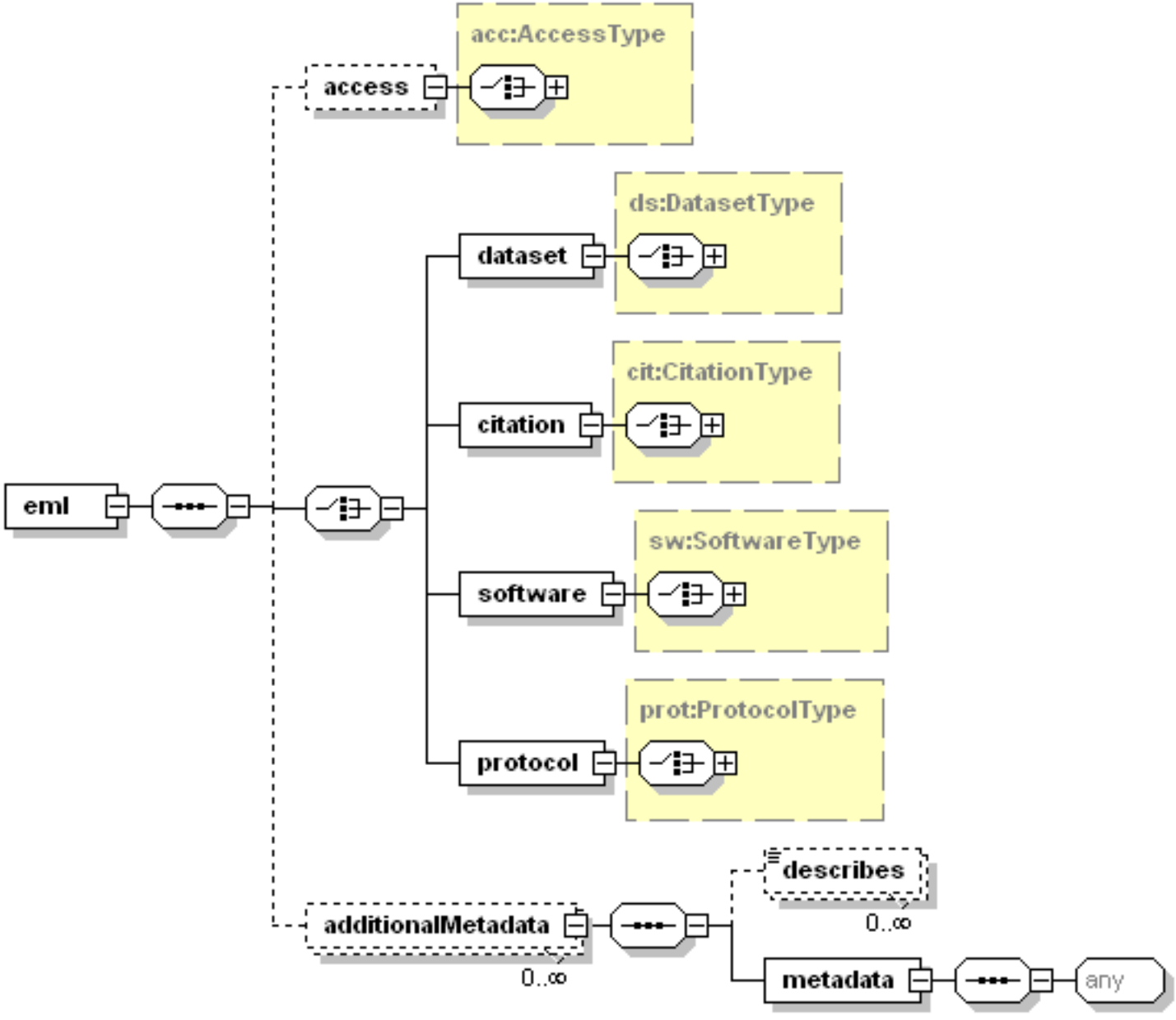
md:MethodsType

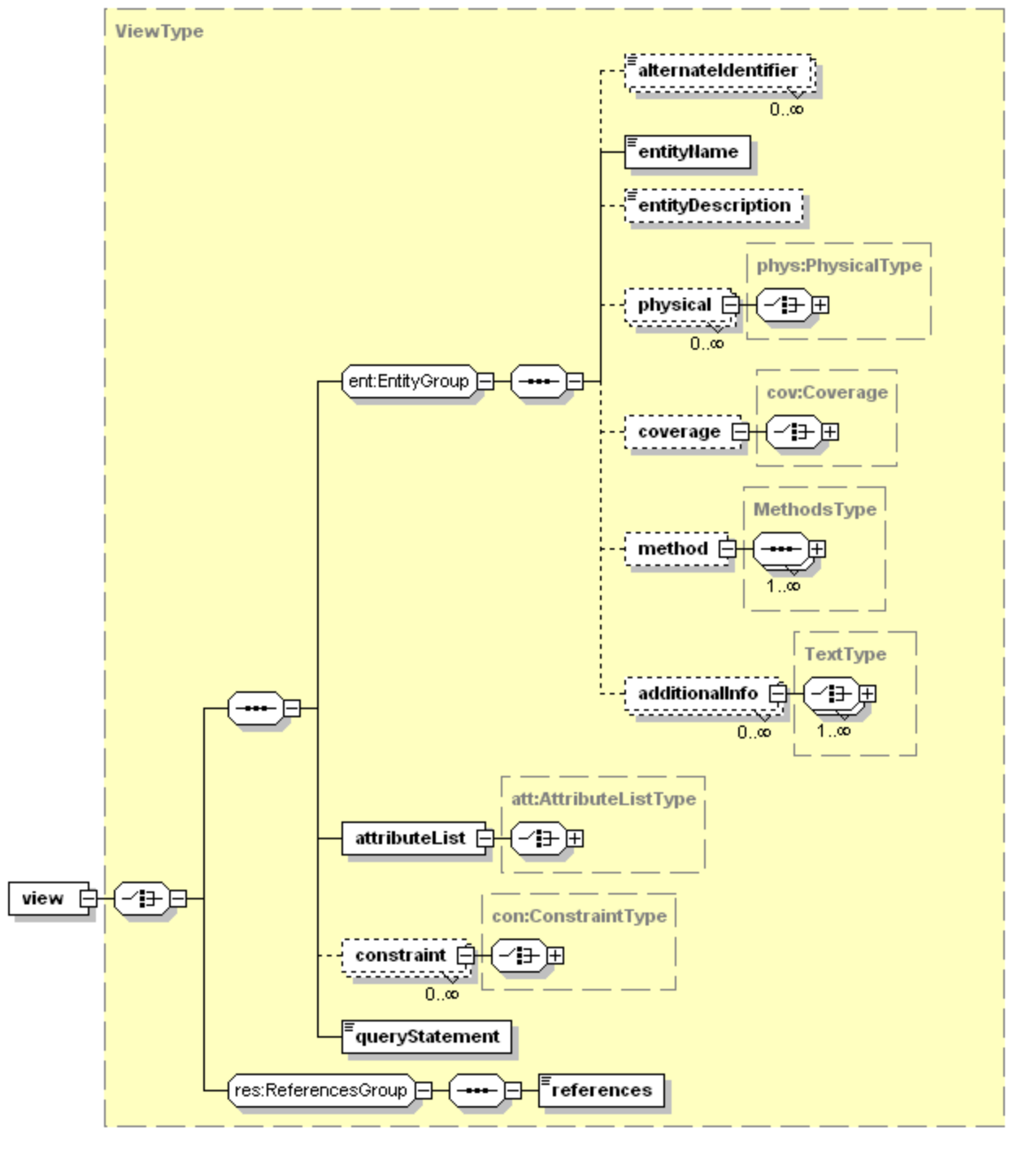
1..∞

res:ReferencesGroup

references


AccessType
AccessRule
principal
1..∞
allow
permission
1..∞
1..∞
AccessRule
principal
1..∞
access
deny
permission
1..∞
res:ReferencesGroup
references


acc:AccessType
access
ds:DatasetType
dataset
cit:CitationType
citation
eml
sw:SoftwareType
software
prot:ProtocolType
protocol
describes
0..∞
additionalMetadata
0..∞
metadata
any


ViewType
alternateIdentifier
0..∞
entityName
entityDescription
phys:PhysicalType
physical
0..∞
ent:EntityGroup
cov:Coverage
coverage
MethodsType
method
1..∞
TextType
additionalInfo
0..∞
1..∞
att:AttributeListType
attributeList
view
con:ConstraintType
constraint
0..∞
queryStatement
res:ReferencesGroup
references