ECHO 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 Rectangle
- Invalid Spatial Representations
  - Polygon Points in Counter-Clockwise Order
  - Twisted Polygon
  - Hole Crosses over Outer Ring
  - Polygon Crosses Ante meridian or Pole
  - Inappropriate Point Density
  - Tolerance
- Orbit Data
  - Backtrack Orbit Model
  - Orbit Data Representation
  - How Data Providers Configure Orbit Data
- Global Data

This chapter describes the terms and concepts relevant to ECHO's spatial model. Examples are provided for each supported spatial representation included 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. ECHO 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 ECHO system accepts spatial data represented in the Cartesian and Geodetic coordinate systems. ECHO 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 size and projection of the original data of the spatial area covered. You may not combine spatial types for granules within the same collection as the representations are mutually exclusive.
The ECHO implementation of the Cartesian coordinate system accepts spatial data types of Point, Line, Bounding Rectangle and Polygon.
The ECHO implementation of the Geodetic With coordinate system (World Geodetic System 84) accepts spatial data types of Point, Line, Bounding Rectangle, and Polygon.

Supported Spatial Data Types

Spatial Data Type	Cartesian	Geodetic	Orbit	Guidelines and restrictions
Point	🙂	🙂	🙂	
Bounding Box	🙂	🙂	☹	ECHO stores bounding rectangle 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 ECHO system will not manipulate any of the spatial input metadata. You are responsible for the correctness and integrity of your spatial metadata. To prepare your metadata such that ECHO can support the correct searching for your data, please observe the information in the following sections.

Collection & Granule Spatial Relationships

The following items are some guidelines that should be considered when generating spatial metadata within ECHO collection and granule records.

- Each collection may specific 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 also specify specific 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 according to the associate rules with the utilized coordinate system.

Geometry Representations

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

Coordinate Systems

Cartesian Coordinate System

The Cartesian coordinate system is a flattened coordinate system with longitude ranged from –180 to 180 degrees and latitude ranged from –90 to 90 degrees. The projected map is flattened and open along Antemeridian 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 ante meridian (unless it is a bounding box) and/or Poles.
- Two vertices will be connected with a straight line.

Geodetic Coordinate System

The Geodetic coordinate system is defined in angular (latitude and longitude) and is defined relative to spherical polar coordinate and Earth Geodetic datum. The ECHO implementation of the Geodetic coordinate system follows OGC standards, which are defined at http://www.opengeospatial.org/. The Geodetic coordinate ECHO chose to support is World Geodetic System 84 (WGS 84). Please be aware of the following Geodetic coordinate system constraints:

- The ECHO Geodetic model uses the great circle distance to connect two vertices to construct 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 Antemeridian 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 rectangles, and/or polygons can be included to define the spatial extent of your data.

Point

ECHO can receive and store spatial data representing one or more points. ECHO also supports searching for spatial data representing one or more points. In the XML metadata, follow the syntax 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

ECHO can receive and store spatial data representing one or more lines. ECHO also supports searching 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

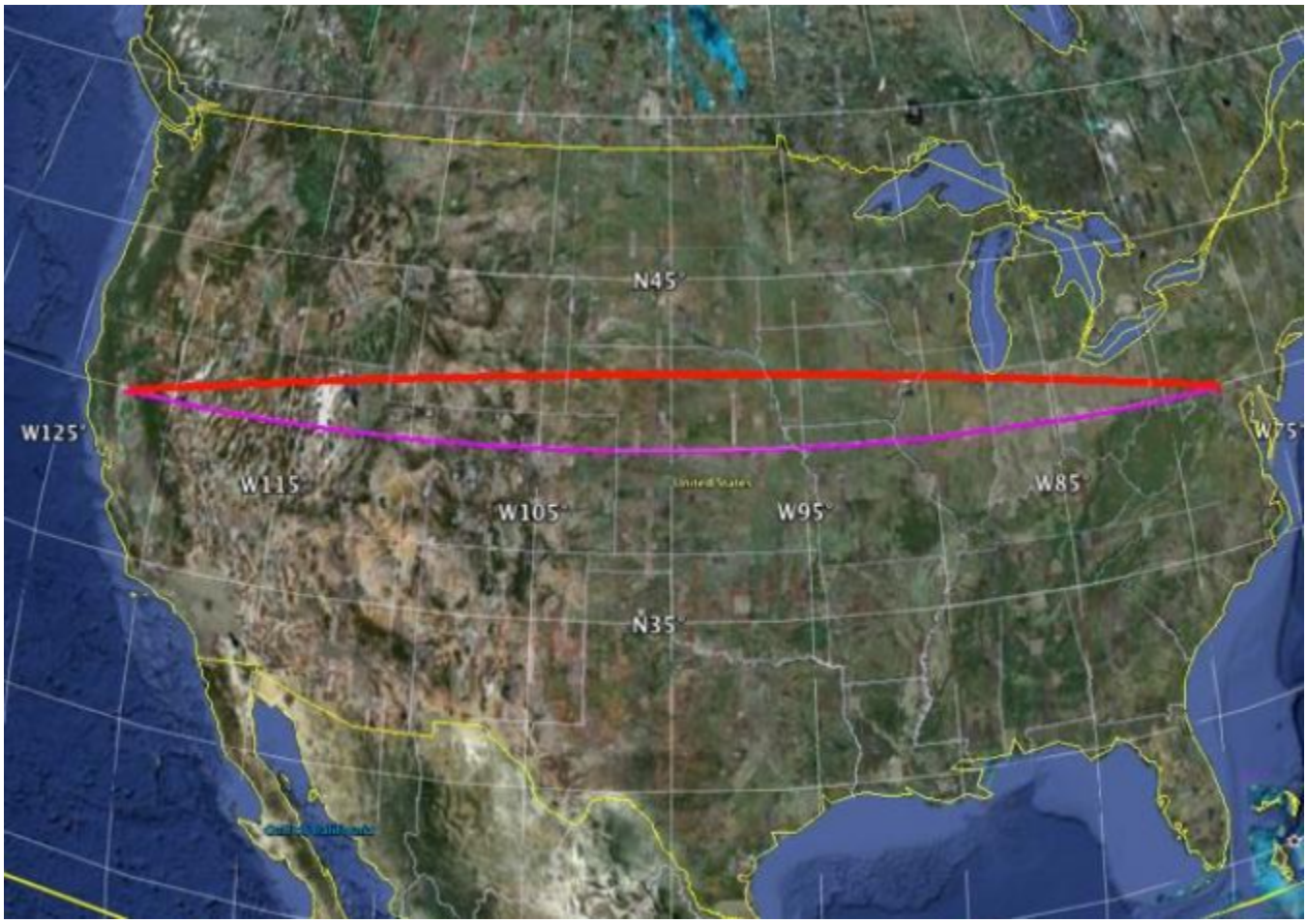
```
<Geometry>
 <Line>
  <Point>
   <PointLongitude>-123.948</PointLongitude>
   <PointLatitude>45.0664</PointLatitude>
  </Point>
  <Point>
   <PointLongitude>-133.546</PointLongitude>
   <PointLatitude>45.0664</PointLatitude>
  </Point>
 </Line>
 <Line>
  <Point>
   <PointLongitude>-123.948</PointLongitude>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

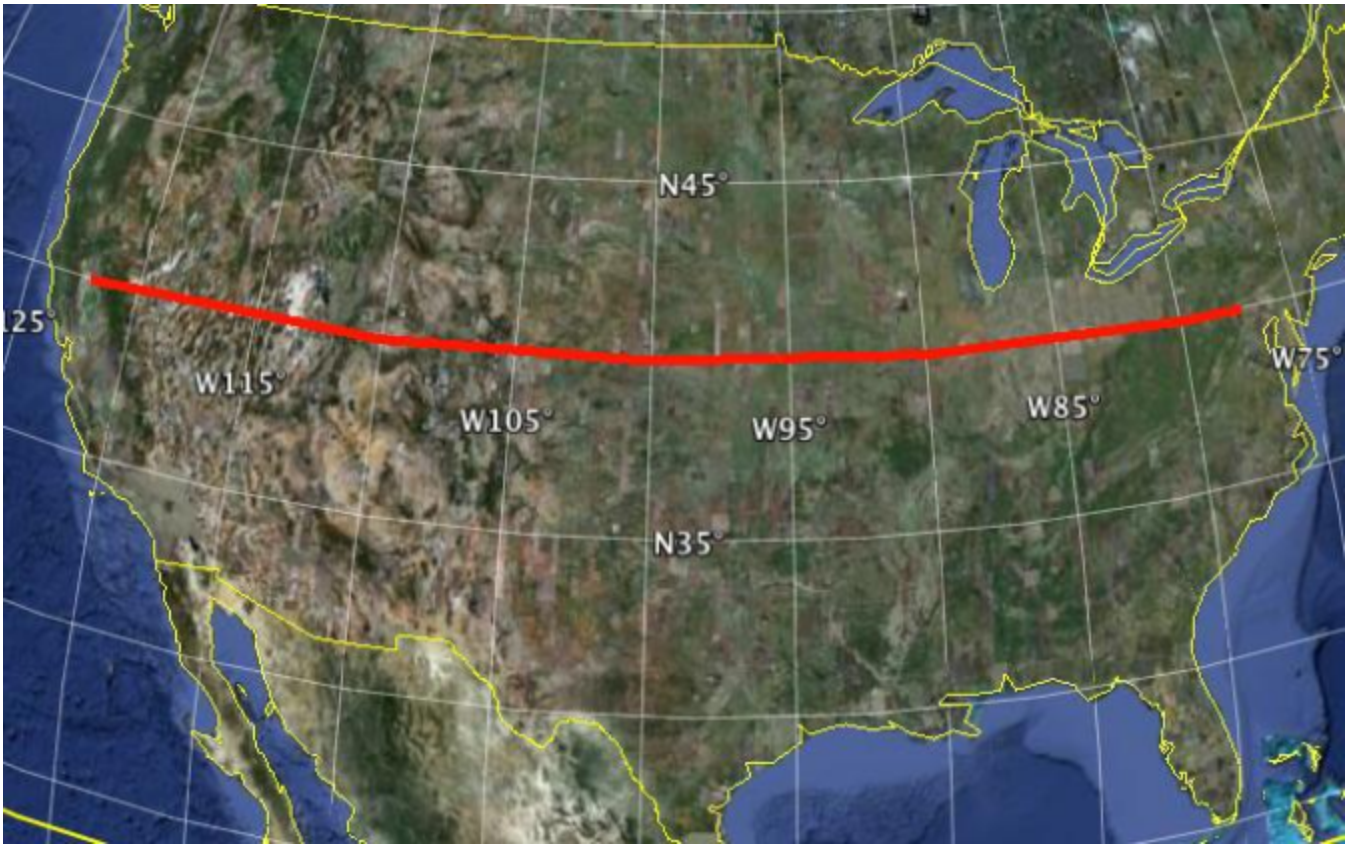
The figure below shows the line in the Cartesian and Geodetic coordinate systems. The Cartesian line is drawn in purple. The Geodetic line is drawn in red.



If it is desired that the Geodetic line follow the same path as the Cartesian line more points can be 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

```
<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

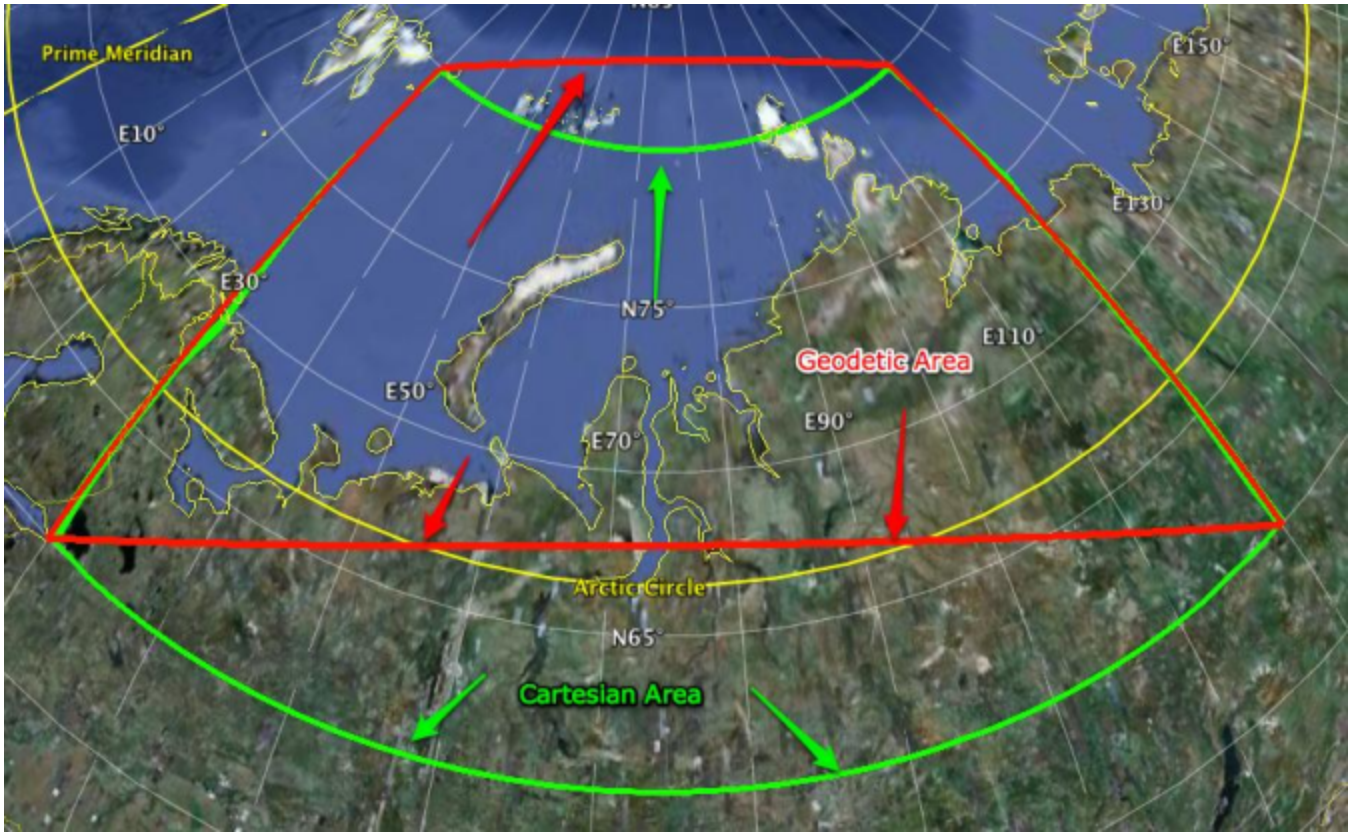
ECHO can receive and store spatial data representing one or more polygons with or without an areas of exclusion. ECHO also supports searching for spatial data representing any of these polygons. In the XML metadata, follow the syntax shown in the following code sample to define a spatial extent as a 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 the order in which they are listed, which must always be in clockwise order. 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 and may not cross the ante meridian and/or poles in the Cartesian coordinate system. A polygon cannot represent more than one half of the earth in the Geodetic coordinate system.

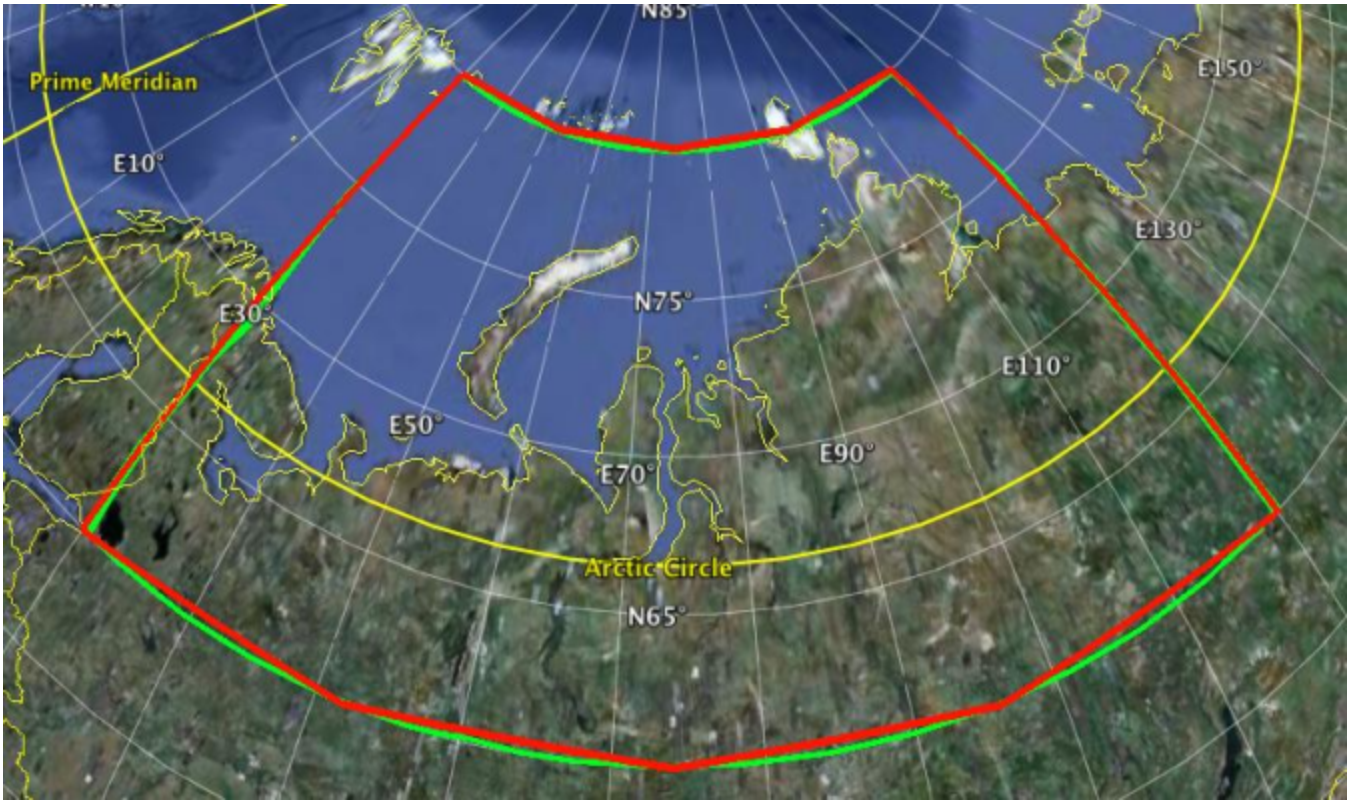
The previously included XML polygon example is shown in the following figures as it would be represented in both the Cartesian and Geodetic coordinate systems. The red outline shows the Geodetic polygon. The green outline shows the Cartesian polygon.



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

Densified Geodetic Polygon

```
<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.

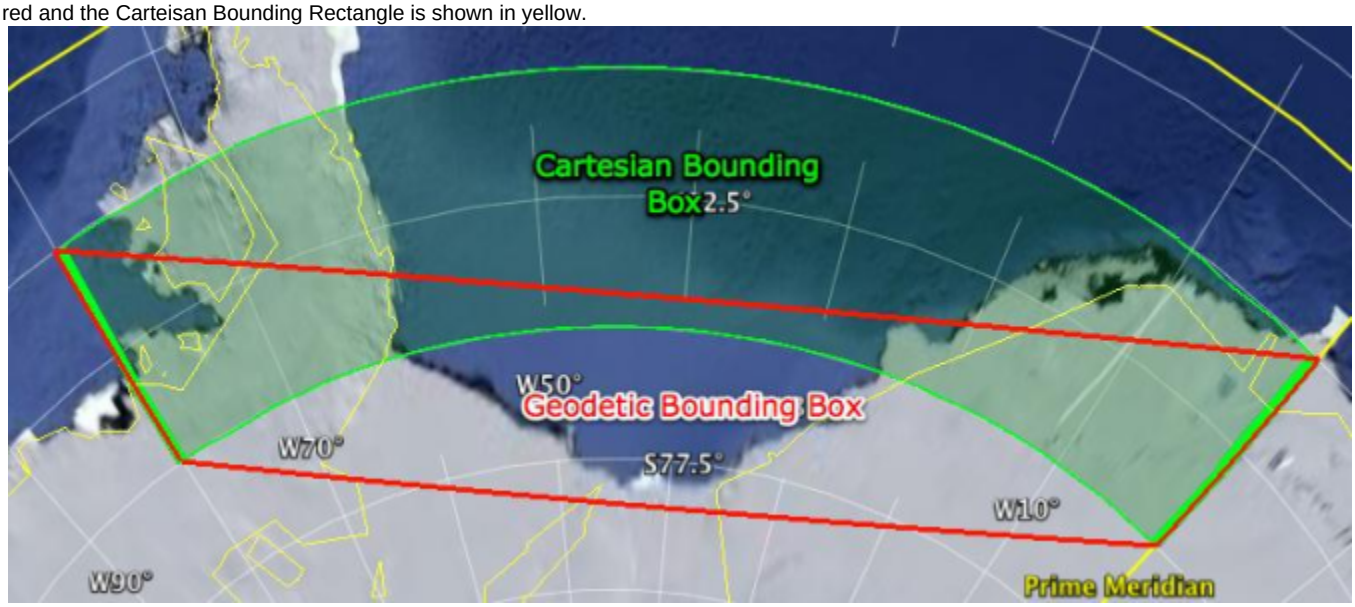
Bounding Rectangle

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

Bounding Rectangle

```
<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 Rectangle is shown in red and the Carteisan Bounding Rectangle is shown in yellow.



ECHO 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 ECHO will consider a spatial area invalid.

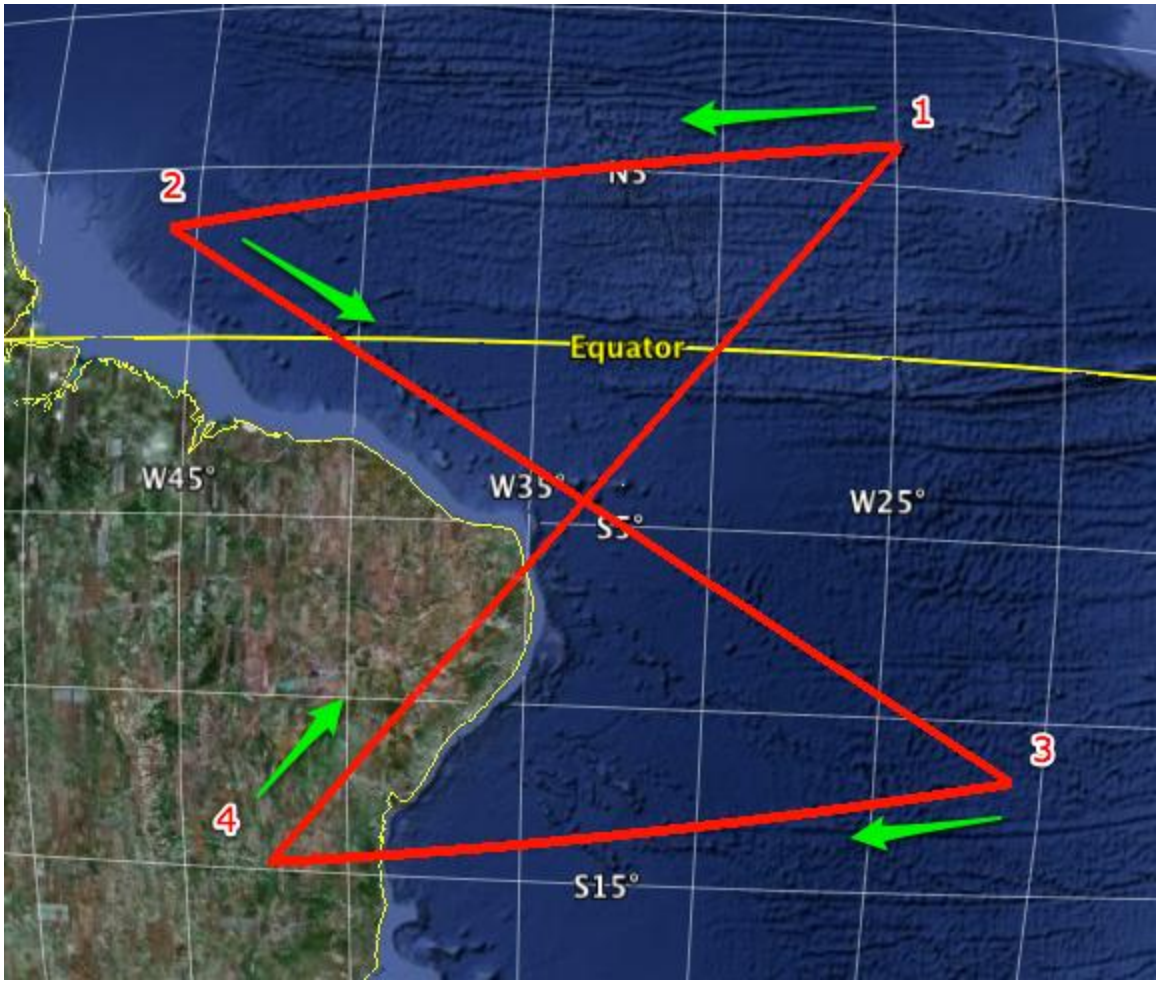
Polygon Points in Counter-Clockwise Order

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 ECHO may accept this polygon, the coverage will be interpreted very differently. The following sample 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

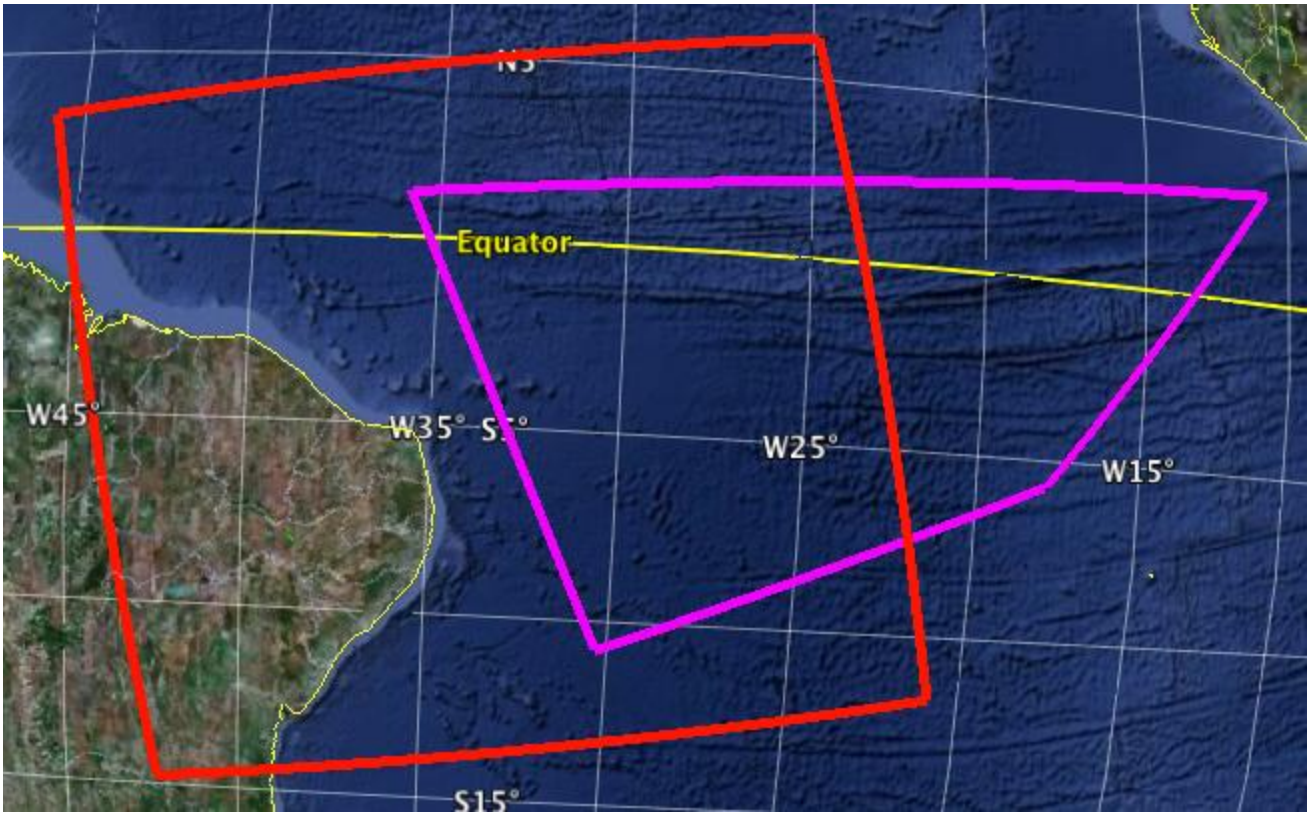


In the case where the points in a polygon cause the line segments to cross, ECHO will reject this as invalid spatial data in both the Cartesian and the Geodetic coordinate systems. The figure above shows how this is represented on the Earth and the following code sample shows the same invalid area.

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

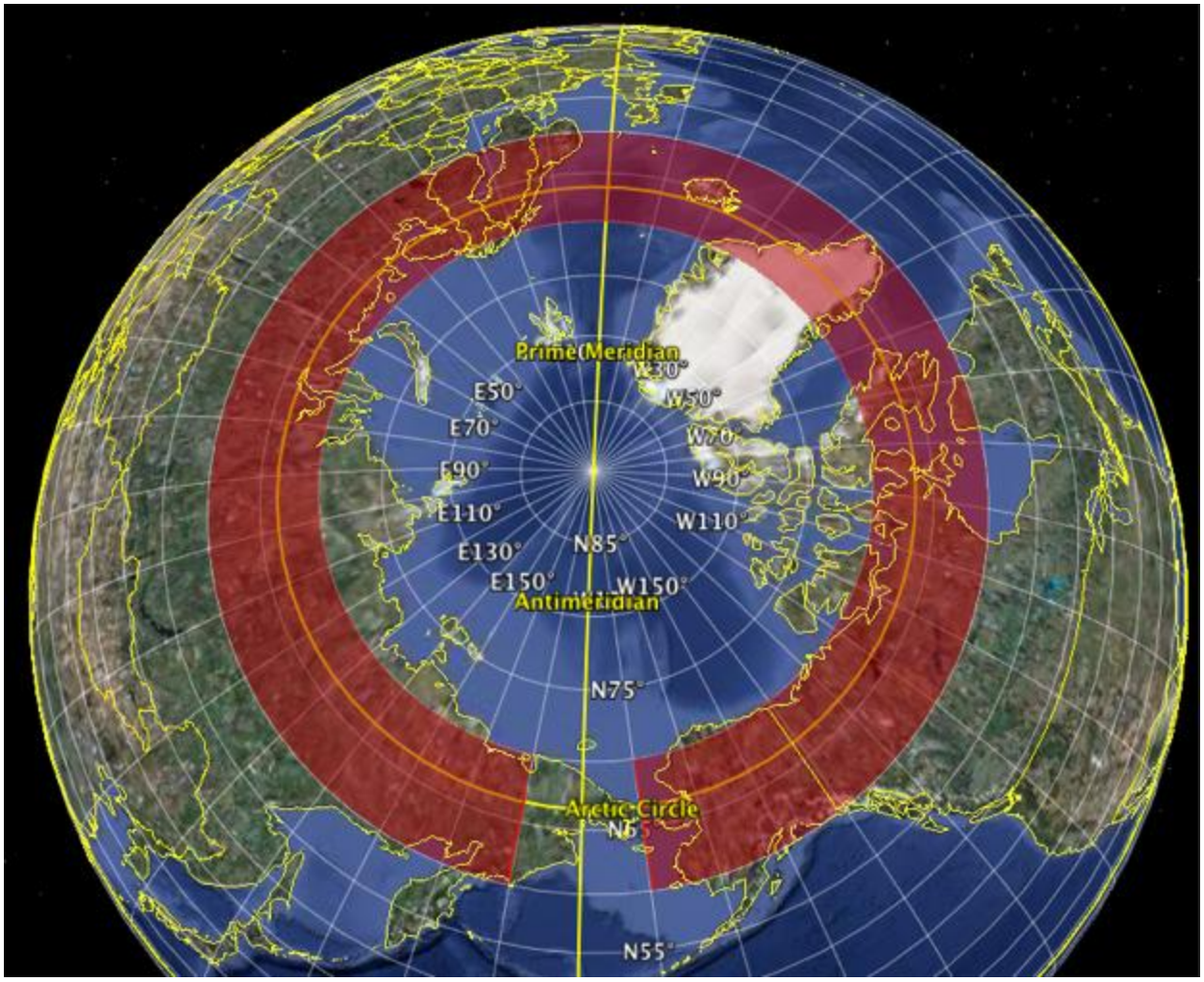


In the case where the an exclusion area (hole) intersects with the bounding polygon, ECHO will reject this as invalid spatial data in both the Cartesian and the Geodetic coordinate systems. The figure above shows how this is represented on the Earth and the following code sample shows the same invalid 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 Ante meridian or Pole



ECHO will not allow a Cartesian polygon to cross the international dateline or the poles. The lines will be connected the long way around the earth as shown above. The figure above shows how this is represented on the Earth and the following code sample shows the same invalid area. To correct this situation, the polygon should be split into two or more spatial areas that are split at the Antemeridian and/or poles as needed.

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>
```

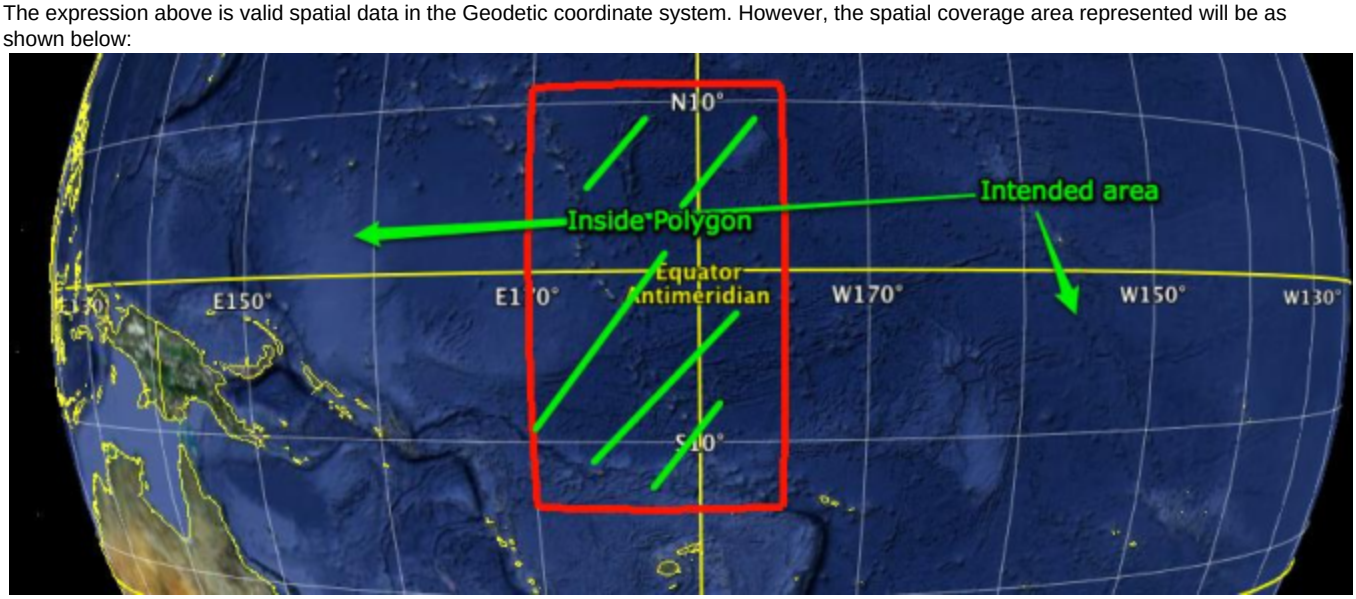
Inappropriate Point Density

Due to the nature of how spatial data is represented in the Geodetic coordinate system, it is possible that ECHO 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 as shown below:



The area shown outside the left and right of the red polygon may have been expected, but 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

ECHO makes use of resolution parameter settings to associate a level of precision with spatial data. ECHO uses these parameters as 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 ECHO spatial constructs require each point to have a unique spatial location.
ECHO defaults for tolerance is:

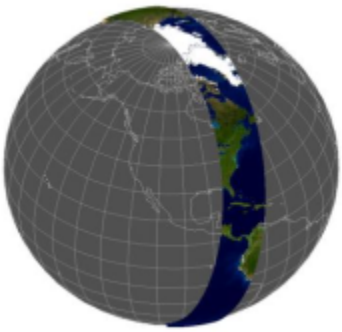
- Cartesian Tolerance: .0001 degrees
- Geodetic Tolerance: 5 centimeters

Orbit Data

Orbit searching is by far the most accurate way to search for level 0-2 orbital swath data. Unfortunately orbital mechanics is a quite difficult field, and the most well known orbit model, the NORAD Propagator, is quite complex. The NORAD Propagator is designed to work with a wide range of possible orbits, from circular to extremely elliptical, and consequently requires quite a bit of information about the orbit to model it well.
To facilitate earth science, the orbits of satellites gathering earth science data are quite restricted compared to the variety of orbits the NORAD Propagator is designed to work with. Generally, the earth science community would like 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), are still interested in having 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 three. 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.
For more information on Backtrack, please see http://geospatialmethods.org/bosa/.

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



Orbit Swath

Backtrack Orbit Model

Three parameters to define an orbit:

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

Orbit Data Representation

Three parameters 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

ECHO 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 ECHO. Due to the usage of a Cartesian coordinate system, the collections and granules will be spatially searchable and be discovered with all spatial areas. ECHO will also facilitate global only searching which will only discover metadata items with the spatial geometry shown below.

Add global datato 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>
 <SpatialCoverageType>Horizontal</SpatialCoverageType>
 <HorizontalSpatialDomain>
  <Geometry>
   <CoordinateSystem>CARTESIAN</CoordinateSystem>
   <BoundingRectangle>
    <WestBoundingCoordinate>-180</WestBoundingCoordinate>
    <NorthBoundingCoordinate>90</NorthBoundingCoordinate>
    <EastBoundingCoordinate>180</EastBoundingCoordinate>
    <SouthBoundingCoordinate>-90</SouthBoundingCoordinate>
   </BoundingRectangle>
  </Geometry>
 </HorizontalSpatialDomain>
 <GranuleSpatialRepresentation>CARTESIAN</GranuleSpatialRepresentation>
</Spatial>
```