CMR Data Partner User Guide

Chapter 1: Introduction and Scope

Quick transition instructions from ECHO REST ingest

The NASA-developed Earth Observing System (EOS) Common Metadata Repository (CMR) is a spatial and temporal metadata registry that stores metadata from a variety of science disciplines and domains —including Climate Variability and Change, Carbon Cycle and Ecosystems, Earth Surface and Interior, Atmospheric Composition, Weather, and Water and Energy Cycles. The CMR is intended to enable broader use of NASA's EOS data by providing a more uniform view of NASA's substantial and diverse data holdings. Its two primary objectives are to: 1) help science communities in need of data from multiple organizations and disciplines more efficiently use search functions and access data and services; and 2) increase the potential for interoperability with new tools and services. As the potential to exchange and inter-operate increases, the value of these resources increases comparably.

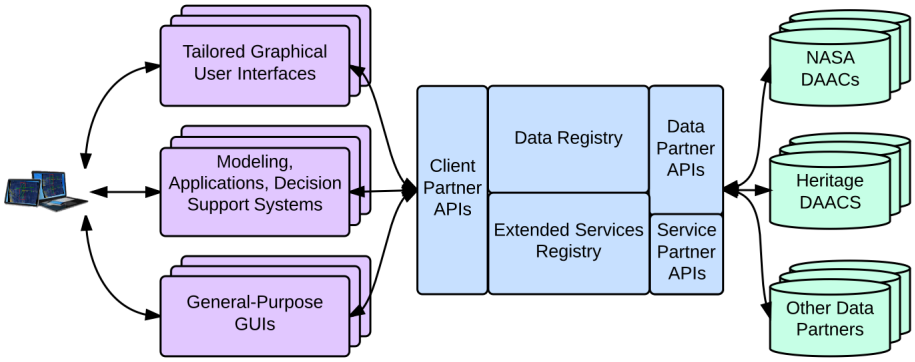
The CMR was designed to accomplish these goals by providing a system with an Application Programming Interface (API). While the API facilitates the discovery, online access, and delivery of a Data Partner's data holdings; it is the responsibility of the CMR Data Partners to add new metadata, remove old metadata, and modify and control access to existing metadata. As such, Data Partners retain complete control over what metadata are represented in the CMR at any given time. This guide is intended to serve as a reference for Data Partners who archive Earth Science Data in the CMR. Client Partners develop client applications that access the CMR API and take advantage of the services made available. These clients, such as Earthdata Search (https://search.earthdata.nasa.gov), CMR open search (https://cmr.earthdata.nasa.gov/opensearch), etc. allow end users to discover data which has been registered in the CMR's holdings; and can be custom made to meet the needs of a general user audience or a specific science application. For more complete information about client applications, refer to this User Guide's companion - the CMR Client Partner's Guide.

NASA's Earth science data has already proven essential to understanding Earth as an integrated system, and other organizations are also providing their Earth science metadata to the CMR for users to search and access. By simplifying discoverability and accessibility to the CMR's Earth Science holdings, and fostering interoperability with new tools and services, the user community will enlarge and the pace of scientific discovery and application will accelerate. For examples of how NASA's Earth science data is helping scientists understand the complexities of our Earth, visit Sensing our Planet and Other Featured Research Articles at https://earthdata.nasa.gov/.

The CMR concept and design

NASA's Earth Science Data and Information System (ESDIS) built the CMR based on Extensible Markup Language (XML), JavaScript Object Notation (JSON), and Web Service technologies. The CMR interfaces with different clients and users through its various APIs. The CMR is an open system with published APIs available to the CMR Development and User community.

Internally, the CMR specifies APIs and provides middleware components in a layered architecture - including data and service search and access functions. The figure below depicts the CMR system context in relation to its public APIs.



CMR System Concept

- Chapter 1: Introduction and Scope
  - The CMR concept and design
    - Security
    - CMR Capability And Functionality
    - CMR as a Spatially Enabled Metadata Search and Retrieval System
  - Data Partner Skills
  - Data Partner Tasks
  - CMR System Environments
- Chapter 2: The 1st Step - Getting Started
  - Meet the CMR Ops Team
    - CMR Provider Identification
    - Provider Administrators
- Chapter 3: The 2nd Step - Generating Compliant Metadata
  - Collections
    - DIF Specification
    - ECHO 10 Collection Specification
    - ISO 19115-2:2009 (MENDS)
    - ISO 19115-2:2009 (SMAP)
  - Granules
  - Services
  - Visualizations
  - Variables
  - Documents
- Chapter 4: The 3rd Step - Ingest
  - Ingest Overview
  - Ingest Components
    - Headers
      - Content-Type headers
      - Accept

Key features of the CMR architecture are:

- *Ease of Partner Participation* – Designed to be low-cost and minimally intrusive, the CMR offers a set of standard ways for Partners to interface with the system through provided web UIs and a metadata exchange approach that accommodates existing partners and technology.
- *Open System / Published APIs* – To accommodate independent CMR clients, CMR uses an open system approach and publishes domain APIs. These APIs are independent of the underlying transport protocols used. CMR communicates using WS-I Basic Profile v1.0 compliant web services for legacy services and RESTful web services for CMR ingest, search, and metadata management.
- *Evolutionary Development* – The CMR system is being developed incrementally to allow for insight and feedback during the development cycle. Industry trends are followed and the use of commercial, off-the-shelf (COTS) products is optimized.

Security

The CMR system requires Secure Sockets Layer (SSL)-based communication from Client Applications to the CMR API, but does not require secure communication from CMR to a Data Partner's data access fulfillment service. Internally, the CMR system is protected through a layer of software and hardware control mechanisms to preserve the integrity of CMR's holdings. When configuring data access fulfillment, Data Partners are strongly encouraged to utilize SSL communications.

CMR Capability And Functionality

CMR provides an infrastructure that allows various communities to share tools, services, and metadata. It supports many data access paradigms - such as navigation and discovery, facilitates data access through appropriate Data Partners, decentralizes end user functionality, and supports interoperability of distributed functions.

Although this Guide focuses on the needs of Data Partners, support is provided for all of the following nonexclusive Partner types:

- *Data Partners* – Organizations that supply metadata representing their data holdings to the CMR system
- *Client Partners* – Organizations that participate by developing software applications to access the Earth science metadata in the CMR system
- *Service Partners* – Organizations that participate by advertising their Earth science-related services to the user community via the CMR, which maintains service descriptions in a Service Catalog (either special services or services that are available as an option on a selected set of granules/collections) and supports the user in accessing those services.

The CMR allows Data Partners to cache copies of their metadata within it. Data Partners are responsible for adding new metadata, modifying existing data, and removing old data; and therefore have complete control over what metadata the CMR represents on their behalf. All CMR metadata is stored as received from the Data Partners, provided that the input meets the minimum validation criteria. From the user perspective, the CMR approach allows users to build their own user interfaces to the CMR, rather than being limited to the data search and access system provided by NASA. Furthermore, the CMR addresses science user needs through a set of well-defined and open interfaces upon which the user community can build its own client applications. In this way, the CMR supports extendable, flexible user interfaces, allowing industry and the science community to drive the progress of available earth science applications. For Data Partners, the system offloads the burden of providing the system resources required for searching and gives users the flexibility to support community-specific services and functionality. The CMR's interoperability features allow all participants to benefit from the distributed development of functions, again reducing dependence on NASA resources.

CMR as a Spatially Enabled Metadata Search and Retrieval System

The CMR allows Data Partners to define the spatial extent of a granule or a collection with different spatial constructs (for example: point and polygon). These spatial extents may be in either the Geodetic or Cartesian coordinate systems. Orbital data may also be provided to describe a collection or granule's spatial extents. A Client Partner can then construct a search using a point, a line, or a polygon (or multiple polygon) spatial type, and the CMR responds with data whose spatial region intersects the described region.
The CMR provides services for interacting with its Catalog of metadata. Queries can be performed in a number of ways, result contents can be specified, and the resulting data sets can be

Headers
- Responses
- Response Headers
  - cmr-request-id
  - HTTP Status Codes
  - Echo-Tokens
  - Examples
- CMR IDs
- CMR Environment URLs
- Validating, Ingesting, Updating, and Deleting Metadata Records
  - To Create a Token
  - Collection Validation Details
  - To Validate Collection Metadata
  - To Ingest Collection Metadata
  - To Delete Collecti

incrementally accessed so that large return sets can be handled gracefully. The system also supports constructing, submitting, and tracking access requests for the data that the metadata represents. The CMR supports both an embedding of a Uniform Resource Locator (URL) within the metadata for accessing the data (which the client simply accesses via Hypertext Transfer Protocol [HTTP] or File Transfer Protocol [FTP]), and a more complicated data access process in which other options are accommodated.

on

Data Partner Skills

Since the CMR uses platform-independent web service definitions for its API, there are no requirements for a client programming language. All examples in this document use curl; however, the code samples provided could be translated to any web service capable language.

CMR Data Partners need to be familiar with basic software development and Service Oriented Architecture (SOA) concepts such as:

- XML and XML Schema (XSD)
- JSON
- RESTful client and service communication programming
- Service-based Application Programming Interface (API)

Data Partner Tasks

CMR Data Partners who are beginning to integrate with CMR should be prepared to perform the following actions - each of which are detailed in later sections:

- Organize a consultation meeting with the CMR operations team to discuss the anticipated amount of data to be ingested; and setup the provider ID, user IDs, and permissions.
- Generate metadata compliant with a supported specification (DIF 9, DIF 10, SERF, ECHO10, ISO 19115-2 (MENDS or SMAP)).
- Ingest metadata into the CMR.
- Manage holdings within the CMR.
- Fulfill orders.

CMR System Environments

Three CMR systems are accessible by Data Providers: CMR PROD, CMR UAT, and CMR SIT. Each of these systems is briefly described below. For additional information, click on the associated link.

- **CMR OPS** (Operations) - The CMR OPS system environment is a publicly accessible server that houses the production environment. The Data Holdings within this system include Earth Science data that has been made available to the Earth Science Community by the CMR Data Partners. This environment is monitored 24/7, updated with enhancements and fixes on a monthly cycle, and experiences virtually no down time.
- **CMR UAT** (User Acceptance Test) - The UAT environment provides a stable test system to serve the needs of the CMR Data, Client, and Service Partners.The Data Holdings within this system consist of whatever the CMR's Data Partners have made available for their own testing purposes. Any enhancements and fixes that are planned for the Production Environment are installed in this environment two weeks prior to production delivery. CMR Partners are encouraged to verify the capabilities when a new release is installed. It is important to note that this environment (and the SIT environment) are designed for functional testing only. Performance and stress-testing of the CMR system is carried out in our own internal environment. If you wish to perform this type of testing on your clients to CMR then we suggest you direct your clients to the production environment for all 'read only' testing like searching. If you have an ingest client you wish to conduct performance testing with then contact the CMR Operations team. **The UAT and SIT environments are not provisioned for large amounts of data.**
- **CMR SIT** (System Integration Test) The SIT system was established in order to facilitate an exchange of ideas and provide an initial testing ground for upcoming capabilities. There is often very little metadata available in this test environment, but it is fully functional. The next operational version is released into this system approximately 1 month before its schedule Operational release date.

Chapter 2: The 1st Step - Getting Started

Meet the CMR Ops Team

Before ingest of a new provider's metadata can commence, the CMR team needs to have an understanding of expected metadata volume, characteristics, delivery mechanism, frequency, and relationship to the study of Earth science; as well as contact information and availability of the new provider. This information permits the CMR team to properly prepare the CMR for providing necessary services and hardware support to the new provider. Thus, the first step in becoming a Data Provider is to meet with the CMR Operations Team.

The CMR Operations Team can be contacted via email at support@earthdata.nasa.gov and will serve as the your primary contact. Additional online resources, such as the CMR Home page, located at https://earthdata.nasa.gov/cmr, and the CMR wiki page (Common Metadata Repository Home), are also available to assist you.

CMR Provider Identification

You will need to choose a unique name to serve as your data center/provider's identification in the CMR. This identifier will appear in the following locations within the system:

- CMR API
- CMR Metadata Item IDs
- Reverb Calendar
- Reverb Metadata Results
- CMR Holdings Report
- CMR Website
- Metadata Management Tool (MMT)
- Ingest Reports
- REST resource URLs specific to your provider - collection and granule resources for example

The identifier must have a short name that satisfies the following criteria: 10 characters or less; unique against all other providers; and capital letter as the first character followed by capital letters, numbers, or an underscore. The long name of the identifier can be much longer.

Please note that this name will become the basis of your provider's existence in the CMR. Changing this value after data has been ingested is possible, but strongly discouraged. Consider your choice carefully and feel free to request suggestions from the CMR Operations Team. You will be asked to disclose your data center/provider's identification to the CMR Operations Team at the CMR kick-off meeting.

Provider Administrators

The CMR Operations Team will request an initial list of individuals who are to be granted administrator access to your CMR data provider. Anyone granted administrative access must have an Earthdata Login (URS) account. Please be prepared to supply the Earthdata Login user IDs for the people on your list. If individuals you wish to have administrative access do not have an account, they may create one by visiting the Earthdata Login web page (https://urs.earthdata.nasa.gov (OPS), https://uat.urs.earthdata.nasa.gov (UAT), https://sit.urs.earthdata.nasa.gov (SIT)) for the mode in which your provider is being configured. Once users have administrator privileges, they can manage and give permissions to other users for the data provider through the Metadata Management Tool (MMT).

Metadata
- To Delete the Token
- Granule Validation Details
- To Validate Granule Metadata
- To Ingest Granule Metadata
- To Delete Granule Metadata

- Chapter 5: The 4th Step - Data Management
  - Access Control Concepts
    - Provider Objects vs. Catalog Items
    - Groups
    - Access Control List (ACL)
    - Permissions
    - Catalog Item Identifiers
      - Catalog Item Type
      - Collection Ide

ntifiers
- Metadata Filters
- Provider "Administrators" Group
- RestrictionFlag
- Access Control Recommendations
  - Data Mgmt & User Services Groups
  - Catalog Item ACLs
- Chapter 6: CMR Metadata - Past the Basics
  - Temporal Data (aka Acquisition Date and Time)
    - Collections
    - Granules
  - Additional Attributes
    - Collections
    - Granules
  - Platforms, Instruments And Sensors
    - Collections
    - Granules
  - Measured Parameters
  - Online Data Access URL And Online Resources URL
  - Keywords
- Chapter 7: Spatial Representations
  - Spatial overview
  - Collection & Granule Spatial Relationships
  - Geometry Representations
    - Coordinate Systems
      - Cartesian Coordinate System
      - Geodetic Coordinate System
  - Data Types and Representation
    - Geometry
    - Point

- Line
- Polygon
- Bounding Box
- Invalid Spatial Representations
  - Polygon Points in Counter-Clockwise Order
  - Twisted Polygon
  - Hole Crosses over Outer Ring
  - Polygon Crosses International Date Line (antimeridian) or Pole
  - Inappropriate Point Density
- Tolerance
- Orbit Data
- Global Data
- Tiling Identification System (Two-Dimensional Coordinate System)
  - Collection Level
  - Granule Level
- Acronyms

Chapter 3: The 2nd Step - Generating Compliant Metadata

Data is represented in the CMR through implementation of the Unified Metadata Model (UMM). The UMM is used by the CMR to drive search and retrieval of metadata cataloged within that system. The UMM encompasses various metadata profiles (or concepts) that consist of: Common (UMM-Common - *Elements common to multiple UMM Documents*), Collections (UMM-C), Granules (UMM-G), Services (UMM-S), Visualizations (UMM-Viz), Variables (UMM-Var), and Documents (UMM-D). While the UMM profiles are not metadata standards, they are agreed upon parameters from which a data provider can choose to represent their metadata. Listed below are the UMM profiles – each with a description detailing profile definition, any additional information about the profile, and the supported metadata standards and specifications for that profile.

- **Collection**
  - A collection is a grouping of science data that all come from the same source, such as a modeling group or institution. Collections are also referred to as datasets; and in ISO 19115, as scope type “series."
  - A collection may contain zero or more granules and has common information across all the granules they contain. Collections also contain a template for describing additional attributes not already part of the metadata model.
  - Supported standards include: DIF 9, DIF 10, ECHO 10, ISO 19115-2:2009 (MENDS AND SMAP dialects), and ISO 19115-1:2013

- **Granule**
  - A granules is the smallest aggregation of data that can be independently managed (described, inventoried, and retrieved). Granules are also referred to as file level metadata; and in ISO 19115, as scope type “dataset.”
  - Granules cannot exist without being associated to a collection. Granules have their own metadata model and support values associated with the additional attributes defined by the parent collection.
  - Supported standards include: ECHO 10, ISO 19115-2:2009 (SMAP dialect), and ISO 19115-1:2013

- **Service**
  - Describes a human or machine item that can be engaged to produce a product or data.
  - Supported standards will include SERF, ISO 19115-2:2009 (MENDS dialect), and ISO 19115-1:2013
  - Not yet supported

- **Visualization**
  - A grouping of parameter data that can be visualized.
  - Not yet supported

- **Variable**
  - An aggregation of data from the granule that specifies a certain variable at a specific time.
  - Not yet supported

- **Document**
  - Descriptions of documents.
  - Not yet supported

The first step in generating compliant metadata is selecting the standard or specification that will be used to represent your data from the options in the supplied profiles. The documentation for each specification and their associated schemas are listed below.

Standard Name	Documentation	Schema
DIF 9	http://gcmd.gsfc.nasa.gov/add/difguide/index.html	https://gcmd.gsfc.nasa.gov/Aboutus/xml/dif/dif_v9.9.3.xsd https://git.earthdata.nasa.gov/projects/EMFD/repos/dif-schema
DIF 10		https://gcmd.gsfc.nasa.gov/Aboutus/xml/dif/dif_v10.2.xsd https://git.earthdata.nasa.gov/projects/EMFD/repos/dif-schema https://git.earthdata.nasa.gov/projects/EMFD/repos/dif-schema
SERF	http://gcmd.gsfc.nasa.gov/add/serfguide/index.html	http://gcmd.gsfc.nasa.gov/Aboutus/xml/serf/serf_v9.9.3.xsd https://git.earthdata.nasa.gov/projects/EMFD/repos/serf-scher
ECHO 10 Collection	https://cmr.earthdata.nasa.gov/ingest/site/docs/ingest/api.html	https://git.earthdata.nasa.gov/projects/EMFD/repos/echo-sche https://git.earthdata.nasa.gov/projects/EMFD/repos/echo-sche
ECHO 10 Granule	https://cmr.earthdata.nasa.gov/ingest/site/docs/ingest/api.html	https://git.earthdata.nasa.gov/projects/EMFD/repos/echo-sche
ISO 19115-2:2009 (MENDS)	ISO 19115-2 https://committee.iso.org/home/tc211	https://cdn.earthdata.nasa.gov/iso/ https://git.earthdata.nasa.gov/projects/EMFD/repos/iso-schem
ISO 19115-2:2009 (SMAP)	NASA ISO for EOSDIS https://committee.iso.org/home/tc211	
ISO 19115-1:2013	https://committee.iso.org/home/tc211 https://github.com/ISO-TC211/UML-Best-Practices	https://github.com/ISO-TC211/XML

Once the standard or specification has been chosen, you can begin creating the metadata based on the desired profile.

Collections

The following table depicts the various specifications supported by the collections profile (X-axis) as well as all the elements in the UMM-C (Y-axis). If the specification supports the UMM-C element, the cell is colored light green and includes the letter "Y". If the specification does not support the UMM-C element, the cell is colored pink and includes the letter "N". A few cells are colored yellow with the word TBD, as these mappings are still in progress. Elements that are required are specified by the presence of the word “YES” in the Required column next to the element name.

UMM-C Element	Required	DIF 9	DIF 10.x	ECHO 10	ISO 19115-2	ISO 19115-1	EMS
METADATA INFORMATION							
Metadata Language		N	N	N	Y	Y	N
Metadata Standard	YES	Y	Y	Y	Y	Y	N
Metadata Date	YES	Y	Y	Y	Y	Y	N
DATA IDENTIFICATION							
Entry ID	YES	Y	Y	Y	Y	Y	Y
Entry Title	YES	Y	Y	Y	Y	Y	Y
Abstract	YES	Y	Y	Y	Y	Y	N
Purpose		Y	Y	Y	Y	Y	N
Data Language		Y	Y	N	Y	Y	N
Data Date	YES	N	Y	Y	Y	Y	N
Organization	YES	Y	Y	Y	Y	Y	Y
Personnel		Y	Y	Y	Y	Y	N
Collection Data Type		Y	Y	Y	TBD	TBD	N
Processing Level	YES	Y	Y	Y	Y	Y	Y
Resource Citation		Y	Y	Y	Y	Y	N
Collection Progress		Y	Y	Y	Y	Y	N
Quality		Y	Y	N	Y	Y	N
Use Constraints		Y	Y	N	Y	Y	N
Access Constraints		Y	Y	Y	Y	Y	N
Metadata Association		Y	Y	Y	Y	Y	N
Publication Reference		Y	Y	N	Y	Y	N
DESCRIPTIVE KEYWORDS							
ISO Topic Category		Y	Y	N	Y	Y	N
Science Keywords	YES	Y	Y	Y	Y	Y	N
Ancillary Keywords		Y	Y	N	Y	Y	N
Additional Attributes		Y	Y	Y	Y	Y	N
DISTRIBUTION INFORMATION							
Related URL		Y	Y	Y	Y	Y	N
Distribution		Y	Y	Y	Y	Y	N
TEMPORAL INFORMATION							
Temporal Extent	YES	Y	Y	Y	Y	Y	N
Temporal Keywords		Y	Y	Y	Y	Y	N
Paleo Temporal Coverage		Y	Y	N	TBD	TBD	N
SPATIAL INFORMATION							
Spatial Extent	YES	Y	Y	Y	Y	Y	N
Tiling Identification System (Formerly TwoDCoordinateSystem)		N	Y	Y	Y	Y	N
Spatial Representation Information		Y	Y	Y	Y	Y	N
Spatial Keywords		Y	Y	Y	Y	Y	N
ACQUISITION INFORMATION							

Platform	YES	Y	Y	Y	Y	Y	Y
Instrument	YES	Y	Y	Y	Y	Y	Y
Sensor		N	Y	Y	Y	Y	Y
Project	YES	Y	Y	Y	Y	Y	Y

DIF Specification

The Directory Interchange Format has two versions, DIF 9 and DIF 10. While DIF 9 is still supported, it is highly recommended that Data Providers use the new DIF 10 format - as it includes many new features. Mapping between the different standards and UMM-C is documented in the UMM-C and UMM-Common **word documents** located here. For convenience the following table lists the top level DIF 9 and DIF 10 elements as required by the UMM.

DIF 9	DIF 10	UMM
/DIF/DIF_Creation_Date /DIF/Last_DIF_Revision_Date	/DIF/Metadata_Dates/Metadata_Creation /DIF/Metadata_Dates/Metadata_Last_Revision	MetadataDate/Dat MetadataDate/Typ creation, lastRevis
/DIF/Entry_ID	/DIF/Entry_ID	EntryId/Id
/DIF/Entry_Title	/DIF/Entry_Title	EntryTitle
/DIF/Summary/Abstract	/DIF/Summary/Abstract	Abstract
N/A	/DIF/Metadata_Dates/Data_Creation /DIF/Metadata_Dates/Data_Last_Revision	DataDate
/DIF/Data_Center	/DIF/Organization	Organization
/DIF/Extended_Metadata.Metadata.Name='ProcessingLevelId' /DIF/Extended_Metadata.Metadata.Name='ProcessingLevelDescription'	/DIF/ProcessingLevelId /DIF/ProcessingLevelDescription	ProcessingLevel/ID ProcessingLevel/D
/DIF/Parameters/	/DIF/ScienceKeywords/	ScienceKeyword
/DIF/Related_URL /DIF/Multimedia_Sample	/DIF/Related_URL /DIF/Multimedia_Sample	RelatedUrl
/DIF/Temporal_Coverage	{Choice of one} 1) /DIF/Temporal_Coverage/Range_DateTime/ 2) /DIF/Temporal_Coverage/Single_DateTime/ 3) /DIF/Temporal_Coverage/Periodic_DateTime/	TemporalExtent
/DIF/Spatial_Coverage/ or /DIF/Extended_Metadata/ for more complicated geometries	/DIF/Spatial_Coverage/	SpatialExtent
/DIF/Source_Name	/DIF/Platform/	Platform
/DIF/Sensor_Name/	/DIF/Platform/Instrument/	Instrument
/DIF/Project/	/DIF/Project/	Project

The correct way to fill out a DIF 9 or DIF 10 collection record can be viewed by clicking on the sample file links below. This will initiate the download of an XML file, that once opened, will provide an example.

DIF 9 sample file

DIF 10 sample file

ECHO 10 Collection Specification

The Earth Observing System (EOS) Clearing House system uses a standard of the same name - ECHO 10. This standard is split into two parts, one for collections, and the other for granules. This section focuses on the collection specification, as the granule specification will be discussed later in the guide. The mapping between the different standards and UMM-C is documented in the UMM-C and UMM-Common **word documents** located here. For convenience the following table lists the top level ECHO 10 elements as required by the UMM.

ECHO 10	UMM
/Collection/RevisionDate	MetadataDate/Date with MetadataDate/Type: creation, lastRevision
/Collection/ShortName with _ AND /Collection/VersionId	EntryId/Id EntryId/Version
/Collection/LongName	EntryTitle
/Collection/Description	Abstract
/Collection/InsertTime /Collection/LastUpdate /Collection/DeleteTime	DataDate with DataDate/Type creation, lastUpdate, delete
/Collection/ProcessingCenter /Collection/ArchiveCenter	Organization
/Collection/ProcessingLevelId /Collection/ProcessingLevelDescription	ProcessingLevel/ID ProcessingLevel/Description
/Collection/ScienceKeywords/ScienceKeyword/	ScienceKeyword
/Collection/OnlineAccessURLs/OnlineAccessURL /Collection/OnlineResources/OnlineResource /Collection/AssociatedBrowseImages /Collection/AssociatedBrowseImageUrls	RelatedUrl
/Collection/Temporal/	TemporalExtent
/Collection/Spatial/	SpatialExtent
/Collection/Platforms/Platform/	Platform
/Collection/Platforms/Platform/Instruments/Instrument/	Instrument
/Collection/Campaigns/Campaign/	Project

The correct way to fill out an ECHO 10 collection record can be viewed by clicking on the ECHO 10 sample file link below. This will initiate the download of an XML file, that once opened, will provide an example.

ECHO 10 sample file

ISO 19115-2:2009 (MENDS)

The ISO 19115-2:2009 MENDS implementation of the ISO 19139 model is supported by the CMR. The mapping between the different standards and UMM-C is documented in the UMM-C and UMM-Common **word documents** located here. For convenience the following table lists the top level ISO 19115-2:2009 elements as required by the UMM.

ISO 19115-2:2009 (MENDS)	UMM
/gmi:MI_Metadata/datestamp – original creation date or revision date or /gmi:MI_Metadata/gmd:citation/gmd:CI_Citation/gmd:date/gmd:CI_Date/gmd:date/gco:DateTime with /gmi:MI_Metadata/gmd:citation/gmd:CI_Citation/gmd:date/gmd:CI_Date/gmd:dateType/gmd:CI_DateTypeCode @codeListValue=	MetadataDate/Da MetadataDate/Ty creation, lastRevi
ID: /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:citation/gmd:CI_Citation/gmd:identifier/gmd:MD_Identifier/gmd:code/gco:CharacterString Version: /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:citation/gmd:CI_Citation/gmd:identifier/gmd:MD_Identifier/gmd:version/gco:CharacterString	EntryId/Id EntryId/Version
/gmi:MI_Metadata/gmd:fileIdentifier/gco:CharacterString and /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:citation/gmd:CI_Citation/gmd:title/gco:CharacterString	EntryTitle
/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification/gmd:abstract/gco:CharacterString	Abstract
/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:citation/gmd:CI_Citation/gmd:date/gmd:CI_Date/gmd:date/gco:DateTime with /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:citation/gmd:CI_Citation/gmd:date/gmd:CI_Date/gmd:dateType/gmd:CI_DateTypeCode codeListValue varies.	DataDate with DataDate/Type cr lastUpdate, delete
gmd:CI_ResponsibleParty	Organization

/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:processingLevel/gmd:MD_Identifier/gmd:code/gco:CharacterString /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:processingLevel/gmd:MD_Identifier/gmd:description/gco:CharacterString and /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:contentInfo/gmd:MD_ImageDescription/gmd:processingLevelCode/gmd:MD_Identifier/gmd:code/gco:CharacterString /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:contentInfo/gmd:MD_ImageDescription/gmd:processingLevelCode/gmd:MD_Identifier /gmd:description/gco:CharacterString	ProcessingLevel/I ProcessingLevel/I
/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:descriptiveKeywords/gmd:MD_Keywords/gmd:keyword/gco:CharacterString with /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:descriptiveKeywords/gmd:MD_Keywords/gmd:keyword/gmd:type /MD_KeywordTypeCode[@codeListValue="theme"]	ScienceKeyword
	RelatedUrl
/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:extent/gmd:EX_Extent/gmd:temporalElement/gmd:EX_TemporalExtent	TemporalExtent
/gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:extent/gmd:EX_Extent/gmd:description/gco:CharacterString with format: VerticalSpatialDomainType= {type}, VerticalSpatialDomainValue= {value} /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification /gmd:extent/gmd:EX_Extent/gmd:geographicElement	SpatialExtent
/gmi:MI_Metadata/gmi:acquisitionInformation/gmi:MI_AcquisitionInformation/gmi:platform/eos:EOS_Platform/ and /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification/gmd:descriptiveKeywords /gmd:MD_Keywords/gmd:keyword/gco:CharacterString with /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification/gmd:descriptiveKeywords /gmd:MD_Keywords/gmd:keyword/gmd:type/MD_KeywordTypeCode[@codeListValue="platform"]	Platform
/gmi:MI_Metadata/gmi:acquisitionInformation/gmi:MI_AcquisitionInformation /gmi:instrument/eos:EOS_Instrument/ and /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification/gmd:descriptiveKeywords /gmd:MD_Keywords/gmd:keyword/gco:CharacterString with /gmi:MI_Metadata/gmd:identificationInfo/gmd:MD_DataIdentification/gmd:descriptiveKeywords /gmd:MD_Keywords/gmd:keyword/gmd:type/MD_KeywordTypeCode[@codeListValue="instrument"]	Instrument

/gmi:MI_Metadata/gmi:acquisitionInformation/gmi:MI_AcquisitionInformation/gmi:operation/gmi:MI_Operation/	Project

The correct way to fill out an ISO 19115-2 MENDS collection record can be viewed by clicking on the 19115-2 MENDS sample file link below. This will initiate the download of an XML file, that once opened, will provide an example.

ISO 19115-2 MENDS sample file

ISO 19115-2:2009 (SMAP)

The ISO 19115-2:2009 SMAP implementation of the ISO 19139 model is supported by the CMR.

The correct way to fill out an ISO 19115-2 SMAP collection record can be viewed by clicking on the 19115-2 SMAP sample file link below. This will initiate the download of an XML file, that once opened, will provide an example.

ISO 19115-2 SMAP sample file (will be available when the data becomes public)

Special Considerations

The following additional items should be considered when generating your metadata for submission to the CMR:

- Additional Attributes
- Science Keywords
- GCMD Dif Entry IDs
- Platform/Instrument/Sensor
- Collection Spatial Representation
- Granule Spatial Representation

For further information, refer to the associated sections in "Chapter 6: CMR Metadata Extra Information" and in "Chapter 7: Spatial Extent" of this guide.

Granules

The following table depicts the various specifications supported by the granules profile (X-axis) as well as all the elements in the UMM-G (Y-axis). If the specification supports the UMM-G element, the cell is colored light green and includes the letter "Y". If the specification does not support the UMM-C element, the cell is colored pink and includes the letter "N". A few cells are colored yellow with the word TBD, as these mappings are still in progress. Elements that are required are specified by the presence of the word “YES” in the Required column next to the element name.

UMM-G Field	Required	Full Documentation	ECHO 10	ISO 19115-2	ISO 19115-1
METADATA INFORMATION					

Metadata Language		UMM Common	N	Y	Y
Metadata Standard		UMM Common	Y	Y	Y
Metadata Dates		UMM Common	Y	Y	Y
DATA IDENTIFICATION					
Universal Reference Identifier	YES	UMM Common			
Data Language		UMM Common	N	Y	Y
Data Dates	YES	UMM Common	Y	Y	Y
Producer Identifier		UMM-G	Y	Y	Y
Local Version Identifier		UMM-G	Y	Y	Y
Browse Graphic		UMM Common	Y	Y	Y
Data Format		UMM -G	Y	Y	Y
Access Constraint Value		UMM Common	Y	Y	Y
Parent Collection Identifier	YES	UMM-G	Y	Y	Y
DESCRIPTIVE KEYWORDS					
Spatial Locality		UMM-G	Y	Y	Y
SPATIAL INFORMATION					
Spatial Extent		UMM Common	Y	Y	Y
Tiling Identification System (Formerly TwoDCoordinateSystem)		UMM Common	Y	Y	Y
TEMPORAL INFORMATION					
Temporal Extent		UMM Common	Y	Y	Y
COVERAGE INFORMATION					
Additional Attributes - Content Information		UMM Common	Y	Y	Y
Day Night Flag		UMM-G	Y	Y	Y
Cloud Cover		UMM-G	Y	Y	Y
DISTRIBUTION INFORMATION					
Related URL		UMM Common	Y	Y	Y
Distribution		UMM Common	Y	Y	Y
ACQUISITION INFORMATION					
Platform		UMM Common	Y	Y	Y
Instrument		UMM Common	Y	Y	Y
Sensor		UMM Common	Y	Y	Y
Project		UMM Common	Y	Y	Y
Platform Orbit Information		UMM Common	Y	Y	Y
Additional Attributes - Platform Information		UMM Common	Y	Y	Y
DATA QUALITY INFORMATION					
Additional Attributes - Quality Information		UMM Common	Y	Y	Y
Reprocessing Lineage		UMM-G	Y	Y	Y
Production Date Time		UMM-G	Y	Y	Y
Product Generation Executive (PGE) Information		UMM-G	Y	Y	Y
Input Granules		UMM-G	Y	Y	Y

ECHO 10 Granules

The Earth Observing System (EOS) Clearing House system uses a standard of the same name - ECHO 10. This standard is split into two parts, one for collections, and the other for granules. This section focuses on the granule specification, as the collection specification was discussed earlier in the guide. The mapping between the different standards and UMM-G is documented in the UMM-G and UMM-Common **word documents** located here. For convenience the following table lists the top level ECHO 10 elements as required by the UMM.

ECHO 10	UMM
/Granule/DataGranule/ProductionDateTime	MetadataDate/Date with MetadataDate/Type: creation, lastRevision
/Granule/GranuleUR /Granule/DataGranule/LocalVersionId	EntryId/Id EntryId/Version
/Granule/GranuleUR	EntryTitle
/Granule/InsertTime /Granule/LastUpdate /Granule/DeleteTime	DataDate with DataDate/Type creation, lastUpdate, delete
/Granule/Collection/ShortName with /Granule/Collection/VersionId or /Granule/Collection/EntryId or /Granule/Collection/DataSetId	Parent Collection Identifier

The correct way to fill out an ECHO 10 Granule record can be viewed by clicking on the ECHO 10 sample file link below. This will initiate the download of an XML file, that once opened, will provide an example.

ECHO 10 sample file

Services

Since services are not yet supported, this section is where future documentation will exist to support writing compliant metadata.

Visualizations

Since visualizations are not yet supported, this section is where future documentation will exist to support writing compliant metadata.

Variables

Since variables are not yet supported, this section is where future documentation will exist to support writing compliant metadata.

Documents

Since documents are not yet supported, this section is where future documentation will exist to support writing compliant metadata.

Chapter 4: The 3rd Step - Ingest

Ingest Overview

Ingest refers to the process of validating, inserting, updating, or deleting metadata in the CMR system and affects only the metadata for the specific Data Partner. The CMR allows Data Partners to ingest metadata records through a RESTful API. This chapter is designed to concisely convey information necessary to successfully ingest metadata in CMR. Additional ingest reference material is located at the following URL: https://cmr.sit.earthdata.nasa.gov/ingest/site/ingest_api_docs.html

Ingest Components

Before discussing the steps required to complete ingest, it's important to understand some important compents that contribute to this process.

Headers

Headers are a part of HTTP requests and for the CMR they provide information such as message content (content type), the format of the data that gets returned (accept), and tokens to allow increased privileges (Echo-Token), etc.

Content-Type headers

Content-Type is a standard HTTP header that specifies the content type of the body of the request for POST method messages. Ingest supports the following content types for ingesting metadata:

Standard	Content-Type
DIF 10	application/dif10+xml
DIF 9	application/dif+xml
ECHO 10	application/echo10+xml
ISO 19115 (MENDS)	application/iso19115+xml
ISO 19115 SMAP	application/iso:smap+xml

Accept Headers

Accept hearders are used in cases where the caller wishes to control whether the data gets returned in XML or JSON format. The following

table lists the valid values. If this header is not used, XML results will be returned by default.

Type Received	Accept HeaderValue
XML	application/xml
JSON	application/json

Responses

Response Headers

cmr-request-id

This header returns the unique id assigned to the request. This can be used to help debug client errors. The value is a long string of the form

`828ef0b8-a876-4579-85db-3cc9d1b5f6e5`

HTTP Status Codes

Status Code	Description
200	Successful update/delete
201	Successful create
400	Bad request. The body will contain errors.
404	Not found. This could be returned either because the URL isn't known by ingest or the item wasn't found.
409	Conflict. This is returned when a revision id conflict occurred while saving the item.
415	Unsupported Media Type. The body will return an error message that contains the list of supported ingest formats.
422	Unprocessable entity. Ingest understood the request, but the concept failed ingest validation rules. The body will contain errors.
500	Internal error. Contact CMR Operations if this occurs.
503	Internal error because a service dependency is not available.

Echo-Tokens

The Echo-Token allows the CMR to know who is making a request. The Token is in the format of XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX. A token must first be generated as described in the next section. Once the requester has the token, the token can be placed into the http header for the necessary API calls.

Examples

Below are some examples concerning how to use headers. The purple part of the example will be explained in this section, while the rest will be addressed later in the document.

Example 1
The following curl command requests that a metadata record called "something1" be validated. In this request the Content-Type, which is the specification and format of something1.xml file being validated, is using the echo10 specification with the XML format. The accept header states that the results are requested in the XML format. The Echo-Token header allows the CMR to know who is making the request for authorization purposes.

curl -v -XPOST -H "Content-Type: application/echo10+xml" -H "Accept: application/xml" -H "Echo-Token: `75E5CEBE-6BBB-2FB5-A613-0368A361D0B6`" -d @../../Downloads/records/something1.xml https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider ID>/validate/collection/something1

Example 2
The following curl command requests that a metadata record called "something1" be validated. In this request the Content-Type, which is the specification and format of something1.xml file being validated, is using the ISO 19115 SMAP specification with the XML format. The accept header states that the results are being requested in the JSON format. The Echo-Token header allows the CMR to know who is making the request for authorization purposes.

curl -v -XPOST -H "Content-Type: application/iso:smap+xml" -H "Accept: application/json" -H "Echo-Token: `75E5CEBE-6BBB-2FB5-A613-0368A361D0B6`" -d @../../Downloads/records/something1.xml https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider ID>/validate/collection/something1

CMR IDs

Below is a selection of IDs that are used in the CMR API. Each ID is defined and will be referred to in subsequent examples.

Provider ID - A provider ID identifies a provider and is composed of a combination of upper case letters, digits, and underscores.

Example
LPDAAC_ECS

Native ID - A native ID is used by the provider client to refer to a granule or collection in the URL. The native ID must be unique within a provider - i.e., two collections cannot share a native ID. While the native ID does not have to match an ID in the metadata, providers are encouraged to use something like the entry ID or entry title as their native ID.

Example
cloud5

CMR Concept ID - A profile or concept is any type of metadata that is managed by the CMR. Currently, the CMR manages two concept types - Collection and Granule. Others will be added as the CMR matures. The concept ID is the unique identifier of concepts in the CMR. The format of the concept ID is: `<letter> <unique-number> "-" <provider-id>`

Example
Ex: C179460405-LPDAAC_ECS.
The letter identifies the concept type - C for collection and G for granule. The uppercase letters following the #'s is the provider ID (see the first ID in this section).

CMR Revision ID - Each individual modification (update or deletion) of a profile or concept is archived as a separate revision in the CMR. This permits the CMR to improve caching, synchronization, and to maintain an audit log of changes to concepts. Every revision is assigned a separate numerical ID starting with the number 1.

Revision ID Example
The enclosed table shows one collection with 4 revisions.

CMR Revision ID 1: Collection was ingested

CMR Revision ID 2: Collection was updated

CMR Revision ID 3: Collection was deleted

CMR Revision ID 4: Collection was re-ingested
**Note: Any Revision ID that records a deletion is called a "tombstone."*

Concept Id	CMR Revision Id	Metadata	Deleted

C1-PROV1	1	...	false
C1-PROV1	2	...	false
C1-PROV1	3	null	true
C1-PROV1	4	...	false

CMR Environment URLs

Note: All of the examples provided below are using the Systems Integration Test environment (SIT). To run the commands in the other environments see the table below.

CMR Environment	Base API URL
Production (PROD)	https://cmr.earthdata.nasa.gov/
User Acceptance Test (UAT)	https://cmr.uat.earthdata.nasa.gov/
Systems Integration Test (SIT)	https://cmr.sit.earthdata.nasa.gov/

Validating, Ingesting, Updating, and Deleting Metadata Records

To validate, ingest, update, or delete metadata records:

1. Create a token
2. Perform one or more of the following in any order:
    - Validate records
    - Ingest records to insert or update them.
    - Delete records
3. Delete the token

Tokens: Tokens are used by the CMR to validate both the requester and their priviledges for each request message (i.e., the http call to CMR) submitted. For most searches, a token is not needed because the metadata records are unrestricted and accessible by anyone. However, when specific metadata records are restricted, privileged users require a token to see and access those records. Data providers also require tokens to validate, ingest, update, or delete metadata records.

To Create a Token

1. Select the environment you will be working in from the CMR environments table below.

CMR Environments Table

CMR Environment	Base API URL	Associated Earthdata Login (URS) Environment
Operational (OPS)	https://cmr.earthdata.nasa.gov	https://urs.earthdata.nasa.gov
User Acceptance Test (UAT)	https://cmr.uat.earthdata.nasa.gov	https://uat.urs.earthdata.nasa.gov
Systems Integration Test (SIT)	https://cmr.sit.earthdata.nasa.gov	https://sit.urs.earthdata.nasa.gov

2. On a terminal window execute the curl command for the environment you selected.

Example

```
curl -X POST --header "Content-Type: application/xml" -d "<token><username>sample_username</user
name><password>sample-password</password><client_id>client_name_of_your_choosing</client_id><use
r_ip_address>your_origin_ip_address</user_ip_address> </token>" https://cmr.earthdata.nasa.gov/le
gacy-services/rest/tokens
```

Note:

- *Depending on the environment you selected, the Base API URL may be different from the example. If so, replace the purple text with the correct Base API URL.*
- *If you are embedding the token REST messages into a programming language, create an HTTP message and place the same components from the curl example into either the message header or body.*
- *If you have special characters in your password, you will probably need to escape them using a backslash.*

If you don't want to escape any characters, but still want to use curl - implement the "file input" option to create a file that looks like the following:

Example

```
<token>
<username>sample_username</username>
  <password>sample-password</password>
  <client_id>client_name_of_your_choosing</client_id>
  <user_ip_address>your_origin_ip_address</user_ip_address>
</token>
```

Note: mytokengenerator.xml can be used as a file name, which simplifies the command. See example underneath step 2 for original command and the example below for simplified command.

Example

```
curl -X POST --header "Content-Type: application/xml" -d @mytokengenerator.xml https://cmr.uat.earthdata.nasa.gov/legacy-services/rest/tokens
```

Note: if using a programming language, just place the curl example parts into the correct http message header or body locations.

Provided a successful response is received, an HTTP success status code of 200 is supplied with the response. Below is a sample response from the curl call - where the value in the ID tag is the token you will use as the value in the Echo-Token header:

<?xmlversion="1.0"encoding="UTF-8"?>

<token>

<id>75E5CEBE-6BBB-2FB5-A613-0368A361D0B6</id>

<username>sample_username</username>

<client_id>client_name_of_your_choosing</client_id>

<user_ip_address>your_origin_ip_address</user_ip_address>

</token>

Collection Validation Details

The following validations are performed when we validate or ingest collection metadata

1. Request validation: Validates format specified and metadata length of the request. Returns errors when fail.
2. XML validation: Validates against the native metadata schema (i.e. DIF, ECHO10, UMM-JSON). Returns errors when fail.
3. UMM validation: Returns warnings as default based on the settings of the ingest configs: return-umm-son-validation-errors and return-umm-spec-validation-errors, which are currently set to false.
   Can return errors by either setting the configs to true or by setting the CMR-Validate-UMM-C header to true.
   a. Validates according to UMM-C and UMM-Common schema defined in https://git.earthdata.nasa.gov/projects/EMFD/repos/unified-metadata-model/browse;
   b. Validation checks field values are in the expected format, within expected range and conform to controlled vocabularies where present.
4. Business rule validations: Returns errors when fail.
   a. Delete time in collection metadata is after the current time.
   b. version is provided.

c. Concept is is correct if provided by user.
d. Collection additional attribute changes do not invalidate existing granules.
e. Collection projects(campaigns) changes do not invalidate existing granules.
f. Collection temporal changes do not invalidate existing granules.
g. Collection spatial changes do not invalidate existing granules..

To Validate Collection Metadata

Collection metadata can be validated without being ingested by sending an HTTP POST with the metadata to the URL: https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider Id>/validate/collection/<Native Id>. Please refer to HTTP Status Codes under Response Headers for status information.

The following example demonstrates how to do a validation using curl:

Example

```
curl -i -XPOST -H "Content-Type: application/echo10+xml" -H "Echo-Token:
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provi
der Id>/validate/collection/<Native Id> -d \

"<Collection>
  <ShortName>ShortName_Larc</ShortName>
  <VersionId>Version01</VersionId>
  <InsertTime>1999-12-31T19:00:00-05:00</InsertTime>
  <LastUpdate>1999-12-31T19:00:00-05:00</LastUpdate>
  <DeleteTime>2015-05-23T22:30:59</DeleteTime>
  <LongName>LarcLongName</LongName>
  <DataSetId>LarcDatasetId</DataSetId>
  <Description>A minimal valid collection</Description>
  <Orderable>true</Orderable>
  <Visible>true</Visible>
</Collection>"
```

The above example uses the SIT environment to validate a metadata record. If the Provider ID is CMR_ONLY and the Native ID is cloud5, the above example would appear as follows:

Example

```
curl -i -XPOST -H "Content-Type: application/echo10+xml" -H "Echo-Token:
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" https://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ON
LY/validate/collection/cloud5 -d \

"<Collection>
  <ShortName>CLOUD5</ShortName>
  <VersionId>Version01</VersionId>
  <InsertTime>1999-12-31T19:00:00-05:00</InsertTime>
  <LastUpdate>1999-12-31T19:00:00-05:00</LastUpdate>
  <DeleteTime>2015-05-23T22:30:59</DeleteTime>
  <LongName>Cloud 5</LongName>
  <DataSetId>Cloud 5</DataSetId>
  <Description>A minimal valid collection</Description>
  <Orderable>true</Orderable>
  <Visible>true</Visible>
</Collection>"
```

Note: The -d is the contents of the POST message, i.e., the metadata record.

The following example is identical to the one above, accepting that in this scenario, curl is reading the contents of a file instead of listing out the full record. Thus, in this case, the collection metadata is located in the Native file.

Example

```
curl -i -XPOST -H "Content-Type: application/echo10+xml" -H "Echo-Token:
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" -d @NativeId.xml

https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider Id>/validate/collection/<Native Id>
```

The response to a validation is either:

1. a success response code of 200 with the body that contains any warnings, or no body if no warnings are returned. Below is an example of a successful validation with warnings:

Successful validation with warnings - XML

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
    <warnings>After translating item to UMM-C the metadata had the following issue: object has missing required properties ([“DataCenters”,“Platforms”,“ProcessingLevel”,“RelatedUrls”,“ScienceKeywords”,“SpatialExtent”])</warnings>
</result>
```

2. a failure response code of 400 with the body that contains the errors. Below is an example error response body in both XML and JSON:

XML

<?xml version="1.0" encoding="UTF-8"?><errors><error>Line 1 - cvc-datatype-valid.1.2.1: '2015-07-10-10-10T00:00:00Z' is not a valid value for 'dateTime'.</error><error>Line 1 - cvc-type.3.1.3: The value '2015-07-10-10-10T00:00:00Z' of element 'InsertTime' is not valid.</error><error>Line 1 - cvc-complex-type.2.4.a: Invalid content was found starting with element 'WestBoundingCoordinate'. One of '{NorthBoundingCoordinate}' is expected.</error></errors>

JSON

{"errors":["Line 1 - cvc-datatype-valid.1.2.1: '2015-07-10-10-10T00:00:00Z' is not a valid value for 'dateTime'.","Line 1 - cvc-type.3.1.3: The value '2015-07-10-10-10T00:00:00Z' of element 'InsertTime' is not valid.","Line 1 - cvc-complex-type.2.4.a: Invalid content was found starting with element 'WestBoundingCoordinate'. One of '{NorthBoundingCoordinate}' is expected."]

To Ingest Collection Metadata

Collection metadata can be ingested (create or update) by sending an HTTP PUT with the metadata to the URL https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider Id>/collections/<Native Id>. Please refer to HTTP Status Codes under Response Headers for status information.

See the example below, where CMR_ONLY is the Provider ID and cloud5 is the Native ID.

Example

```
curl -i -XPUT -H "Content-Type: application/echo10+xml" -H "Echo-Token: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" -H "Accept: application/xml" -d @cloud5.xml

https://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/collections/cloud5
```

If ingest is successful, the response code is 201 for create and 200 for update/delete and the response body will include the Concept ID and the Revision ID as shown below. Any warnings from UMM validation will be returned as well. When the Cmr-Validate-Umm-C request header is set to true, any warnings will be returned as errors and ingest will fail. The return format will be either XML or JSON depending on what, if anything, was specified by the accept header.

XML

<?xml version=\"1.0\" encoding=\"UTF-8\"?><result><concept-id>C1200018690-CMR_ONLY</concept-id><revision-id>1</revision-id></result>

JSON

```
{\"concept-id\":\"C1200018690-CMR_ONLY\",\"revision-id\":1}
```

if ingest is unsuccessful, the error messages will be returned to the requester in the body with a response code of 400 (see example below).

Unsuccesful Ingest

```
<errors>
   <error>Parent collection for granule [SC:AE_5DSno.002:30500511] does not exist.</error>
</errors>
```

Note: Other response error codes exist. Refer to the ingest API for more information.

To Delete Collection Metadata

Collection metadata can be deleted by sending an HTTP DELETE to the URL https://cmr.sit.earthdata.nasa.gov/ingest/providers/<provider id >/collections/<native id>. The response will include the Concept ID and the Revision ID of the tombstone. For this example CMR_ONLY is the provider ID and cloud5 is the native ID. Please refer to HTTP Status Codes under Response Headers for status information.

Example

```
curl -i -XDELETE -H "Echo-Token: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" -H "Accept:
application/json"

http://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/collections/cloud5
```

If deletion is successful, the response code is 201 and the response body will include the Concept ID and the Revision ID as shown below. The return format will be either XML or JSON depending on what, if anything, was specified by the accept header.

XML

```
<?xml version="1.0"
encoding="UTF-8"?><result><concept-id>C1200018690-CMR_ONLY</concept-id><revision-id>2</revision-id></result>
```

JSON

```
{\"concept-id\":\"C1200018690-CMR_ONLY\",\"revision-id\":2}
```

If any errors occurred, the error responses will be the same as the ingest error messages.

To Delete the Token

To delete a token, execute the following curl command for your selected environment.

Substitute your token where <token> is written in the command below. *An example token is: 08D386C0-D020-A2BD-A65E-F54593A56FDB*

Example

```
curl -X DELETE --header "Content-Type: application/xml" https://cmr.earthdata.nasa.gov/legacy-se
rvices/rest/tokens/<token>
```

Note:

- *Depending on the environment you selected, the Base API URL may be different from the example. If so, just replace the purple text with the correct Base API URL that you need.*
- *If you are embedding the token REST messages into a programming language, create an HTTP message and place the same components from the curl example into either the message header or body.*

The return code should be a 204, otherwise an error has occurred.

Note: The process for validating, ingesting, and deleting granule metadata is identical to the one described above. Simply substitute the commands listed below:

Granule Validation Details

The following validations are performed when we validate or ingest granule metadata.

1. request Validation: Validates format specified and metadata length of the request. Returns errors when fail.
2. xml Validation: Validates against the native metadata schema (i.e. DIF, ECHO10, UMM-JSON). Returns errors when fail.
3. UMM validation: Returns warnings as default based on the setting of the ingest config: return-umm-spec-validation-errors, which is currently set to false.
   Can return errors when the config is set to true.
   a. Validation checks field values are in the expected format, within expected range and conform to controlled vocabularies where present.
   b. Sets the granule collection reference and validates the collection reference contains the required fields: short-name, version-id, entry-title and entry-id
   .

To Validate Granule Metadata

Just like collections, granule metadata can be validated without being ingested. However, a collection is required when validating a granule. The granule being validated can either refer to an existing collection in the CMR or the collection can be sent in a multi-part HTTP request. The URL for validation is: https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider Id>/validate/granule/<Native Id>. Please refer to HTTP Status Codes under Response Headers for status information.

The curl command for validating a granule with a collection that exists within the CMR where Provider ID is CMR_ONLY and the Native ID is granule1 is as follows:

Example

```
curl -i -XPOST -H "Content-Type: application/echo10+xml" -H "Echo-Token:
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"

" http://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/validate/granule/granule1 -d \

"<Granule>
   <GranuleUR>SC:AE_5DSno.002:30500511</GranuleUR>
   <InsertTime>2009-05-11T20:09:16.340Z</InsertTime>
   <LastUpdate>2014-03-19T09:59:12.207Z</LastUpdate>
   <Collection>
     <DataSetId>LarcDatasetId</DataSetId>
   </Collection>
   <Orderable>true</Orderable>
</Granule>"
```

A second option for granule validation without ingesting metadata is sending the parent collection with the granule.
The collection and granule contents are sent over HTTP using form multi-part parameters. The collection and granule metadata exist in files and are read into the post request body as parameters such as granule=<... granule xml ...> collection=<... collection xml ...>. In the example below, the granule metadata exists in the granule.xml file, the collection metadata exists in the collection.xml file, the Provider ID is CMR_ONLY, and the Native ID is "granule1."

Example

```
curl -i -XPOST -H "Echo-Token: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" \

-F "granule=<granule.xml;type=application/echo10+xml" \
-F "collection=<collection.xml;type=application/echo10+xml" \
"http://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/validate/granule/granule1"
```

The response messages for granules are the same those of collections.

Note: Specific metadata fields have been designated to require an enforced 'inheritance' validation between collections and associated

granules. This means that values for these fields, which are specified at the collection level, include a superset of all values which may be specified at the granule level. For example, if a collection specifies Campaigns A, B, and C — all associated granules may have any subset of these campaigns. However, if a granule specifies a Campaign D, it will be rejected. Collection and granule updates will always perform this validation check to ensure the inheritance is valid. Affected fields include the following:

- *Campaign*
- *Platform*
- *Instrument*
- *Sensor*
- *Additional Attributes*
- *Temporal Extent*
- *Spatial Extent*

To Ingest Granule Metadata

In order to ingest granules, a parent collection must already have been successfully ingested. A granule metadata can be ingested (create or update) by sending an HTTP PUT with the metadata to the URL https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider Id>/granules/<Native Id>. Once a granule is ingested to reference a parent collection, the granule can not be updated to reference a different collection as its parent collection. Please refer to HTTP Status Codes under Response Headers for status information.

Below is an example of a granule ingest using CMR_ONLY as the Provider ID and "granule1" as the Native ID.

Example

```
curl -i -XPUT -H "Content-Type: application/echo10+xml" -H "Echo-Token:
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" -d @granule1.xml
http://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/granules/granule1
```

The response messages for ingesting granules are the same as with ingesting collections.

To Delete Granule Metadata

Granule metadata can be deleted by sending an HTTP DELETE to the URL https://cmr.sit.earthdata.nasa.gov/ingest/providers/<Provider id>/granules/<Native Id>. The response will include the Concept ID and the Revision ID of the tombstone and is the same as the delete collection response. The following example uses CMR_ONY as the Provider ID and "granule1" as the Native ID. Please refer to HTTP Status Codes under Response Headers for status information.

Example

```
curl -i -XDELETE -H "Echo-Token: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
http://cmr.sit.earthdata.nasa.gov/ingest/providers/CMR_ONLY/granules/granule1
```

Warning
Deleting a collection will cascade and result in the deletion of all associated granules. However, a cascaded or explicit granule deletion will not cascade to delete associated browse records.

Chapter 5: The 4th Step - Data Management

Data Management tasks may be performed through the Metadata Management Tool (MMT) or, if the provider would like to write their own tool - the ECHO DataManagementService, AccessControlService (ACL), and Group2ManagementService APIs. For more information regarding MMT, refer to the Data Partners > Managing Data section of the ECHO website (https://earthdata.nasa.gov/echo/). For more information regarding how to use the ACL and Group Management functionality, refer to the following resources on the ECHO website (https://earthdata.nasa.gov/library/echo-opscon-documents): "Group Management & ACLs How To" guide (ECHO_Guide_003), the "ECHO ACLs and Roles" (ECHO_OpsCon_013), and the "ECHO Group Management" (ECHO_OpsCon_014) operations concepts.

Access Control Concepts

As a Data Partner interacting with the CMR access control capability, you will need to understand the following basic concepts.

Provider Objects vs. Catalog Items

Data Partner using the ECHO API can control access to two types of data objects within the CMR system:

- **Provider Objects** – Provider information items that are accessible only by members of the Data Partner and CMR Operations Teams. Examples consist of provider orders, order policies, and provider groups.
- **Catalog Items** – Provider metadata items that are made available for discovery and ordering to end users. This includes the control of collection and granule items, but not explicitly browse.

Groups

"Group" refers to provider-defined groups of CMR registered users that may have specific access to Provider Objects or Catalog Items based upon the group's permissions. In addition, CMR Operations manages system level groups for their own access. Virtual system groups (e.g. guest users) are also available for specific purposes.

A CMR Group is identified by the following information:

- **Name** – Unique group name within the Data Partner's list of groups
- **Description** – The purpose of the group and a roster of user members.
- **Members** – List of ECHO registered users who are a part of the group.
- **Provider** – The provider who owns the group, or possibly the system if it is a system level group.

CMR groups are associated exclusively with one CMR provider and appear only in the context of that provider. In addition, the ECHO API supports retrieving groups for a single provider at a time.
As such, all groups owned by a single provider must have unique names, but can duplicate names of groups belonging to other providers in the system. CMR Data Partners may create as many groups as are needed in order to fulfill their data access control needs.

The CMR also supports the concept of system level groups, which are managed by the CMR Operations Team. These groups allow the CMR Operations Team to manage their own access to *Provider Objects* and *Catalog Items* without needing to coordinate with each provider. Although the CMR Operations Team will have all permissions on all objects, they will continue to communicate changes with the CMR Data Partners to ensure good coordination.

In addition to the CMR Operation team's managed system groups, the CMR also has the concept of *virtual* system groups. These groups are "managed" by the CMR system and include a *Registered Users* group and *Guest Users* group. Due to the nature of these lists, the CMR will dynamically associate a user with one of these virtual groups when assessing permissions. Permissions to **both** of these groups must be independently managed.

The permissions to create new groups and view existing groups are managed by assigning permissions on the *Group* Provider Object ACL. Permissions to update or delete existing groups are managed by assigning permissions on the *Group Management* Provider Object ACL. When creating a new group, an initial group must be specified as the *Initial Management Group*. This group will be given permissions to update and delete the new group, and others may be added later.

Access Control List (ACL)

An Access Control List is responsible for linking groups to a specific *Provider Object* or *Catalog Item* and describing the permissions the group has been granted. All *Provider Objects* and *Catalog Items* are **inaccessible** by default. A CMR Data Partner uses ACLs to grant permissions to CMR groups, so that group members will have access to specific *Provider Objects* or sets of *Catalog Items*. An ACL may exist without any assigned permissions. For example, a CMR Data Partner may wish not to assign any permissions to the "Extended Services" *Provider Object*, or they may define a *Catalog Item* ACL that controls access to a specific data set, but choose to not assign any permissions at the present time.

**Note: Catalog Item ACLs are assigned to a custom, static, or dynamic listing of collection and granule items.*

Permissions

Each *Provider Object* or *Catalog Item* has a specific set of permissions (e.g. Create, Read, Update, Delete) that may be granted to a group. The permissions for each *Catalog Item* may be *View* or *Order,* without exception. The permissions for each *Provider Object* may be *Create, Read, Update,* or *Delete* - but the available permissions differ depending on the nature of that object. For example, the *Provider Policies* object can be granted the *Read* or *Update* permissions, while the *Provider Audit Report* object can only be granted the *Read* permission. Some permissions, such as the ability to read option definitions, are not grantable on some *Provider Objects* because access is open to any user. The following table outlines the grantable permissions for each *Provider Object*.

Provider Object Grantable Permissions

Provider Object	Grantable Permissions	Description
Audit Report	Read	Allows the viewing of an audit report for actions associated with a specific provider
Dataset Information	Read	Allows the usage of the reconciliation GetDatasetInformation() method.
Extended Services (all types)	Create, Update, Delete	Allows the creation, updating, and deletion of extended services.
Ingest Operations	Read, Update	Controls access to who can log into the EIAT (Not used by ECHO Ingest).
Groups	Create, Read	Allows the creation of new groups or viewing of existing provider groups
Group Management	Update, Delete	Allows the updating or deletion of an existing provider group.
Option Assignments	Create, Read, Delete	Allows the assignment of an option definition to one or more datasets.
Option Definitions	Create, Delete	Allows the creation and deleting of an option definition.
Option Definition Deprecation	Create	Allows the deprecation of an option definition.
Provider Context	Read	Allows a user to act as a provider and perform all permitted provider actions.
Provider Holdings	Read	Allows the viewing of a provider‘s holdings (dataset & granule count).
Provider Information	Update	Allows provider information to be updated.
Provider Orders	Read	Allows the viewing of all orders associated with a specific provider.
Provider Order Resubmission	Create	Allows the resubmission of a provider‘s order
Provider Order Acceptance	Create	Allows the acceptance of a provider‘s order. Order Fulfillment Service Users (EWOC) will use this ACL.
Provider Order Rejection	Create	Allows the rejection of a provider‘s order. Order Fulfillment Service Users (EWOC) will use this ACL.
Provider Order Closure	Create	Allows the closure of a provider‘s order. Order Fulfillment Service Users (EWOC) will use this ACL.
Provider Order Tracking Id	Update	Allows an order to be updated with a provider tracking ID. Order Fulfillment Service Users (EWOC) will use this ACL.
Provider Policies	Read, Update, Delete	Allows the editing of provider policies.
User	Read	Allows the viewing, updating, and deletion of an ECHO user.
Authenticator Definition	Create, Delete	Allows the creation, deletion of provider authenticators. (Not currently being used)

Catalog Item Identifiers

Each *Catalog Item* ACL will have a unique set of identifiers that specify the catalog items to which the ACL will apply. The three main classes of identifiers are: Catalog Item Type, Collection Identifiers, and Metadata Filters.

Catalog Item Type

When creating a *Catalog Item* ACL, you may choose to have it apply to *Collections*, *Granules*, or *Both* by selecting the appropriate catalog item type.

- Select "*Collections*" if you want the ACL to be used when granting access to collection items. This is useful if a collection doesn't have granules or if the granules will require different permissions.
- Select "*Granules*" if you want the ACL to be used when granting permission for access granule items but not collections.
- Select "*Both*" if you want the ACL to be used when granting access to both collection and granule items. This is useful if the

collections and granules will have the same permissions.

Collection Identifiers

Each *Catalog Item* ACL has a list of collections to which the ACL applies. This list is used to identify both the collections themselves or granules within the collections. The list of collections may exist as a:

- **Selected List** – A static list of collections which must be manually managed.
- **Any Collection** – A dynamic list of all collections in the Data Partner's holdings
- **Collection Pattern Matching** – Patterns may be selected to perform text matching on the Data Set ID, Short Name, and/or Version ID.

Metadata Filters

In order to facilitate access control of collections and granules based on metadata fields, a *Catalog Item* ACL may contain collection and granule metadata filters. The filters are based on temporal fields, restriction flag values, or specific granule UR values. Temporal filters may apply to the acquisition, production, or metadata dates of Insert or Last Update fields and may be described using an intersection, containment, or disjoint comparator. The full listing of filters is included below:

- **Collection Filters**
  - **Temporal Range**
  - **Rolling Temporal Range**
  - **Restriction Flag**
- **Granule Filters**
  - **Temporal Range**
  - **Rolling Temporal Range**
  - **Restriction Flag**
  - **Granule UR Pattern**

Provider "Administrators" Group

At the start, each CMR Data Partner will be configured with a group named "Administrators." This group will be granted all permissions on all *Provider Object* ACLs and the permissions to manage *Catalog Item* ACLs. This group is initially managed by itself and the system "Administrators" group, which consists of the CMR Operations team. The provider "Administrators" group may manage *Provider Object* ACLs and grant permissions to other groups, however the ability to manage and grant permissions may not be given to other provider groups. The ability to grant management of *Catalog Item* ACLs can be granted to other groups. This is distinctly different than *Provider Object* ACL management and is designed to allow for groups such as a Data Partner's User Services team to manage *Catalog Item* ACLs without coordinating with the provider "Administrators."

RestrictionFlag

The restriction flag is a decimal value that is specified in a Data Partner's collection or granule metadata and is used to restrict the availablility of the data. A granule or collection's restriction flag cannot be updated through the API. To update the restriction flag, the data must be re-ingested or the catalogue item ACL must be modified.

Access Control Recommendations

The following recommendations outline some suggested access control mechanisms to facilitate the data management needs of both the Data Partner and the CMR Operations team.

Data Mgmt & User Services Groups

As has been described, a provider "Administrators" group will be created and granted all permissions on all *Provider Objects*. It is suggested that membership in this group be limited to those individuals who have need of managing access to the provider. There are two distinct "roles" which may be facilitated by provider groups: a "Data Management" group and a "User Services" group. The "Data Management" group is given a subset of *Provider Object* ACL permissions relevant to their job role. The "User Services" group has a slightly different subset of *Pr ovider Object* ACL permissions, but has the ability to manage *catalog item* ACLs.

Catalog Item ACLs

The CMR Operations team will define two *Catalog Item* ACLs during the initial configuration:

- **All Collections (No Granules)** – This ACL dynamically applies to all collections within a Data Partner's holdings, and only affects access to collection metadata. The Operations team uses this ACL to assign view permissions for the WIST valids process.
- **All Collections and Granules** – This ACL dynamically applies to all granules in all collections within a Data Partner's holdings, and affects access to both collection and granule metadata. This ACL is useful for assigning full permissions to members of the Data Partner team. The Operations team uses this ACL to assign *view* permissions for the System "Administrators" group.

The following *Catalog Item* ACL conditions are suggested in order to facilitate general data management:

- **Public Collections and Granules** – This ACL dynamically applies to all granules within a static listing of public collections. As collections are added to the provider's holdings, the provider may add the collection to the listing used by this ACL. *View* permissions to this ACL may be granted to the *Registered Users* and *Guest Users* system level groups to allow for data discovery.
- **Orderable Granules** – This ACL dynamically applies to all granules within a static listing of collections that contain orderable granules. *Order* permissions to this ACL may be granted to the *Registered Users* and *Guest Users* system level groups to allow for order creation and submission.

Chapter 6: CMR Metadata - Past the Basics

Temporal Data (aka Acquisition Date and Time)

Temporal metadata refers to the date and time data in a collection or granule was acquired. Temporal data is an essential search criteria for collections and granules within the CMR. Temporal information is not a required element for collections or granules; however, if a granule specifies temporal information, the collection must do so as well. CMR ingest will validate that a granule's temporal data falls within the range of its parent collection's temporal data. If a collection specifies an open ended temporal range, CMR will accept all granules that provide temporal data subsequent to the collection's starting date.

Collections

A collection may be associated with one or more of the following temporal expression types:

- **Single Date Time** – A single date and time.
- **Range Date Time** – A date and time range - both defined by a beginning and end. An ending date is not required in order to designate an on-going collection.
- **Periodic Date Time** – A repeating date and time range, both defined by a beginning and end, with a period cycle and duration.

*Note: Ranges within a collection may also include periods of time for which no data was collected. As such, CMR will not only dilineate the start and stop time of acquisition for the entire collection, but will also internally calculate the beginning and end date/times of ranges when no data was collected.

In addition, the following may be specified in order to provide users with additional collection temporal information:

- **TimeType** – The time system used to represent the time values in the temporal ranges. (e.g. UTC)
- **DateType** – The date system used to represent the date values in the temporal ranges. (e.g. Gregorian)
- **Temporal RangeType** – Designates how the temporal coverage is defined. (e.g. Continuous Range)
- **Precision of Seconds** – The degree of exactness used in the measurement of seconds
- **Ends At Present** – Boolean flag denoting that a data collection, which temporally covers a discontinuous range, currently ends at the present date. Thus, the granules that are continuously being added to the collection inventory don't need to update the collection metadata.

These expressions are used to provide the necessary information in order to uniquely identify the temporal range for which data may be discovered within the collection.

Expression of Temporal Information Using ECHO10 Specification (Range DateTime)

```
<Temporal>
    <TimeType>UTC</TimeType>
    <DateType>Gregorian</DateType>
    <TemporalRangeType>Continuous Range</TemporalRangeType>
    <PrecisionofSeconds>1</PrecisionofSeconds>
    <EndsatPresentFlag>Y</EndsatPresentFlag>
    <RangeDateTime>
        <BeginningDateTime>1998-01-01T00:00:00.0Z</BeginningDateTime>
    </RangeDateTime>
</Temporal>
```

Granules

A granule may be associated with one of the following temporal expressions types:

- **Single Date Time** – A single date and time.
- **Range Date Time** – A date and time range - both defined by a beginning and end. An ending date is not required in order to designate an on-going granule.

*Note: Granules may also include periods of time for which no data was collected. As such, CMR will not only dilineate the start and stop time of acquisition for the entire granule, but will also internally calculate the beginning and end date/times of ranges when no data was collected.

These expressions are used to provide the necessary information in order to uniquely identify the temporal range for which data in the granule represents.

Expression of Temporal Information Using ECHO10 Specification (RangeDateTime)

```
<Temporal>
    <RangeDateTime>
        <BeginningDateTime>1998-01-01T00:00:00.0Z</BeginningDateTime>
        <EndingDateTime>1998-01-01T00:00:00.0Z</EndingDateTime>
    </RangeDateTime>
</Temporal>
```

Additional Attributes

Additional attributes, also known as Provider-Specific Attributes (PSAs), are parameters which further describe the data represented by each granule in a collection. These values are important search criteria for the granules. Additional attribute examples include values for MODIS Tile grid coordinates and elevation information. All additional attribute definitions must be specified at collection level metadata. A collection is permitted to specify a value that is applicable to all granules within that collection. Granules reference defined additional attributes and supply a value that is associated to that granule. While granules can have different values for the additional attribute parameter, they may not define a new additional attribute. All new additional attributes must be defined at the collection level.

The following table shows the supported additional attribute data types within CMR. The XML Type column lists the XML schema type that will be used to validate any attribute values specified in the collection. The Range Validation column specifies whether range values are allowed at the collection level, and, if present, whether granules additional attribute values will be validated against the supplied ranges. The *_STRING types allow providers to specify date, time, and datetime attribute values without the associated data validation performed for the pure date and time types.

Supported Additional Attribute Data Types

Attribute Type	XML Type	Range Validation
STRING	string	No
FLOAT	float	Yes
INT	int	Yes
BOOLEAN	boolean	No
DATE	date	Yes
TIME	time	Yes
DATETIME	dateTime	Yes
DATE_STRING	string	No
TIME_STRING	string	No
DATETIME_STRING	string	No

Collections

An additional attribute, as defined within collection metadata, must include the following information:

- **Name** – Unique name of the additional attribute.
- **DataType** – Selected from the supported list of types.
- **Description** – A textual description of the attribute to help end users understand the purpose and data represented by the attribute.

An additional attribute definition may also contain the following elements:

- **Measurement Resolution** – Identifies the smallest unit increment to which the parameter value is measured.
- **Parameter Range Begin** – Minimum value of all attribute values that will be provided by the collection or granules.
- **Parameter Range End** – Maximum value of all attribute values that will be provided by the collection or granules.
- **Parameter Units Of Measure** – Unit of measure used for quantifying the parameter (e.g. AVHRR: unit of geophysical parameter-units of geophysical parameter.)
- **Parameter Value Accuracy** – Estimate of the variation from the actual attribute value. This can be specified in percent or the unit with which the parameter is measured.
- **Value Accuracy Explanation** – Defines the method used for determining the parameter value accuracy.

As was discussed previously, CMR Ingest will perform value validation based upon the attribute's data type and range values. See the table in the previous section for more information regarding validation.

collection-Level Additional Attributes using ECHO10 specification

```
<AdditionalAttributes>
    <AdditionalAttribute>
        <Name>PROCESSVERSION</Name>
        <DataType>STRING</DataType>
        <Description>foo</Description>
    </AdditionalAttribute>
    <AdditionalAttribute>
        <Name>HORIZONTALTILENUMBER</Name>
        <DataType>INT</DataType>
        <Description>foo</Description>
        <ParameterRangeBegin>1</ParameterRangeBegin>
        <ParameterRangeEnd>100</ParameterRangeEnd>
    </AdditionalAttribute>
</AdditionalAttributes>
```

Granules

An additional attribute, as defined within granule metadata, must include the following information:

- **Name** – Unique name of the additional attribute as defined by the collection.
- **Values** – One or more values for the additional attribute.

As was discussed previously, CMR Ingest will validate that an additional attribute with the same name is defined within the granule's collection. Also, Ingest may perform value validation based upon the attribute's data type and range values. See the table in the previous section for more information regarding validation. If more than one value is provided, CMR will preserve the order of the values when queried through the CMR API.

Granule level additional attributes using ECHO10 Specification

```
<AdditionalAttributes>
    <AdditionalAttribute>
        <Name>VERTICALTILENUMBER</Name>
        <Values>
            <Value>12</Value>
        </Values>
    </AdditionalAttribute>
    <AdditionalAttribute>
        <Name>TileID</Name>
        <Values>
            <Value>51013012</Value>
            <Value>51013013</Value>
        </Values>
    </AdditionalAttribute>
</AdditionalAttributes>
```

Platforms, Instruments And Sensors

CMR adopts a layered representation of platforms, instruments (also known as sources), and sensors in collection and granule metadata. This layered representation is as follows:

```
Platform->* Instrument->* Sensor
```

All possible platforms, instruments, and sensors that may exist within the granules must be defined within the collection metadata.
A collection could be associated with zero (0) or more platforms; each platform could contain zero (0) or more instruments, and each instrument could contain zero (0) or more sensors. Each item is uniquely identified by that item's **Short Name** element value. Granules reference defined platform/instrument/sensor combinations associated with that granule. Granules may not define a new platform/instrument/sensor combination that is not defined by the collection.
The platform, instrument, and sensor must comply with the GCMD standard, located at http://gcmd.gsfc.nasa.gov/Resources/valids/index.html.

Collections

A collection definition contains the following platform sub-elements :

- **ShortName** – Unique name of the platform within the collection.
- **LongName** – The expanded or long name of the platform associated with an instrument. (Required)
- **Type** – The most relevant platform type. (Required)

- **Characteristics** – The characteristics of platform specific attributes. The characteristic names must be unique on this platform; however the names do not have to be unique across platforms.

A collection definition contains the following instrument sub-elements :

- **ShortName** – Unique name of the instrument within the collection.
- **LongName** – The expanded name of the primary sensory instrument.
- **Technique** - Technique applied for this instrument in the configuration.
- **Number Of Sensors** - Number of sensors used on the instrument when acquiring the granule data.
- **Characteristics** - The characteristics of this instrument expressed as custom attributes. The characteristic names must be unique on this instrument; however the names do not have to be unique across instruments.
- **Operation Modes** - The operation mode applied on the instrument when acquiring the granule data.

A collection definition contains the following sensor sub-elements:

- **ShortName** – Unique name of the sensor within the collection.
- **LongName** – The expanded name of the sensor.
- **Technique** - Technique applied for this sensor in the configuration.
- **Characteristics** - The characteristics of this sensor expressed as custom attributes. The characteristic names must be unique on this sensor; however the names do not have to be unique across sensor.

Full Platform/Instrument/Sensor Description Using ECHO10 Specification

```
<Platform>
    <ShortName>Terra</ShortName>
    <LongName>First EOS Polar Orbiting Satellite, 10:30 AM Descending
Equator Crossing </LongName>
    <Type>Spacecraft</Type>
    <Characteristics>
        <Characteristic>
            <Name>EquatorCrossingTime</Name>
            <Description>Local time of the equator crossing and
direction (ascending or descending)</Description>
            <DataType>varchar</DataType>
            <Unit>Local Mean Time</Unit>
            <Value>10:30, descending</Value>
        </Characteristic>
    </Characteristics>
    <Instruments>
        <Instrument>
            <ShortName>MODIS</ShortName>
            <LongName>Moderate-Resolution Imaging
Spectroradiometer</LongName>
            <Technique>Imaging Spectroradiometry</Technique>
            <NumberOfSensors>2</NumberOfSensors>
            <Characteristics />
            <Sensors>
                <Sensor>
                    <ShortName>MODIS</ShortName>
                    <LongName>Cross-track Scanning Radiometer</LongName>

                    <Technique>Radiometry</Technique>
                    <Characteristics />
                </Sensor>
            </Sensors>
        </Instrument>
    </Instruments>
</Platform>
```

Granules

The granule metadata includes the following platform sub-elements:

- **ShortName** – Unique name of the platform within the collection.

The granule metadata includes the following instrument sub-elements:

- **ShortName** – Unique name of the instrument within the collection.
- **Characteristics** - The characteristics of this instrument expressed as custom attributes. The characteristic names must be unique on this instrument; however the names do not have to be unique across instruments.
- **Operation Modes** - The operation mode applied on the instrument when acquiring the granule data.

The granule metadata includes the following sensor sub-elements:

- **ShortName** – Unique name of the sensor within the collection.
- **Characteristics** - The characteristics of this sensor expressed as custom attributes. The characteristic names must be unique on this sensor; however the names do not have to be unique across sensors.

As was discussed previously, CMR Ingest will validate that an additional attribute with the same name is defined within the granule's collection. Also, ingest may perform value validation based upon the attribute's data type and range values. See the table in the previous section for more information regarding validation. If more than one value is provided, CMR will preserve the order of the values when queried through the CMR API.

Sources and Sensors Using ECHO10 Specification

```
<Platform>
    <ShortName>Terra</ShortName>
    <Instruments>
        <Instrument>
            <ShortName>MODIS</ShortName>
            <Sensors>
                <Sensor>
                    <ShortName>MODIS</ShortName>
                </Sensor>
            </Sensors>
        </Instrument>
    </Instruments>
</Platform>
```

Measured Parameters

Measured parameters are applicable exclusively to granules and serve as important granule level search criteria. For some providers, the value of certain measured parameters determines the visibility of the granule.

Measured parameters contain the name of the geophysical parameter expressed in the data as well as associated quality status and quality flags.The quality status contains measures of quality for the granule. The parameters used to set these measures are not preset and will be determined by the data producer. Each set of measures can occur many times either for the granule as a whole or for individual parameters. The quality flags contain the science, operational, and automatic quality flags that indicate the overall quality assurance levels of specific parameter values within a granule.
A measured parameter is uniquely identified by its **Parameter Name** element, and has the following information:

- **QA Stats** – The name of the geophysical parameter expressed in the data as well as associated quality flags and quality status
  - **QA Percent Missing Data** - Granule level % missing data. This attribute can be repeated for individual parameters within a granule.
  - **QA Percent Out Of Bounds Data** – Granule level % out of bounds data. This attribute can be repeated for individual parameters within a granule.
  - **QA Percent Interpolated Data** – Granule level % interpolated data. This attribute can be repeated for individual parameters within a granule.
  - **QA Percent Cloud Cover** – This attribute is used to characterize the cloud cover amount of a granule. This attribute may be repeated for individual parameters within a granule. (Note - there may be more than one way to define a cloud or it's effects within a product containing several parameters; i.e. this attribute may be parameter specific)
- **QA Flags** – The name of the geophysical parameter expressed in the data as well as associated quality flags and quality status.
  - **Automatic Quality Flag** – The granule level flag applying generally to the granule and specifically to parameters the granule level. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The parameters determine whether the flag is set are defined by the developer and documented in the Quality Flag Explanation.
  - **Automatic Quality Flag Explanation** – A text explanation of the criteria used to set automatic quality flag, including thresholds or other criteria.
  - **Operational Quality Flag** – The granule level flag applying both generally to a granule and specifically to parameters at the granule level. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The parameters determining whether the flag is set are defined by the developers and documented in the Operational Quality Flag Explanation.
  - **Operational Quality Flag Explanation** – A text explanation of the criteria used to set operational quality flag; including thresholds or other criteria.
  - **Science Quality Flag** – Granule level flag applying to a granule, and specifically to parameters. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The parameters determining whether the flag is set are defined by the developers and documented in the Science Quality Flag Explanation.
  - **Science Quality Flag Explanation** – A text explanation of the criteria used to set science quality flag; including thresholds or other criteria.

Measured Parameters Using ECHO10 Specification

```
<MeasuredParameters>
    <MeasuredParameter>
        <ParameterName>Snow_Cover_Daily_Tile</ParameterName>
        <QAStats>
            <QAPercentMissingData>0</QAPercentMissingData>
            <QAPercentCloudCover>65</QAPercentCloudCover>
        </QAStats>
        <QAFlags>
            <AutomaticQualityFlag>Passed</AutomaticQualityFlag>
            <AutomaticQualityFlagExplanation>
                No automatic quality assessment done in the PGE \
            </AutomaticQualityFlagExplanation>
            <OperationalQualityFlag>Passed</OperationalQualityFlag>

<OperationalQualityFlagExplanation>Passed</OperationalQualityFlagExplana
tion>
            <ScienceQualityFlag>Not Investigated</ScienceQualityFlag>
            <ScienceQualityFlagExplanation>
                See
http://landweb.nascom.nasa.gov/cgi-bin/QA_WWW/qaFlagPage.cgi? sat=terra
for the product Science Quality status.
            </ScienceQualityFlagExplanation>
        </QAFlags>
    </MeasuredParameter>
</MeasuredParameters>
```

Online Data Access URL And Online Resources URL

For some collections or granules, the raw data are made available online via FTP or web URL. The FTP or web URL is stored in the CMR. Directly accessible data require the <OnlineAccessURLs> tag and include the URL to that data. Use *Online Access URLs* only for the actual data. *Online Access URLs* specified in collection and granule metadata include the following elements:

- **URL** – URL for the online data. (Required)
- **URL Description** – Description of the data available via the supplied URL.
- **Mime Type** – The specification of the online data.

Any other online information covering aspects of the data, such as guides, product listings, validation information, etc., should be listed in <OnlineResources>, along with the URLs to that information. Online resource URLs specified in collection and granule metadata include the following elements:

- **URL** – URL for the online resource. (Required)
- **URL Description** – Description of the resource available via the supplied URL.
- **Type** - The type of the resource such as 'collection Guide' or 'Campaign Guide' etc. This value should be a short phrase that can be used in a CMR client for displaying the URL. (Required)
- **Mime Type** – The specification of the online resource

Online Access and Resource URLs Using ECHO10 Specification

```
<OnlineAccessURLs> <OnlineAccessURL>
<URL>ftp://daac.nasa.gov/granule_1234.zip</URL>
<URLDescription>Compressed data granule</URLDescription>
<MimeType>application/zip</MimeType> </OnlineAccessURL>
</OnlineAccessURLs> <OnlineResources> <OnlineResource>
<URL>http://daac.nasa.gov/products/product_A.html</URL>
<URLDescription>Main product overview page.</URLDescription>
<Type>Product Overview</Type> <MimeType>text/html</MimeType>
</OnlineResource> </OnlineResources>
```

Keywords

The CMR supports three kinds of keyword associations for collections: science keywords, spatial keywords, and temporal keywords. Science keyword and spatial keyword values should come from the Global Change Master Directory (GCMD) keywords standard located at http://gcmd.nasa.gov/learn/keyword_list.html.

The science keyword metadata element fully implements the GCMD keyword hierarchy, which contains the following fields. There are no associated granule metadata elements for the science keyword.

- **Category** –. Keyword used to describe the general category of the collection.
- **Topic** – Keyword used to describe the general topic of the collection.
- **Term** – Keyword used to describe the science parameter area.
- **VariableLevel1** – Keyword containing the first level science keyword variable.
- **VariableLevel2** – Keyword containing the second level science keyword variable.
- **VariableLevel3** – Keyword containing the third level science keyword variable.
- **DetailedVariable**– Keyword containing a free form field for further keyword specification.

Sample Science Keyword Using ECHO10 Specification

```
<ScienceKeywords>
  <ScienceKeyword>
     <CategoryKeyword>EARTH SCIENCE</CategoryKeyword>
     <TopicKeyword>CLIMATE INDICATORS</TopicKeyword>
     <TermKeyword>ATMOSPHERIC/OCEAN INDICATORS</TermKeyword>
     <VariableLevel1Keyword>
         <Value>OCEAN UPWELLING INDICES</Value>
         <VariableLevel2Keyword>
             <Value>OCEAN COASTAL UPWELLING INDEX</Value>
             <VariableLevel3Keyword>CUI</VariableLevel3Keyword>
         </VariableLevel2Keyword>
     </VariableLevel1Keyword>
     <DetailedVariableKeyword>MORE DETAILED DESCRIPTIVE
KEYWORD</DetailedVariableKeyword>
  </ScienceKeyword>
</ScienceKeywords>
```

The spatial keyword metadata element does not currently implement the updated GCMD keyword hierarchy. The CMR spatial keyword contains a single value, which can be understood as the 'Location' field in the GCMD managed list of keywords. There are no associated granule metadata elements for the spatial keyword.

Sample Spatial Keyword Using ECHO10 Specification

```
<SpatialKeywords>
    <Keyword>GEOGRAPHIC REGION</Keyword>
</SpatialKeywords>
```

The temporal keyword metadata element is not associated with a GCMD managed keyword list. The CMR temporal keyword contains a single value. There are no associated granule metadata elements for the temporal keyword.

Sample Temporal Keyword Using ECHO10 Specification

```
<TemporalKeywords>
    <Keyword>UTC</Keyword>
</TemporalKeywords>
```

Chapter 7: Spatial Representations

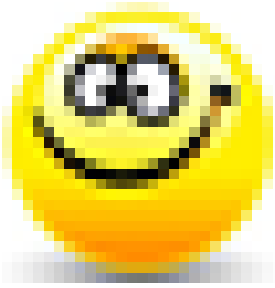
This chapter describes the terms and concepts relevant to the CMR's spatial model. Examples are provided for each supported spatial representation and include discussions regarding known spatial topics.

Spatial overview

Spatial metadata refers to the area of the Earth that a collection or granule covers. The spatial coverage area for a granule should be within the spatial coverage area of its primary collection. Ingest will not perform validation for this relationship. To ensure that spatial searches will return the correct results, you must prepare the spatial metadata according to the detailed guidelines discussed below.
The CMR system accepts spatial data represented in the Cartesian and Geodetic coordinate systems and also accepts spatial information representing orbital data. Refer to the table below for supported spatial data types and guidelines for limitations to granularity. You should choose a coordinate system based on the original data's spatial area size and projection. You may not combine spatial types for granules within the same collection as the representations are mutually exclusive.

Implementation of both the Cartesian coordinate system and the Geodetic coordinate system (World Geodetic System 84) accepts spatial data types of Point, Line, Bounding Box , and Polygon.

Supported Spatial Data Types

Spatial Data Type	Cartesian	Geodetic	Orbit	Guidelines and restrictions
Point				

Bounding Box				Stores bounding box data as a polygon with four vertices.
Line				A line may not have consecutive vertices with the same latitude and longitude. A line must be less than one half the circumference of the Earth in the Geodetic coordinate system.
Polygon				A polygon‘s vertices must be stored in order of vertex connection. Provide the vertices in clockwise order. No consecutive vertices may have the same latitude and longitude, that is, no repeating points. Also, polygonal lines may not cross each other within the polygon. No polygon should cover more than half the Earth in the Geodetic coordinate system.

The CMR system will not manipulate any of the spatial input metadata. It is your responsibility to ensure the accuracy and integrity of your spatial metadata. To prepare your metadata so that it can be searched, follow the guidelines in the sections below.

Collection & Granule Spatial Relationships

The following items should be considered when generating spatial metadata.

- Each collection may specify only one coordinate system for its spatial coverage.
- Each collection‘s coordinate system is independent of all other collections.
- Each collection‘s coordinate system is independent of its granule spatial representation — i.e., a collection‘s spatial extent may be expressed in the Cartesian geometry, but have all of it‘s granules specify their spatial extents in the Geodetic geometry.
- A collection specifies its granules‘ spatial representation, which cannot be overridden by a granule.
- A collection with an orbital granule spatial representation must specify exact orbit parameters in order to facilitate granule discovery via spatial constraints.
- Ingest for a metadata record will fail if any spatial metadata input is invalid with respect to the associate rules of the utilized coordinate system.

Geometry Representations

Spatial data are most commonly described using geometry terms - such as a polygon or line. They are stored as spatial objects to record shape, spatial locations of corner points, and the spatial coordinate system employed (e.g. Cartesian or Geodetic). It is important to ensure that the correct coordinate system is specified. The same set of coordinates in the Cartesian and Geodeditic system will respresent two different spatial areas; and thus affects the discovery of data and the representation of that data to users via CMR clients.

Coordinate Systems

Cartesian Coordinate System

The Cartesian coordinate system is a flattened coordinate system with longitude ranging from –180 to 180 degrees and latitude ranging from

–90 to 90 degrees. The projected map is flattened and open along the International Date Line (antimeridian) with the North Pole and South Pole as top and bottom line respectively. Please be aware of the following Cartesian coordinate system constraints:

- Any single spatial area may not cross the International Date Line (unless it is a bounding box) or Poles.
- Two vertices will be connected with a straight line.

Geodetic Coordinate System

The Geodetic coordinate systems are angular coordinates (longitude and latitude) and are closely related to spherical polar coordinates. They are defined relative to a particular Earth geodetic datum. The CMR implementation of the Geodetic coordinate system follows OGC standards, which are defined at http://www.opengeospatial.org/. The World Geodetic System 84 (WGS 84) is the supported geodetic system. Please be aware of the following geodetic coordinate system constraints:

- The implemented Geodetic model uses the great circle distance to connect two vertices for constructing a polygon area or line. If there is not enough density (that is, the number of points) for a set of vertices, then the line or the polygon area might be misinterpreted or the metadata might be considered invalid.
- Any single spatial area may cross the International Date Line and/or Poles
- Any single spatial area may not cover more than one half of the earth.

Data Types and Representation

Geometry

Spatial data in Cartesian or Geodetic coordinate systems are specified within a <Geometry> tag.

Geometry Example

```
<Spatial>
    …
    <HorizontalSpatialDomain>
        …
        <Geometry>
            …
        </Geometry>
        …
    </HorizontalSpatialDomain>
</Spatial>
```

Within a <Geometry> tag - points , lines, bounding boxes, and/or polygons can be included to define the spatial extent of your data.

Point

The CMR can receive, store, and search for spatial data representing one or more points. In the XML metadata, follow the syntax as shown in the following code sample to define a spatial extent as one or more points:

Single Point Example

```
<Geometry>
    <Point>
        <PointLongitude>-123.948</PointLongitude>
<PointLatitude>45.0664</PointLatitude>
    </Point>
</Geometry>
```

Multiple Points

```
<Geometry>
    <Point>
        <PointLongitude>-123.948</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
    </Point>
    <Point>
        <PointLongitude>-133.546</PointLongitude>
        <PointLatitude>45.0664</PointLatitude>
    </Point>
</Geometry>
```

Line

The CMR can receive, store, and search for spatial data representing one or more lines. In the XML metadata, follow the syntax shown in the following code sample to define a spatial extent as one or more lines:

Single Line Example

```
<Geometry>
    <Line>
        <Point>
            <PointLongitude>-123.948</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-133.546</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
    </Line>
</Geometry>
```

Multiple Line Example

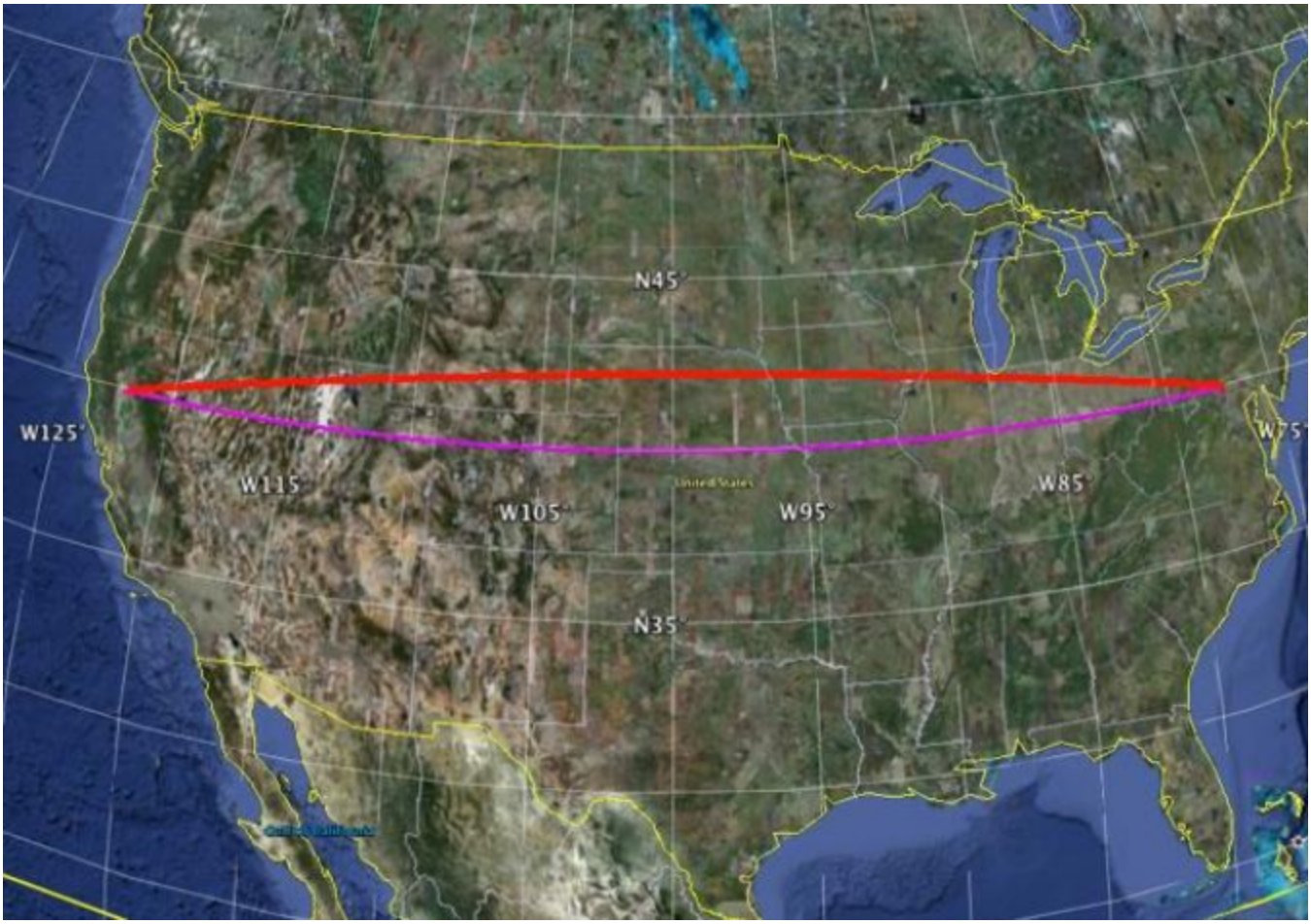
```
<Geometry>
    <Line>
        <Point>
            <PointLongitude>-123.948</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-133.546</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
    </Line>
    <Line>
        <Point>
            <PointLongitude>-123.948</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-133.546</PointLongitude>
            <PointLatitude>45.0664</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-143.546</PointLongitude>
            <PointLatitude>40.0664</PointLatitude>
        </Point>
    </Line>
</Geometry>
```

Due the differences in how the Cartesian and Geodetic coordinate systems represent spatial data, the same line will represent a noticeably different path on the Earth's surface depending on the employed coordinate system. The following line segment is shown in its Cartesian and Geodetic representation.

Interpreted in Cartesian and Geodetic Systems

```
<Geometry>
    <Line>
        <Point>
            <PointLongitude>-122.1</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-77.35</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
    </Line>
</Geometry>
```

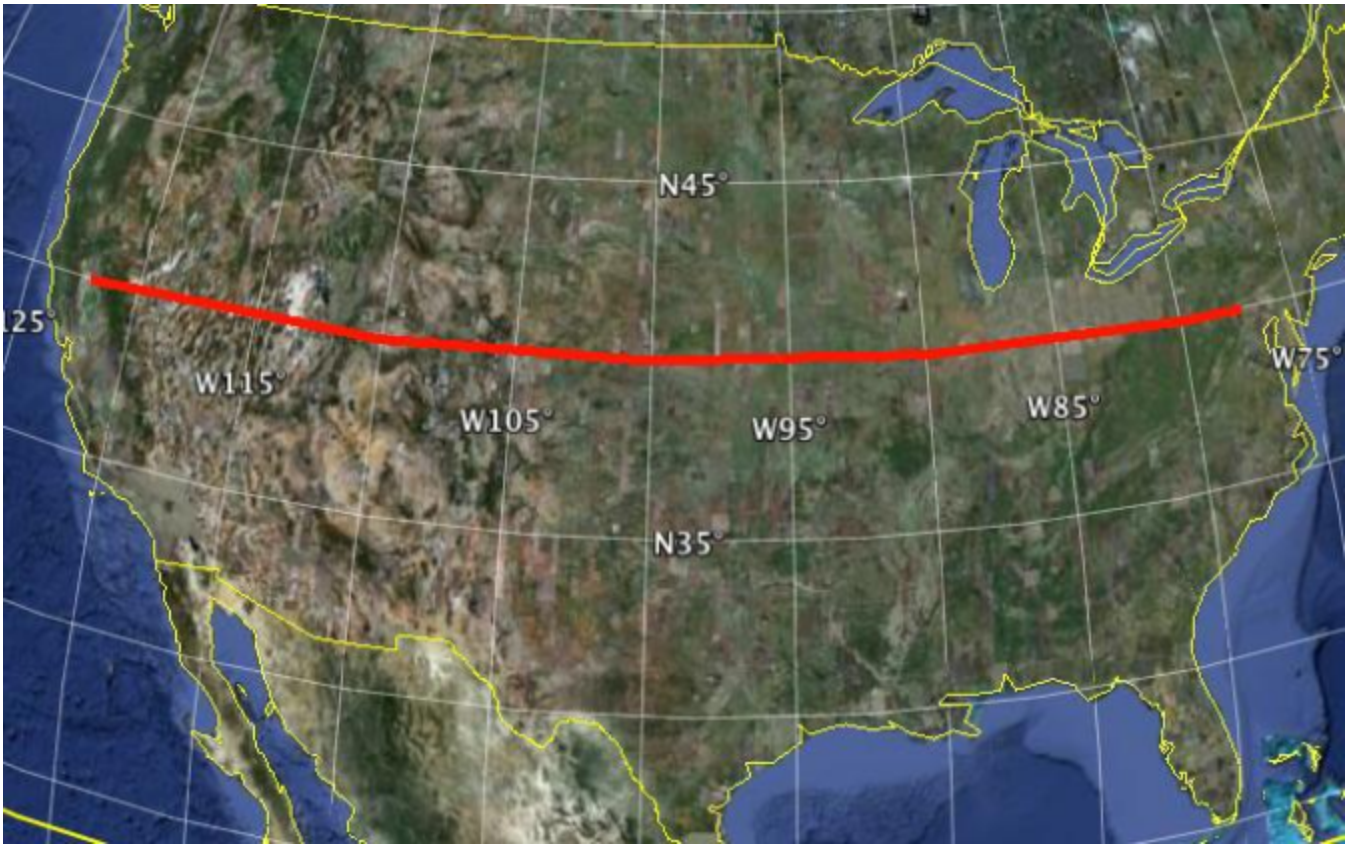
The figure below shows the line in the Cartesian and Geodetic coordinate systems. The Cartesian line is drawn in purple. The Geodetic line is drawn in red.



If it is desired that the Geodetic line follow the same path as the Cartesian line, more points can be specified, giving the line more density. The more points that are added, the closer the Geodetic Line will follow the Cartesian path. The following code expression shows an example of adding more points in the Geodetic coordinate system.

Adding Density

```xml
<Geometry>
    <Line>
        <Point>
            <PointLongitude>-122.1</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-110</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-100</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-90</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-80</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
        <Point>
            <PointLongitude>-77.35</PointLongitude>
            <PointLatitude>39.98</PointLatitude>
        </Point>
    </Line>
</Geometry>
```



Polygon

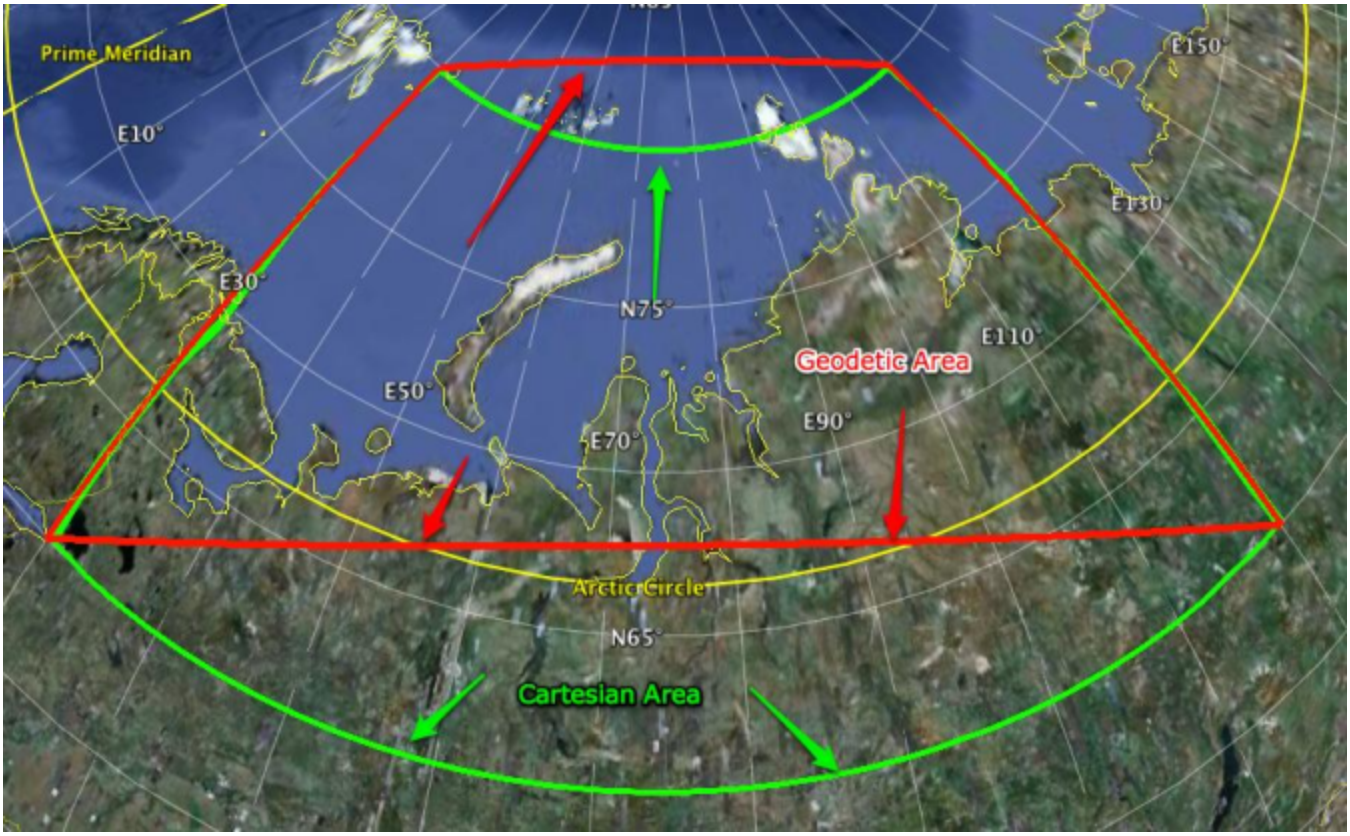
The CMR can receive, store, and search for spatial data representing one or more polygons with or without an area of exclusion. In the XML metadata, follow the syntax shown in the following code sample to define a spatial extent as a polygon:

Single Polygon

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>120</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>30</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>30</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>120</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```

A single polygon can have multiple holes, each represented by a single outer ring surrounding the area within it. In the Cartesian coordinate system, straight lines connect the points of the ring in a clockwise direction, which follows the order in which they are listed. In the Geodetic coordinate system, the points are connected using a great circle arc according to the shortest distance between two points. Remember that polygonal coverage cannot span more than half the earth in either system and may not cross the International Date Line or poles in the Cartesian coordinate system.

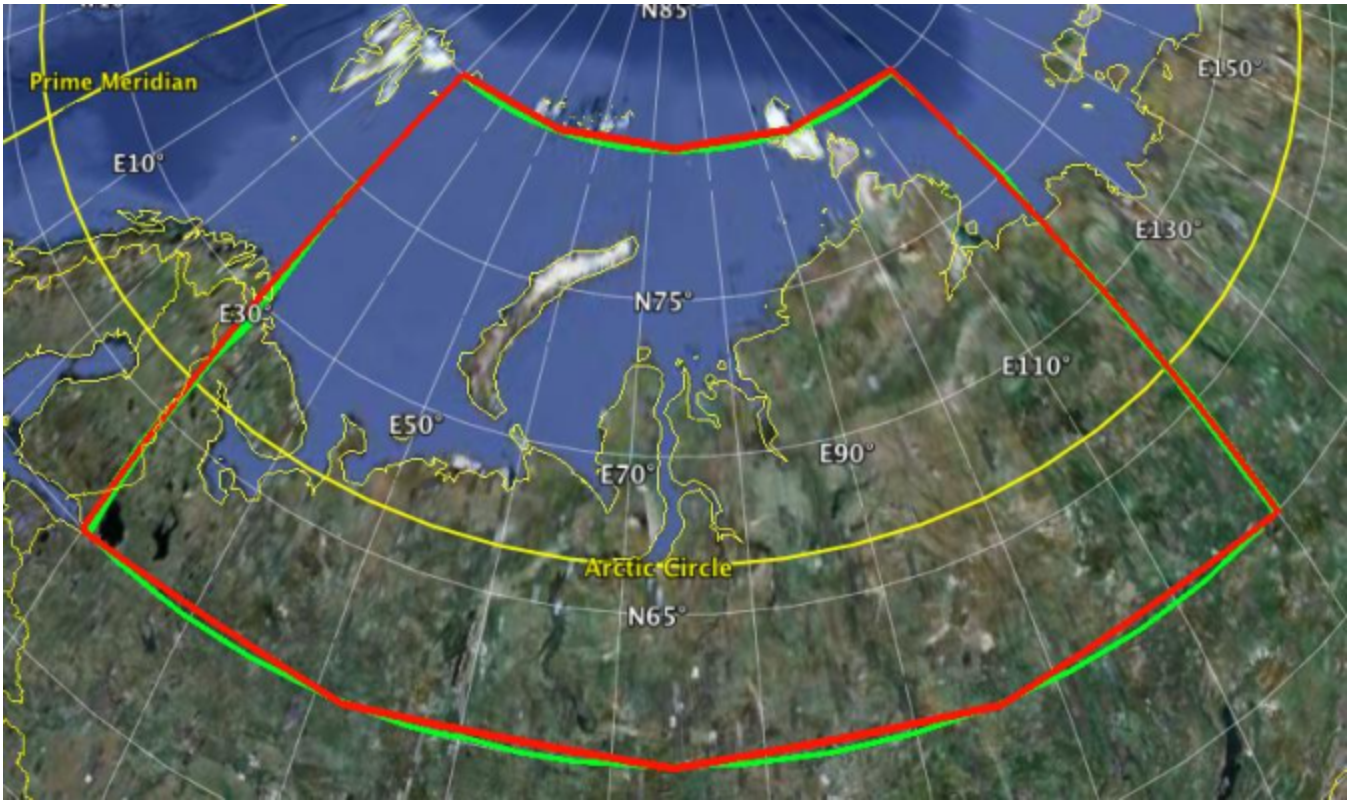
The previously included XML polygon example is shown in the following figures as it would be represented in both the Cartesian and Geodetic coordinate systems. The red outline shows the Geodetic polygon. The green outline shows the Cartesian polygon.



Applying a similar process of densification, as describe previously in conjunction with lines, the polygon can be densified into the following XML metadata and would appear as is shown below.

Densified Geodetic Polygon

```xml
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>120</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>96</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>74</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>52</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>30</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>30</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>52</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>74</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>96</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>120</PointLongitude>
                <PointLatitude>80</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```



The following sample metadata shows a single polygon with a hole in its spatial coverage. The figure below shows how this polygon will be represented in the Geodetic coordinate system.

Single Polygon with a Hole

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>-20.9342</PointLongitude>
                <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-42.3067</PointLongitude>
                <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-45.7985</PointLongitude>
                <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-24.8982</PointLongitude>
                <PointLatitude>6.1665</PointLatitude>
            </Point>
        </Boundary>
        <ExclusiveZone>
            <Boundary>
                <Point>
                    <PointLongitude>-22.9342</PointLongitude>
                    <PointLatitude>-5.9045</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-42.3067</PointLongitude>
                    <PointLatitude>-9.7732</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-34.7985</PointLongitude>
                    <PointLatitude>1.198</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-29.8982</PointLongitude>
                    <PointLatitude>3.1665</PointLatitude>
                </Point>
            </Boundary>
        </ExclusiveZone>
    </GPolygon>
</Geometry>
```

While a single polygon with a hole can have only one outer ring that represents the area surrounded within, it can have multiple inner rings that represent holes. All the rules, restrictions, and discussions for the outer ring in both coordinate systems apply to inner rings as well. An inner ring should be completely contained within the outer ring.

Form more information - see Polygon Support in CMR Search and Ingest Interfaces

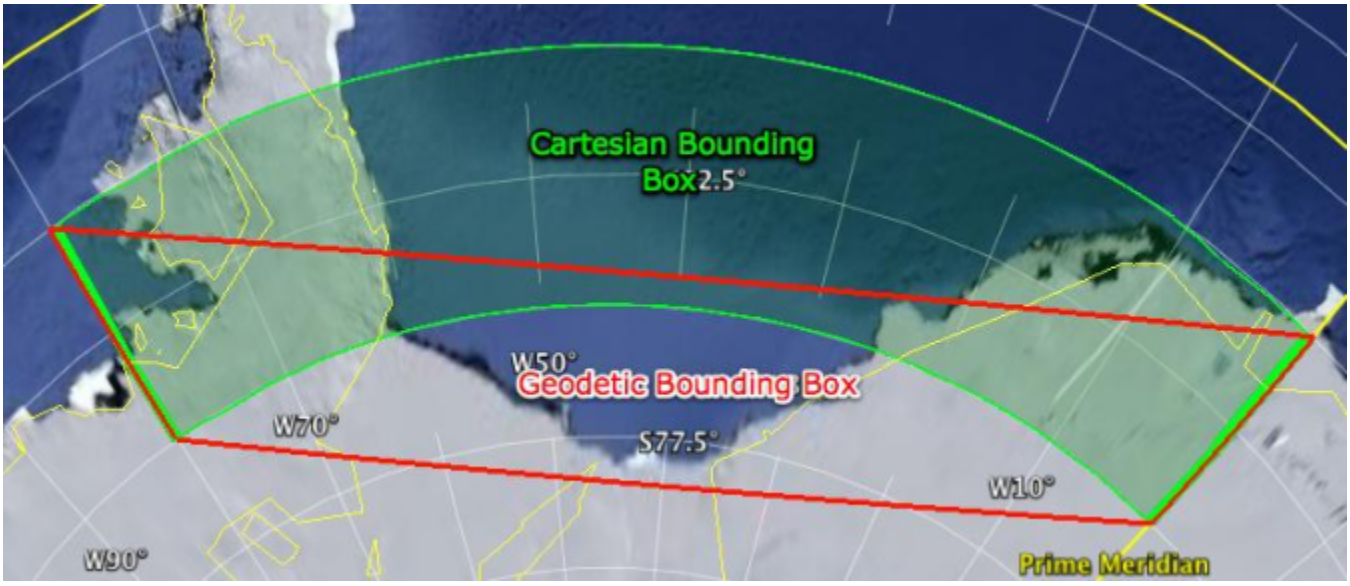
Bounding Box

In the Cartesian and Geodetic coordinate system, the CMR is capable of receiving, storing, and supporting the search on spatial data representing one or more bounding box. In the XML metadata, follow the syntax shown in the code sample below to define a spatial extent as a bounding box:

Bounding Rectangle (aka Bounding Box)

```
<Geometry>
    <BoundingRectangle>
        <WestBoundingCoordinate>-80</WestBoundingCoordinate>
        <NorthBoundingCoordinate>-70</NorthBoundingCoordinate>
        <EastBoundingCoordinate>0</EastBoundingCoordinate>
        <SouthBoundingCoordinate>-75</SouthBoundingCoordinate>
    </BoundingRectangle>
</Geometry>
```

The figure below represents the spatial area covered when applying the code shown above. The Geodetic Bounding Box is shown in red and the Carteisan Bounding Box is shown in green.



The CMR stores a bounding rectangle as a four-pointed polygon, subject to the specifications and constraints described for the polygon.

Invalid Spatial Representations

The following sections outline specific instances where the CMR will consider a spatial area invalid.

Polygon Points in Counter-Clockwise Order

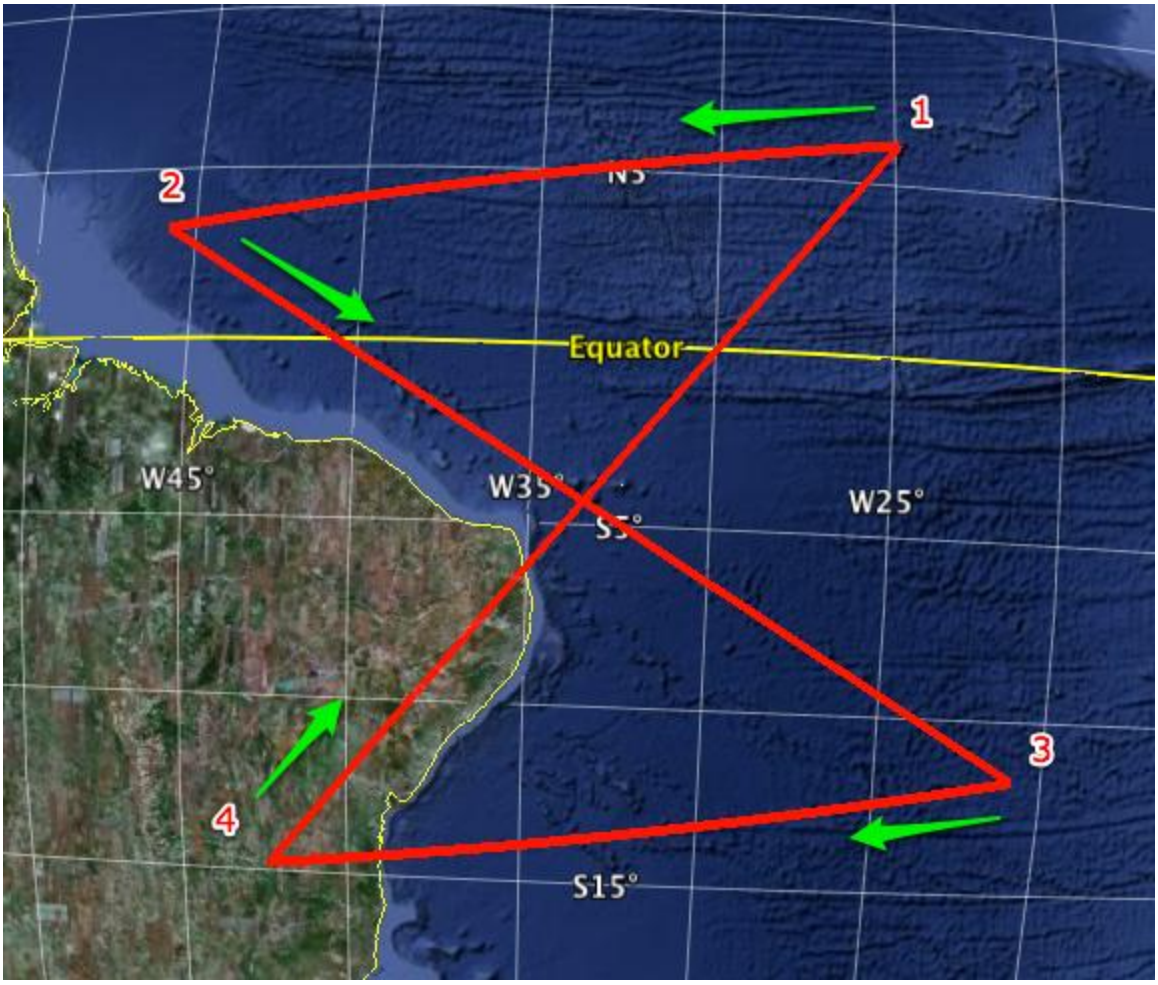
This spatial area expression is invalid in the Cartesian coordinate system. However, this same expression may be considered valid in the Geodetic coordinate system if the inversion does not cause the coverage to be more than one half of the Earth. Although the CMR may accept this polygon, the coverage will be interpreted very differently. The following example shows a polygon with points in reversed, counter-clockwise order.

Polygon with Points in Counter-Clockwise Order

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>170</PointLongitude>
                <PointLatitude>30</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-170</PointLongitude>
                <PointLatitude>30</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-170</PointLongitude>
                <PointLatitude>-30</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>170</PointLongitude>
                <PointLatitude>-30</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```

Twisted Polygon

In the case where the points in a polygon result in the line segments crossing, the CMR will reject this as invalid spatial data in both the Cartesian and the Geodetic coordinate systems. The figure below shows how this is represented on the Earth and the subsequent code example shows the same invalid area.

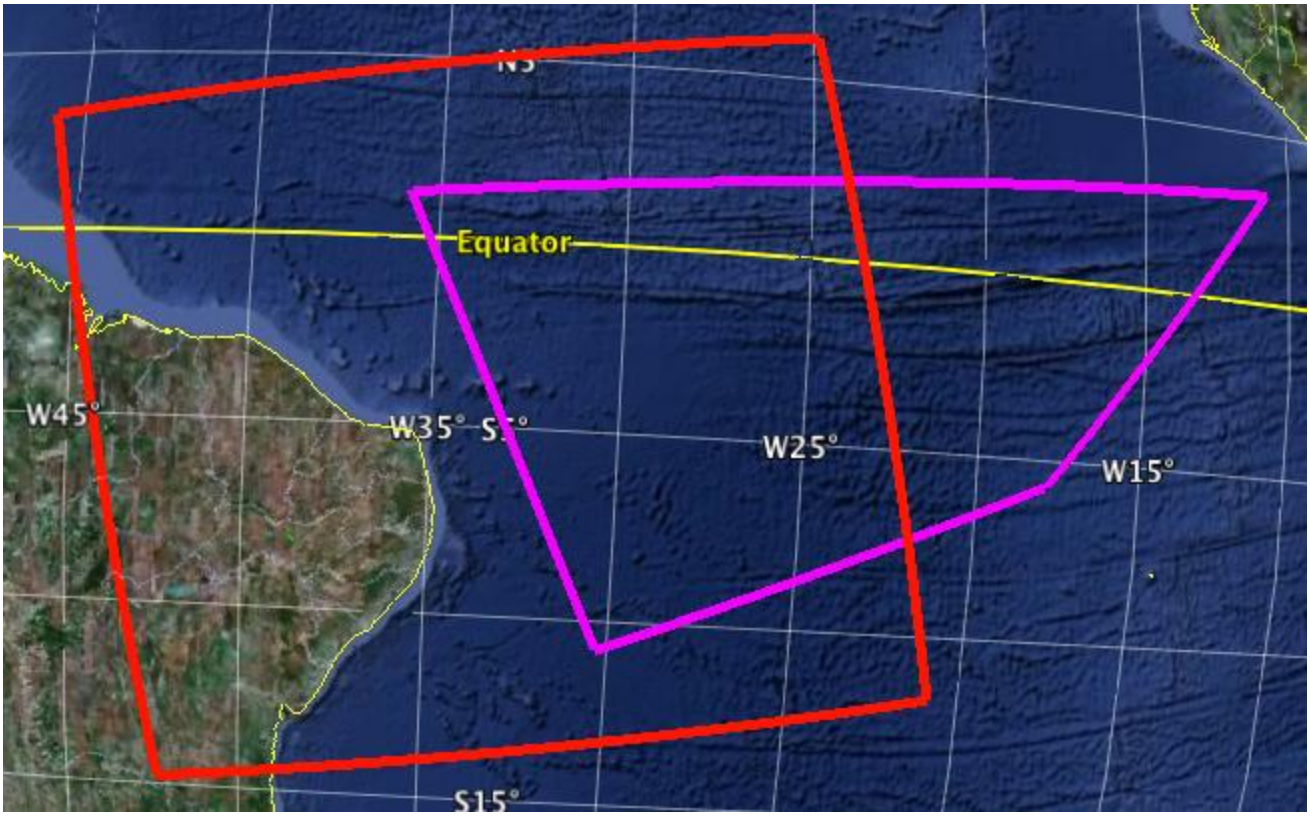


Twisted Polygon

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>-20.9342</PointLongitude>
                <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-42.3067</PointLongitude>
                <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-24.8982</PointLongitude>
                <PointLatitude>6.1665</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-45.7985</PointLongitude>
                <PointLatitude>3.198</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```

Hole Crosses over Outer Ring

In cases where an exclusion area (hole) intersects with the bounding box, the CMR will reject this as invalid spatial data in both the Cartesian and the Geodetic coordinate systems. The figure below shows how this is represented on the Earth and the subsequent code example shows the same invalid area.

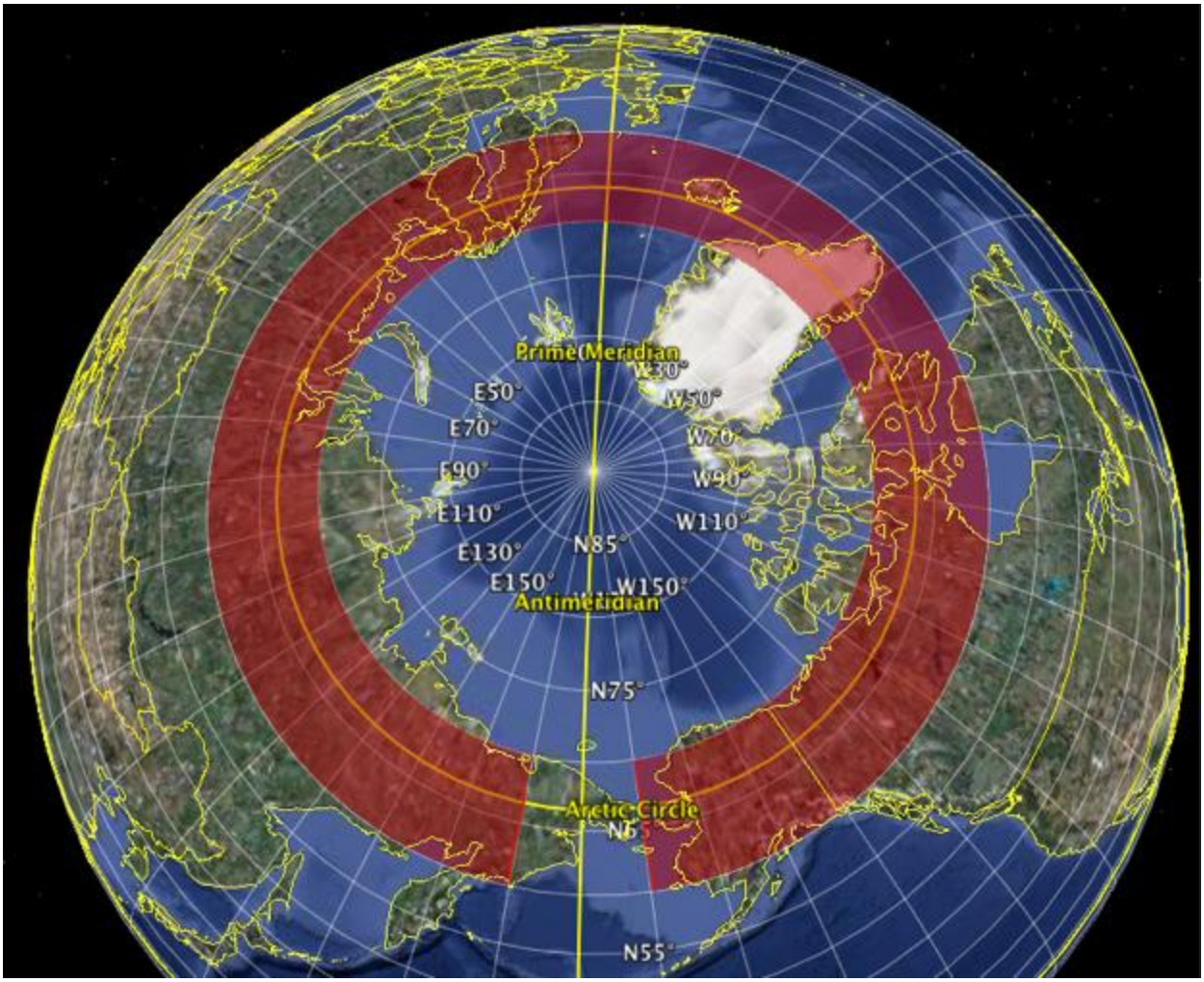


Hole Crosses over the Outer Ring

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>-20.9342</PointLongitude>
                <PointLatitude>-11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-42.3067</PointLongitude>
                <PointLatitude>-14.7732</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-45.7985</PointLongitude>
                <PointLatitude>3.198</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-24.8982</PointLongitude>
                <PointLatitude>6.1665</PointLatitude>
            </Point>
        </Boundary>
        <ExclusiveZone>
            <Boundary>
                <Point>
                    <PointLongitude>-17.9342</PointLongitude>
                    <PointLatitude>-5.7045</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-30.3067</PointLongitude>
                    <PointLatitude>-10.7732</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-35.7985</PointLongitude>
                    <PointLatitude>1.198</PointLatitude>
                </Point>
                <Point>
                    <PointLongitude>-10.8982</PointLongitude>
                    <PointLatitude>3.1665</PointLatitude>
                </Point>
            </Boundary>
        </ExclusiveZone>
    </GPolygon>
</Geometry>
```

Polygon Crosses International Date Line (antimeridian) or Pole

The CMR will not allow a Cartesian polygon to cross the International Date Line or the poles. The lines will be connected the long way around the earth as shown below. The figure deomstrates how this is represented on the Earth and the following code example shows the same invalid area. To correct this situation, the polygon should be divided into two or more spatial areas situated on both sides of the International Date Line and/or poles.



Polygon Crosses International Dateline

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>170</PointLongitude>
                <PointLatitude>70</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>170</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-170</PointLongitude>
                <PointLatitude>60</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-170</PointLongitude>
                <PointLatitude>70</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```

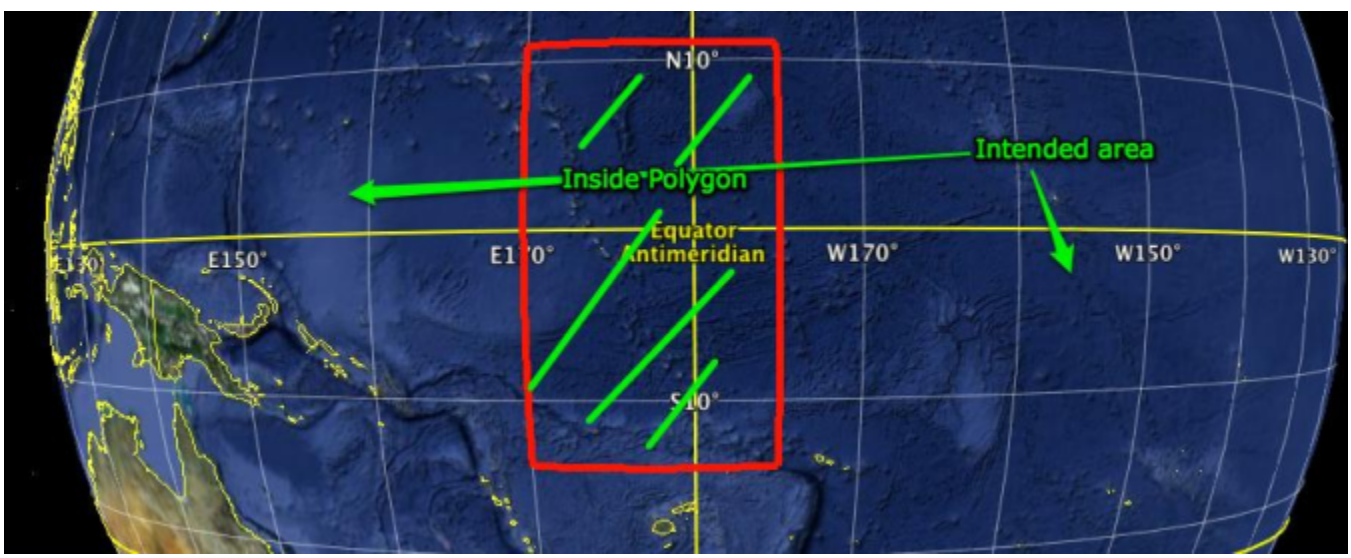
Inappropriate Point Density

Due to the nature of how spatial data is represented in the Geodetic coordinate system, it is possible that the CMR will validate a low density polygon, even though the spatial area does not represent what is expected. As an example, consider the following sample polygon metadata.

Incorrect Density

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>170.9342</PointLongitude>
                <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-175.3067</PointLongitude>
                <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-175.3067</PointLongitude>
                <PointLatitude>-13.198</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>170.9342</PointLongitude>
                <PointLatitude>-13.198</PointLatitude>
            </Point>
    </Boundary>
    </GPolygon>
</Geometry>
```

The expression above is valid spatial data in the Geodetic coordinate system. However, the spatial coverage area represented will be depicted as shown below:



While the area outside the left and right of the red polygon was the intended spatial representation, the area inside the red polygon is what will be used for spatial comparison. To represent this spatial coverage correctly, you must increase the point density by adding extra points. The sample below shows one way you might express these additional points, to represent this spatial coverage area correctly.

Correct Density

```
<Geometry>
    <GPolygon>
        <Boundary>
            <Point>
                <PointLongitude>170.9342</PointLongitude>
                <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>0.0</PointLongitude>
                <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-175.3067</PointLongitude>
                <PointLatitude>11.7045</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>-175.3067</PointLongitude>
                <PointLatitude>-13.198</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>0.0</PointLongitude>
                <PointLatitude>-13.198</PointLatitude>
            </Point>
            <Point>
                <PointLongitude>170.9342</PointLongitude>
                <PointLatitude>-13.198</PointLatitude>
            </Point>
        </Boundary>
    </GPolygon>
</Geometry>
```

Tolerance

The CMR makes use of resolution parameter settings to associate a level of precision with spatial data by using evaluation parameters when validating spatial data input. Cartesian tolerance is specified as fractions of a degree and Geodetic tolerance is specified in meters. If the Cartesian tolerance is 0.05 for both latitude and longitude, and if the distance between two points is less than 0.05 degrees for both longitude and latitude, then those two points are considered the same point. In this situation, the spatial expression is invalid because the CMR spatial constructs require each point to have a unique spatial location.

The CMR defaults for tolerance is:

- Cartesian Tolerance: .0001 degrees
- Geodetic Tolerance: 5 centimeters

Orbit Data

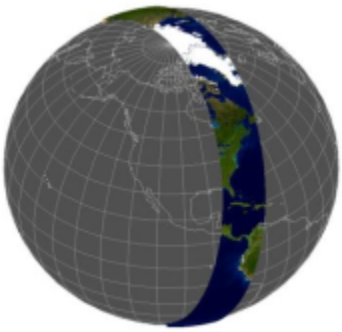
Orbit searching is by far the most accurate way to search for level 0-2 orbital swath data. However, orbital mechanics is very complex, and the most well known orbit model - the NORAD Propagator - is extraordinarily complicated. The NORAD Propagator is designed to work with a wide range of possible orbits, from circular to highly elliptical, and consequently requires a lot of information about the orbit to model it thoroughly and correctly.

On the contrary, Earth science data gathering satellites have significantly less variety of orbits than the multitude designed to work with the

NORAD Propagator. Generally, the earth science community prefers global coverage, with a constant field of view, at the same time every day. For this reason, most earth science satellites are in a sun-synchronous, near-polar orbit. Even missions that are not interested in global coverage, e.g., the Tropical Rainfall Measuring Mission (TRMM), still prefer a constant field of view, so the coverage of the sensor is at a constant resolution. For this reason, ALL earth science satellites are in circular orbits.

The Backtrack Orbit Search Algorithm, designed and developed by Ross Swick, exploits this fact to simplify the orbit model by modeling an orbit as a great circle under which the Earth rotates. This reduces the number of orbital elements required for the model from 22 to 3. Moreover, the NORAD Propagator is designed to predict future orbits based on current status, and consequently must be reinitialized periodically to correct for cumulative error as the model spins forward. As the name implies, Backtrack spins the orbit backwards, and in practice spins backwards at most one orbit, so there is no cumulative error.

Orbit granules may not be ingested unless the parent collections have orbit parameters defined.



Orbit Swath

- Three parameters are required to define an orbit:

1. Instrument swath width (in kilometers)
2. Satellite declination or inclination (in degrees)
3. Satellite period (in minutes)

For more information on Backtrack, please see http://geospatialmethods.org/bosa/.

- Three parameters are required to represent orbit data:

1. Equatorial crossing longitude (in degrees)
2. Start circular latitude (or start latitude and start direction)
3. End circular latitude (or end latitude and end direction)

How Data Providers Configure Orbit Data

Add orbit data to Granule Metadata

```
<Spatial>
    <HorizontalSpatialDomain>
        <Orbit>
            <AscendingCrossing>160.14462465545338</AscendingCrossing>
            <StartLat>69.021242</StartLat>
            <StartDirection>D</StartDirection>
            <EndLat>-68.995831</EndLat>
            <EndDirection>A</EndDirection>
        </Orbit>
    </HorizontalSpatialDomain>
</Spatial>
```

Add orbit parameters to Collection Metadata

```
<Spatial>
    <SpatialCoverageType>Horizontal</SpatialCoverageType>
    <OrbitParameters>
        <SwathWidth>400</SwathWidth>
        <Period>98.88</Period>
        <InclinationAngle>98.2</InclinationAngle>
    </OrbitParameters>
    <GranuleSpatialRepresentation>ORBIT</GranuleSpatialRepresentation>
</Spatial>
```

Global Data

The CMR does not support an explicit "global" designation for a collection's or granule's spatial representation. Instead, a Cartesian bounding box with corners covering the entire earth will be interpreted as "global" within the CMR. Due to the usage of a Cartesian coordinate system, the collections and granules will be spatially searchable and be discovered with all spatial areas. The CMR will also facilitate "global only" searching which will only discover metadata items with the spatial geometry shown below.

Add global data to Granule Metadata

```
<Spatial>
    <HorizontalSpatialDomain>
        <Geometry>
            <BoundingRectangle>
                <WestBoundingCoordinate>-180</WestBoundingCoordinate>
                <NorthBoundingCoordinate>90</NorthBoundingCoordinate>
                <EastBoundingCoordinate>180</EastBoundingCoordinate>
                <SouthBoundingCoordinate>-90</SouthBoundingCoordinate>
            </BoundingRectangle>
        </Geometry>
    </HorizontalSpatialDomain>
</Spatial>
```

Add global data to Collection Metadata

```
<Spatial>
    <HorizontalSpatialDomain>
        <Geometry>
            <BoundingRectangle>
                <WestBoundingCoordinate>-180</WestBoundingCoordinate>
                <NorthBoundingCoordinate>90</NorthBoundingCoordinate>
                <EastBoundingCoordinate>180</EastBoundingCoordinate>
                <SouthBoundingCoordinate>-90</SouthBoundingCoordinate>
            </BoundingRectangle>
        </Geometry>
    </HorizontalSpatialDomain>
</Spatial>
```

Tiling Identification System (Two-Dimensional Coordinate System)

A tiling identification system may be used to facilitate discovery via an alternate mechanism. Examples of two-dimensional coordinate system values are path/row for Worldwide Reference System (WRS) data and Moderate Resolution Imaging Spectroradiometer (MODIS) tile IDs. A collection must define all tiling identification systems that granules in that collection may use. CMR Ingest will validate that a granule's Tiling Identification System element references a valid Tiling system defined in the collection. If the collection's definition for the system defines axis minimum and maximum values, CMR Ingest will validate the granule's metadata against these constraints.

Tiling Identification System Axis Labels

The CMR data model does not allow providers to configure axis labels for each Tiling Identification System. This restriction was implemented to avoid creating significant complications for CMR Client developers. Instead, CMR identified the following unique coordinate systems, with the axis labels included below. This listing of labels is recommended for CMR Client Partners. If additional coordinate systems are utilized, Data Partners should coordinate with CMR Operations to ensure that the labeling recommendations are correct.

Coordinate System	X Axis	Y Axis
WRS-1	Path	Row
WRS-2	Path	Row
MISR	Path	Block
MODIS Tile	Horizontal Tile	Vertical Tile
Calipso	Orbit	Path

Collection Level

The Tiling Identification System that exists at the collection level contains the following information:

- **TwoDCoordinateSystemName** – The identifying name of the coordinate system
- **Start/EndCoordinate1** – The X' axis coordinate minimum and maximum range values
- **Start/EndCoordinate2** – The Y' axis coordinate minimum and maximum range values

The minimum and maximum range values are not required for both the 'X' and 'Y' axes. This allows for finite, infite (in both directions), or no range validation for a specific coordinate system axis. The following example provides a finite range for Coordinate1 and infinite ending for Coordinate 2.

Collection Tiling Identification System Coordinates Using ECHO 10 Specification

```
<TwoDCoordinateSystem>
    <TwoDCoordinateSystemName>WRS2</TwoDCoordinateSystemName>
        <Coordinate1>
            <MinimumValue>0</MinimumValue>
            <MaximumValue>38</MaximumValue>
        <Coordinate1/>
        <Coordinate2>
            <MinimumValue>0</MinimumValue>
        <Coordinate2/>
</TwoDCoordinateSystem>
```

Granule Level

The Tiling Identification System that exists at the granule level contains the following information:

- **Start/EndCoordinate1** – The 'X' axis start and end (not required) coordinate values
- **Start/EndCoordinate2** – The 'Y' axis start and end (not required) coordinate values
- **TwoDCoordinateSystemName** – The identifying name of the coordinate system

Granule Tiling Identification System Coordinates Using ECHO10 Specification

```
<TwoDCoordinateSystem>
    <StartCoordinate1>21</StartCoordinate1>
    <StartCoordinate2>29</StartCoordinate2>
    <EndCoordinate2>33</EndCoordinate2>
    <TwoDCoordinateSystemName>WRS2</TwoDCoordinateSystemName>
</TwoDCoordinateSystem>
```

Acronyms

Acronym	Expansion
ACL	Access Control List
API	Application Programming Interface
AQL	Alternative Query Language
ASTER	Advanced Spaceborne Thermal Emission and Reflection Radiometer
CMR	Common Metadata Repository
COTS	Commercial Off The Shelf
DAAC	Distributed Active Archive Center
DB	DataBase
DTD	Document Type Definition
ECHO	EOS Clearinghouse
ECS	EOSDIS Core System

EDC	EROS Data Center
EMD	EOSDIS Maintenance and Development
EOS	Earth Observing System
EOSDIS	EOS Data and Information System
EROS	Earth Resources Observation Systems
ESDIS	Earth Science Data and Information System
ESIP	Earth Science Information Partner
FTP	File Transfer Protocol
GCMD	Global Change Master Directory
GES DAAC	GSFC Earth Sciences DAAC
GHRC	Global Hydrology Resource Center
GIS	Geographic Information System
GML	Geography Markup Language
GMT	Greenwich Mean Time
GSFC	Goddard Space Flight Center
GUI	Graphical User Interface
GUID	Globally Unique Identifier
IIMS	Independent Information Management Subsystem
LAADS	Level 1 and Atmosphere Archive and Distribution System
LP DAAC	Land Processes DAAC
MISR	Multiangle Imaging SpectroRadiometer
MMT	Metadata Management Tool
MODIS	Moderate Resolution Imaging Spectroradiometer
NASA	National Aeronautics and Space Administration
NSIDC DAAC	National Snow and Ice Data Center DAAC
ODL	Object Description Language
OGC	OpenGIS Consortium
ORNL DAAC	Oak Ridge National Laboratory DAAC
PGE	Product Generation Executives
PO.DAAC	Physical Oceanography DAAC
PSA	Product Specific Attribute
QA	Quality Assurance
SEDAC	Socioeconomic Data and Applications Center
SIT	Software Integration Test
SOAP	Simple Object Access Protocol
SSC	Stennis Space Center
SSL	Secure Sockets Layer

UAT	User Acceptance Test
UDDI	Universal Description, Discovery, and Integration
UI	User Interface
UR	Universal Reference
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Universal Time, Coordinated (also called GMT/UTC)
WGS	World Geodetic System
WRS	Worldwide Reference System
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XSLT	eXtensible Style Language Transformation