GeoEvent Server – Implementing a 'Snap To' Process

Contents

GeoEvent Server – Implementing a 'Snap To' Process

Introduction

When a Real-Time GIS system receives data from a GPS receiver there is always some sort of accuracy that goes along with that point. Frequently, the GPS locations are vehicles traveling on roads (e.g. snow plows) where it is important to know which road that vehicle is on. However, with GPS accuracies in the 3-10 meter range (9-30 feet) the GPS location often doesn't fall directly on a road. It would be desirable to take the GPS location and 'Snap To' the nearest road location.

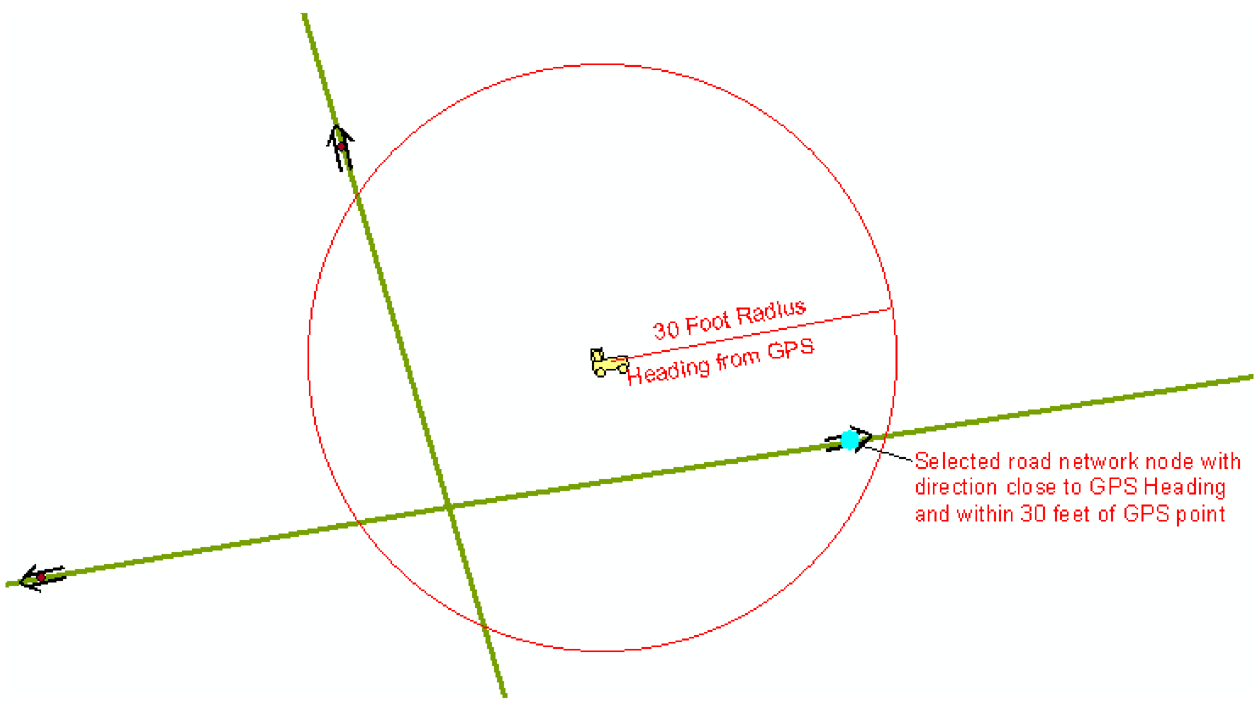
This document will walk you through the implementation of a 'Snap To' process within GeoEvent Server. This process relies on a pre-built road network dataset provided by a feature service and utilizes a Spatial [Query] Field Enricher to select the road segment to snap to. Please note that the selected segment may not be the nearest segment but will be within a well-defined tolerance (e.g. 30 feet).

Methodology

From an existing road network, we will create a set of network nodes. Each node will be attributed with the road attributes (name, ID, etc.) and the road segment direction of travel bearing. For bi-directional travel roads, a node for each travel direction will be created. This set of nodes will be published as a feature service.

For a given GPS location, we will use the Spatial Field Enricher to request the set of nodes within a given distance and a bearing within a specified range.

Example: For the diagram below we will select the first node that is within 30 feet of the given GPS point and has a bearing that is the GPS heading +- 10 degrees.



GeoEvent Server – Implementing a 'Snap To' Process

Creating the Road Network Nodes

The starting point is a set of data that includes the road centerlines. If you have each direction of roadway travel that is even better, but not required. Each road centerline should contain information about your road network such as road name, id, etc.

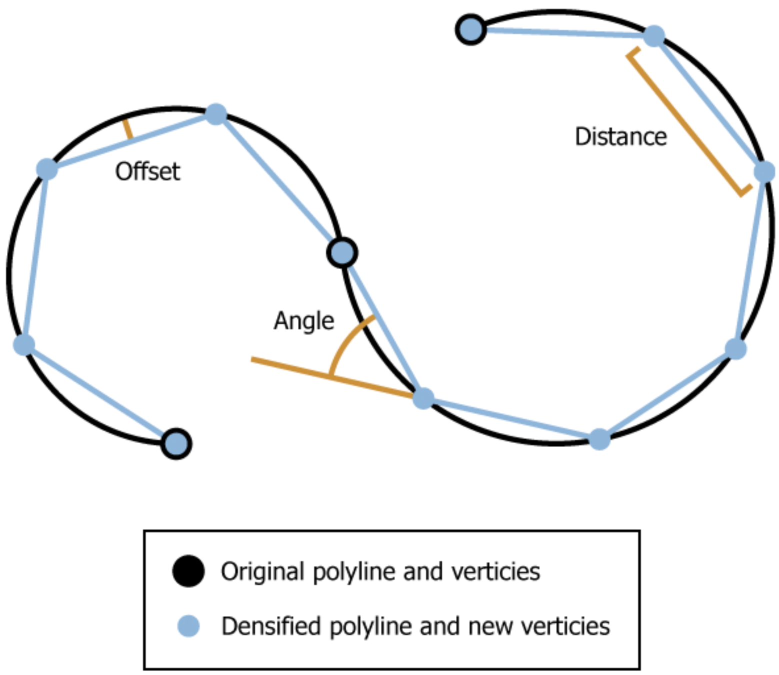
Densify the Centerlines

Note that this will modify the original feature class, so you may want to make a copy before running densify. You may have to experiment with this tool to obtain the best results for your road network (e.g. running first with offset then distance or making the distance value smaller/larger).

https://pro.arcgis.com/en/pro-app/tool-reference/editing/densify.htm

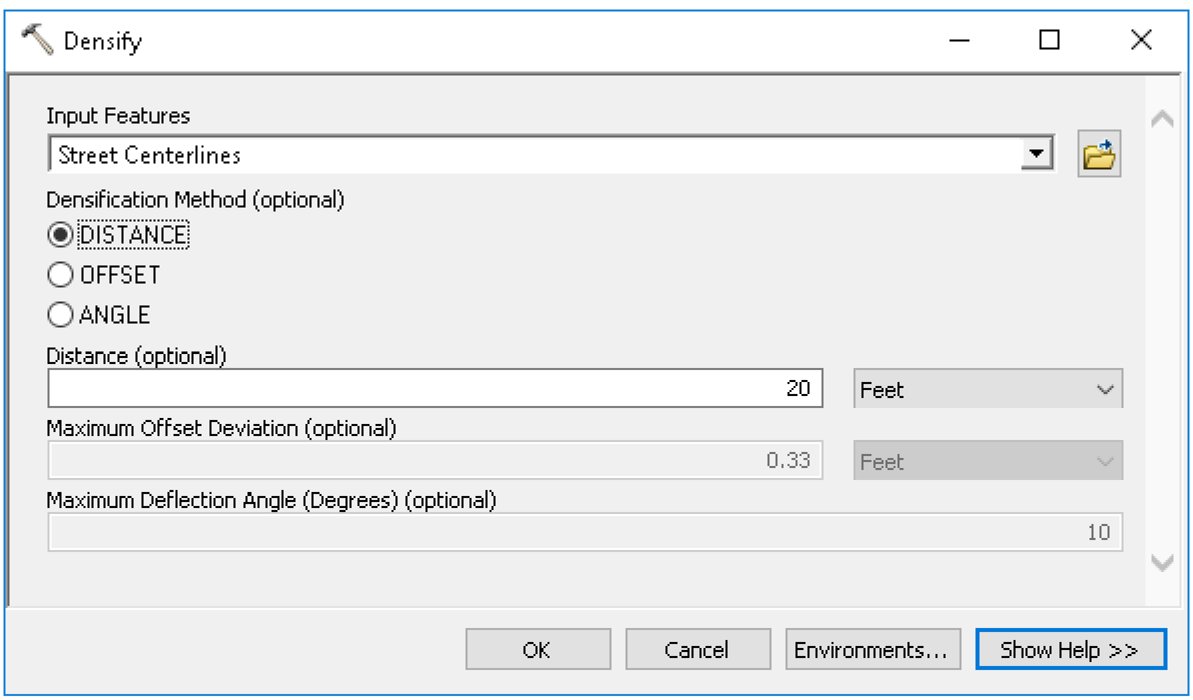
- **Input Features**: Select the copy of the street centerlines feature class
- **Densification Method**: DISTANCE
- **Distance**: Choose a value between 10 and 50 feet. This will affect the number of nodes in your dataset and may impact the resolution of the 'snap'.

GeoEvent Server – Implementing a ‘Snap To’ Process



The curve is densified into linear segments by either the Offset, Distance, or Angle.

⚠ **Caution:** This tool modifies the input data. See [Tools that do not create output datasets]() for more information and strategies to avoid undesired data changes.



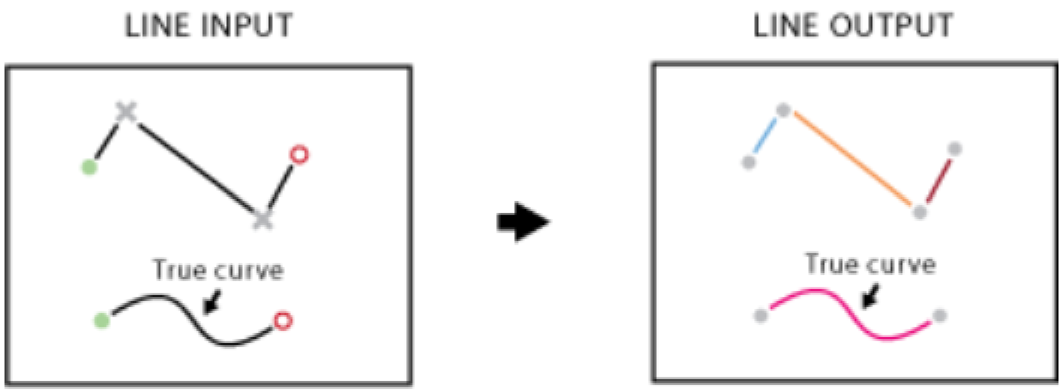
Split Line at Vertices

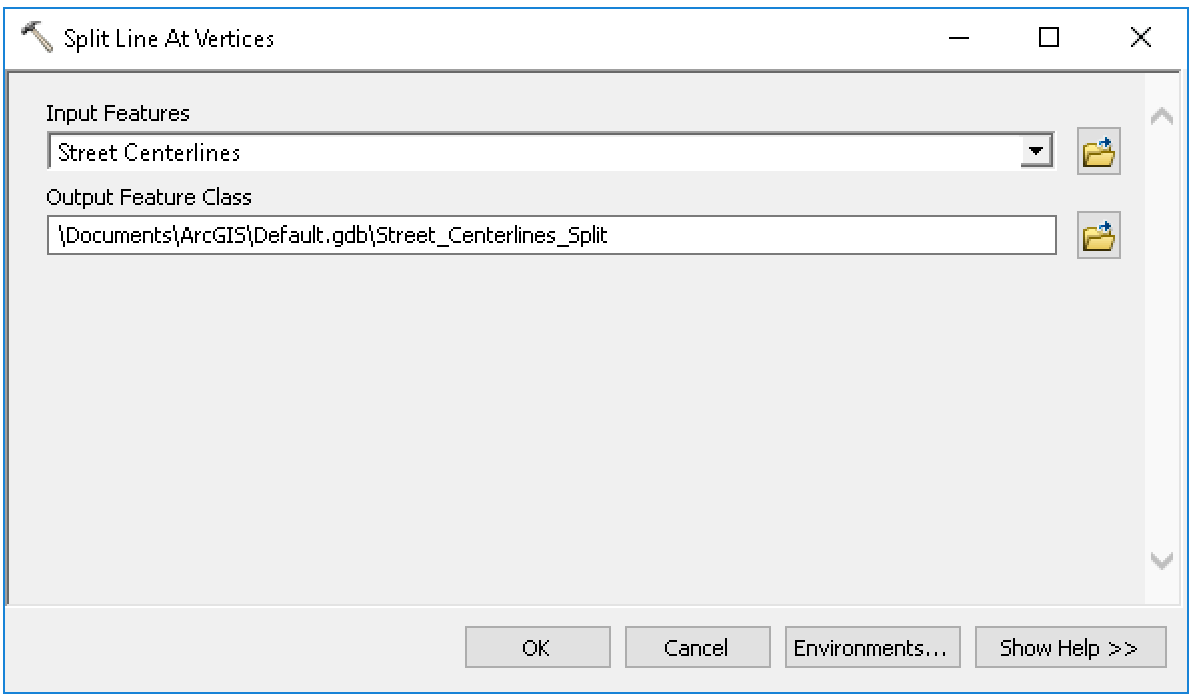
Next, we will split each line at each of the vertices. Since we densified the lines, each resulting segment should be less than or equal in length to the densified distance above (e.g. 20 feet).

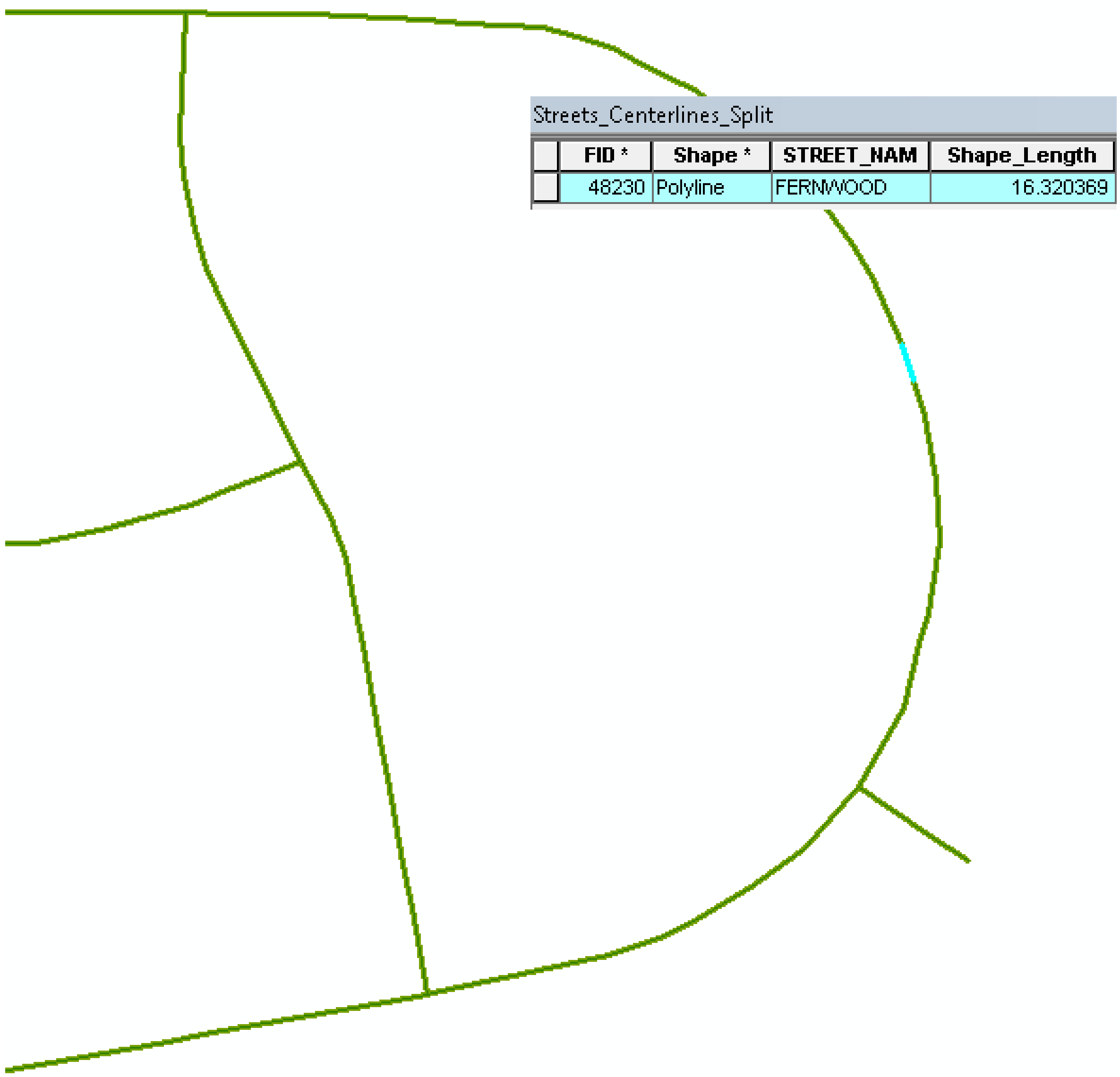
https://pro.arcgis.com/en/pro-app/tool-reference/data-management/split-line-at-vertices.htm

GeoEvent Server – Implementing a 'Snap To' Process

- **Input Features**: The densified street centerlines feature class from above.
- **Output Features**: A new feature class containing the split street centerlines.







Add Geometry Attributes

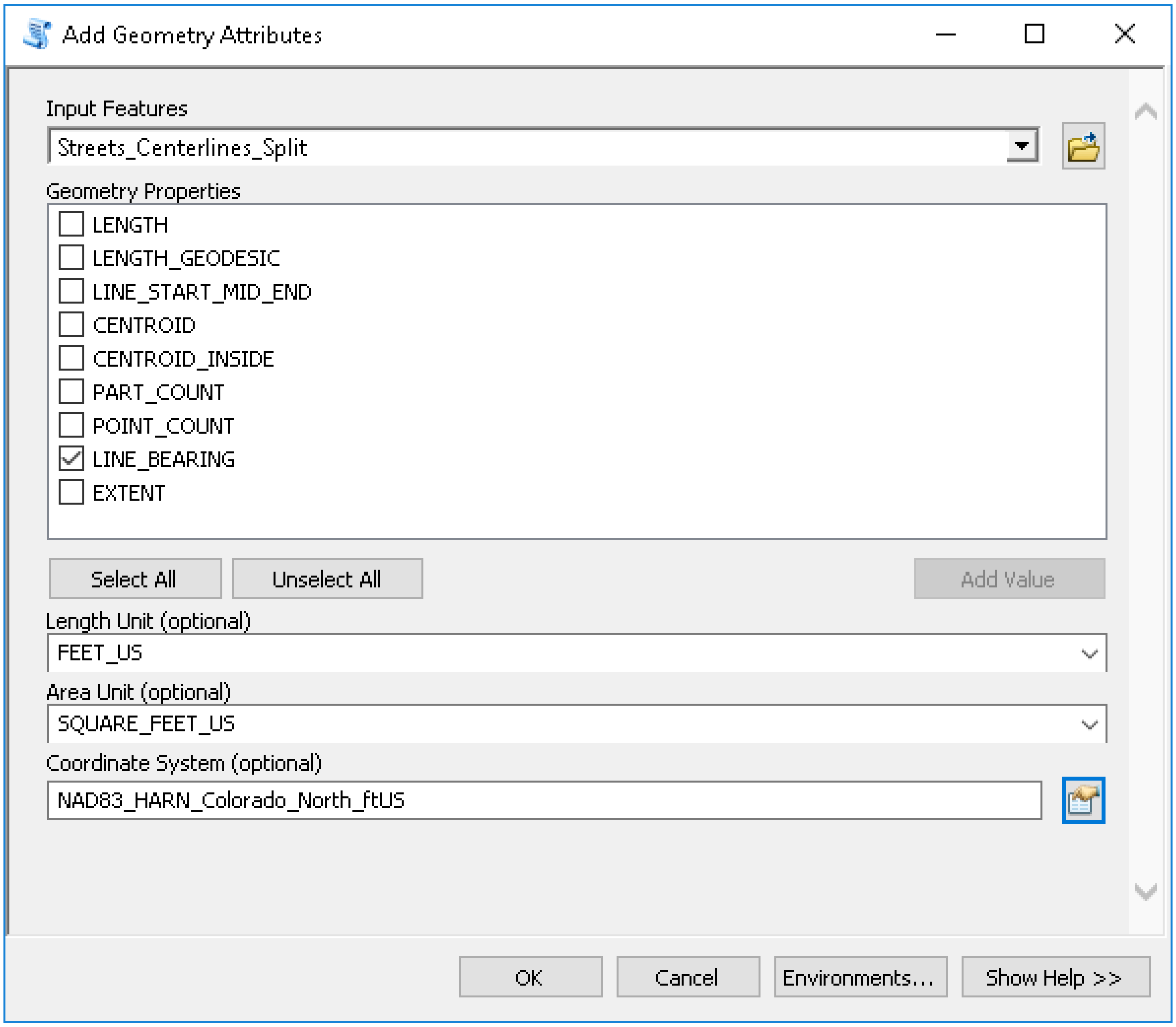
Next, we will use the Add Geometry Attributes tool to add the line bearing to each line.

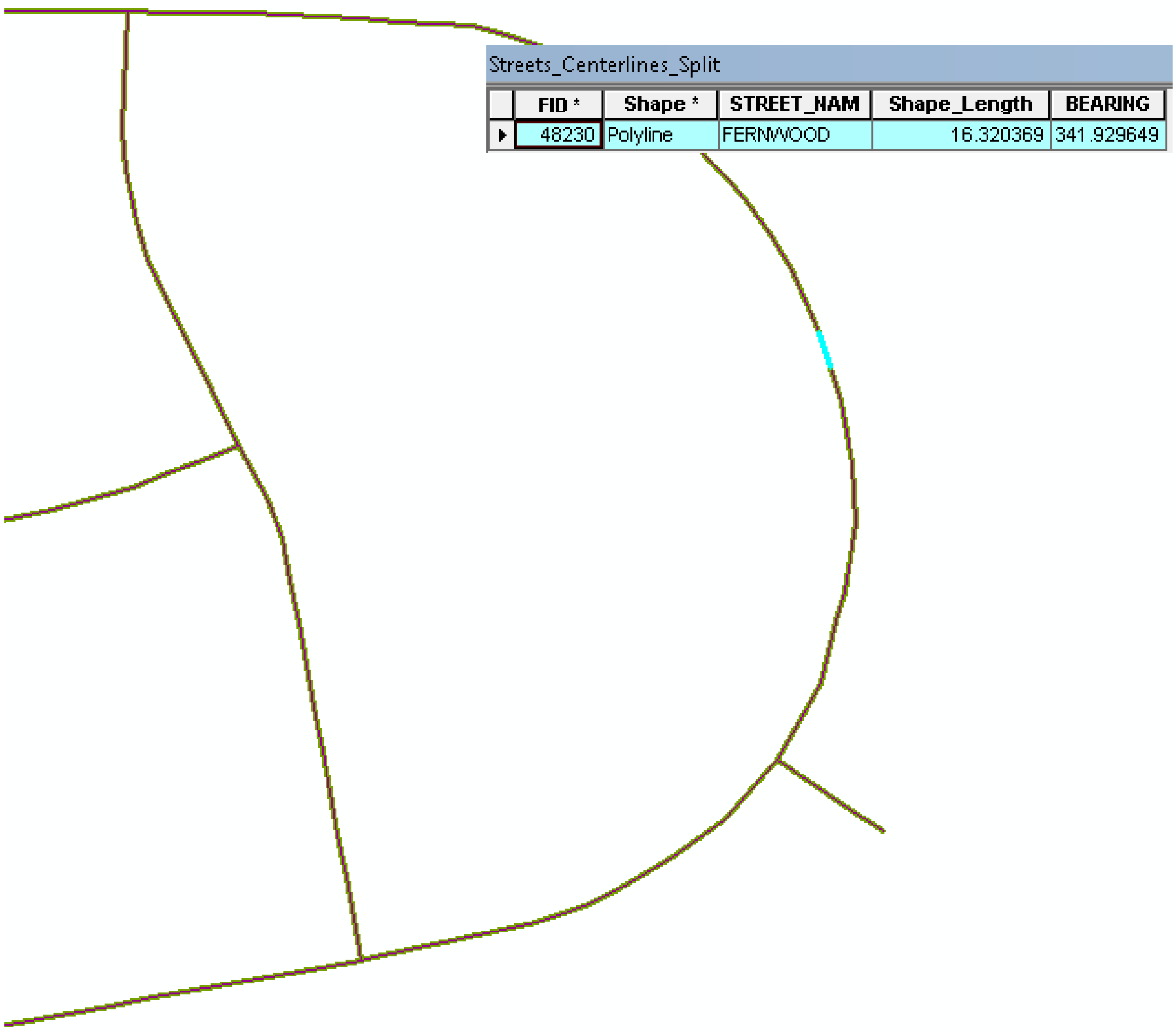
https://desktop.arcgis.com/en/arcmap/10.3/tools/data-management-toolbox/add-geometry-attributes.htm

- **Input Features**: Streets_Centerline_Split from above
- **Geometry Properties**: LINE_BEARING
- All other properties are optional.

LINE_BEARING	**BEARING**: The start-to-end bearing of the line. Values range from 0 to 360, with 0 meaning north, 90 east, 180 south, 270 west, and so on.

GeoEvent Server – Implementing a 'Snap To' Process





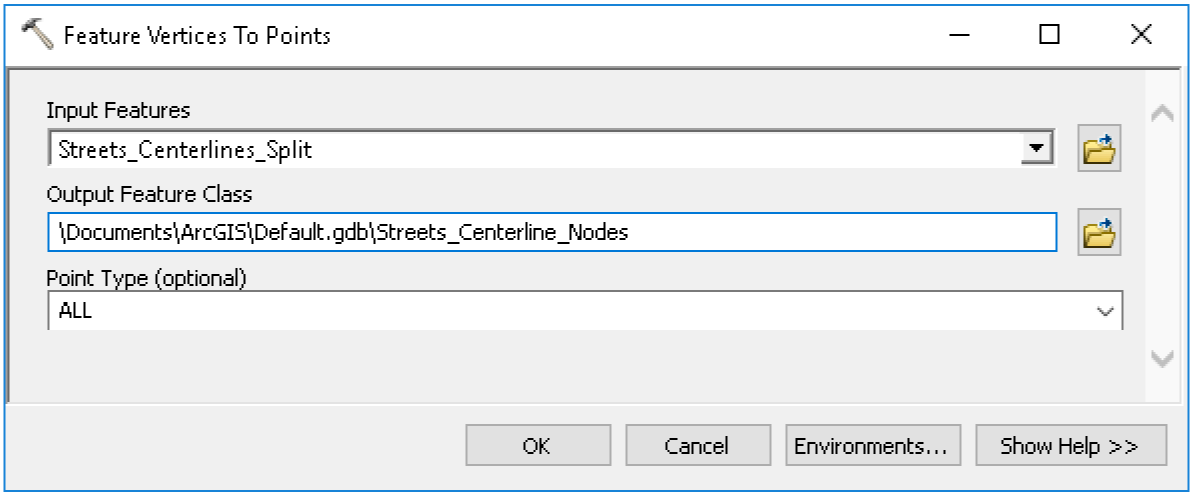
Create Nodes

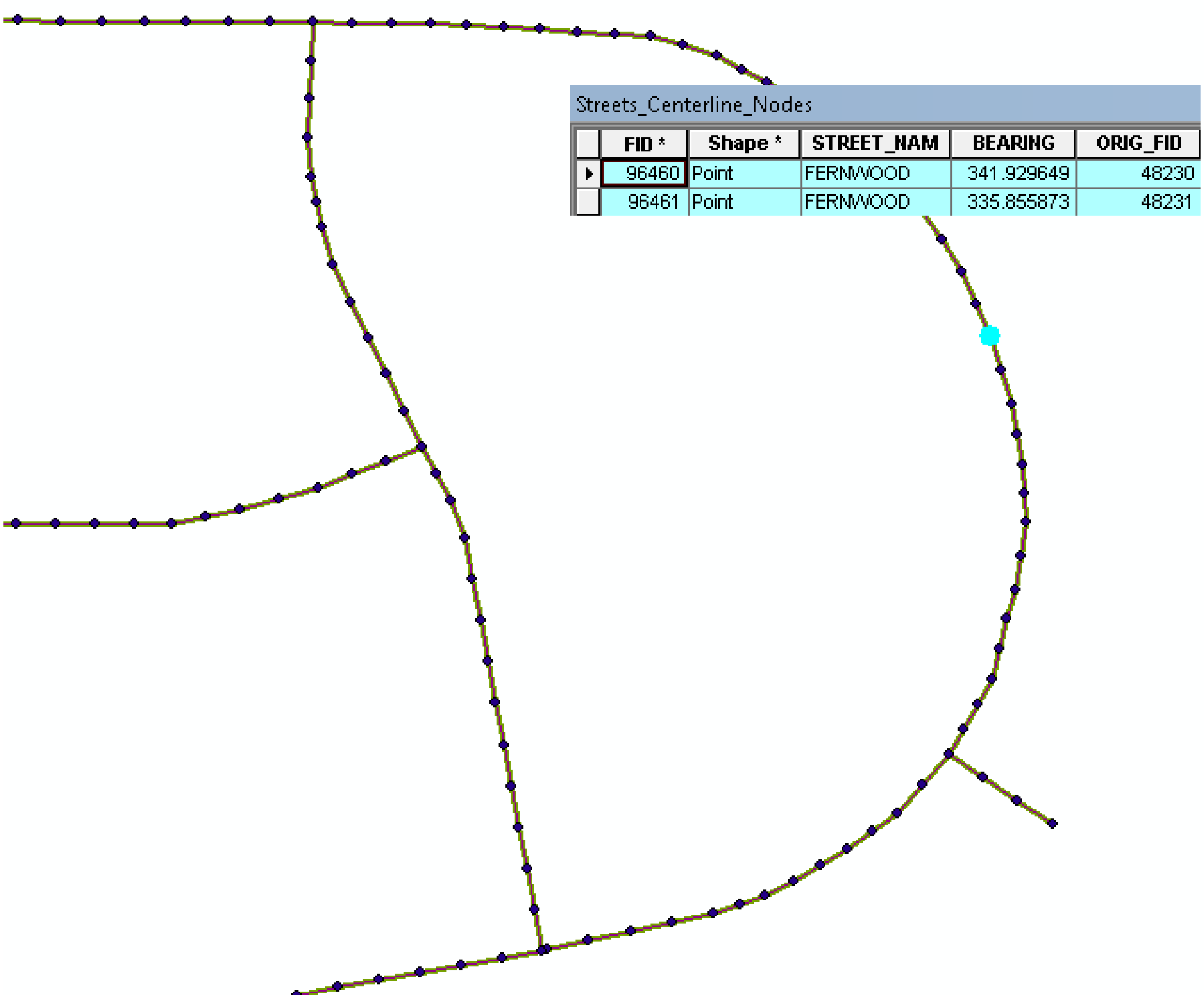
Finally, we convert the segment vertices to points. These points represent the nodes within our network. Note that running this tool may create overlapping node points. That is ok, since we will later be selecting only one matching node.

https://desktop.arcgis.com/en/arcmap/latest/extensions/production-mapping/converting-vertices-to-point-features.htm

- **Input Features**: The Street_Centerlines_Split features from above with the added bearing.
- **Output Feature Class**: Street_Centerline_Nodes
- **Point Type**: All

GeoEvent Server – Implementing a 'Snap To' Process





[Optional] Add Two-Way Street Segment Nodes

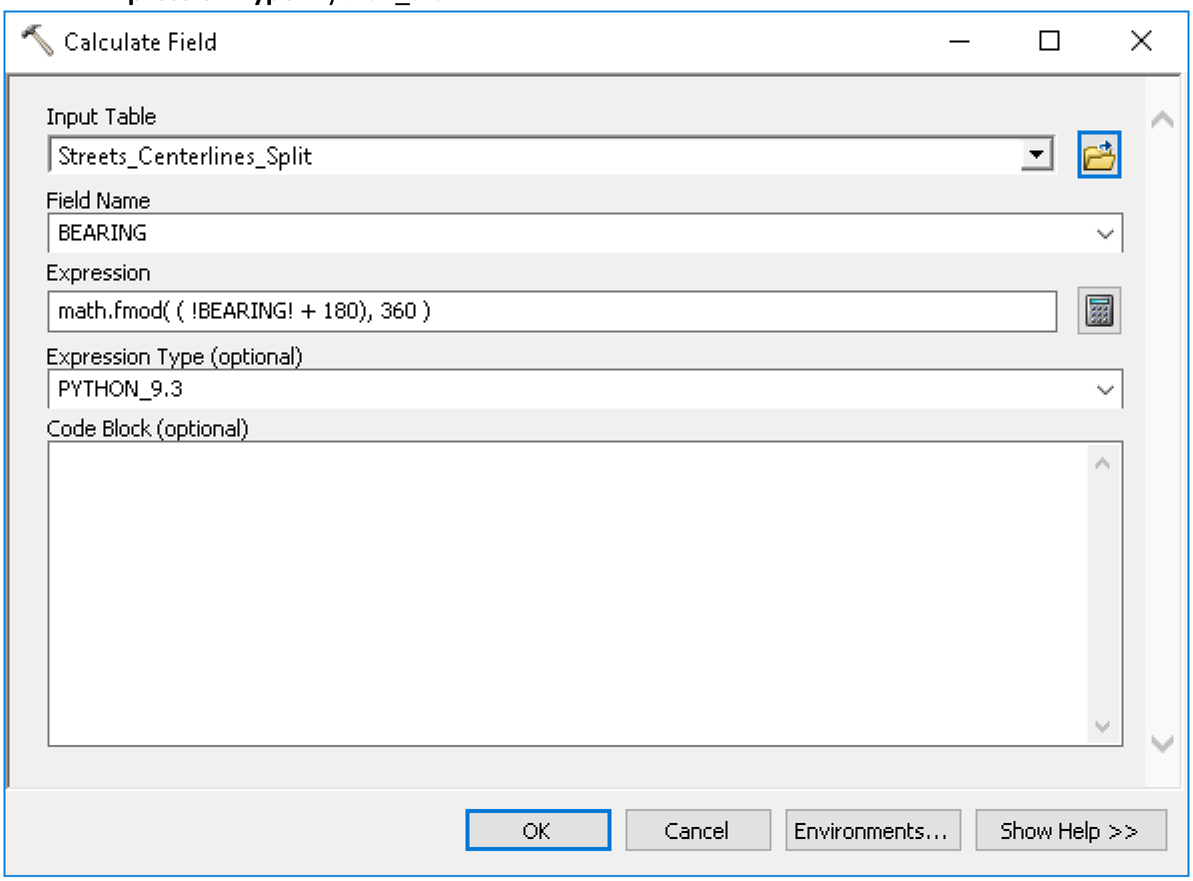
If your road centerlines don't capture two-way traffic directions (one line for two directions of travel), then you will want to complete this step to ensure you can snap GPS points to either direction.

1. Select all road centerlines from the Densified dataset above that are single lines but represent travel in two directions.
2. Copy the selected segments out to a separate feature class.
3. Complete the remaining steps for the copied line segments (split, add bearing, convert vertices to points).
4. Use the field calculation tool to reverse the bearing value

   Field Name: BEARING

   Expression: math.fmod((!BEARING! + 180), 360)

   Expression Type: Python_9.3



5. Append the resulting nodes to the original node feature class.

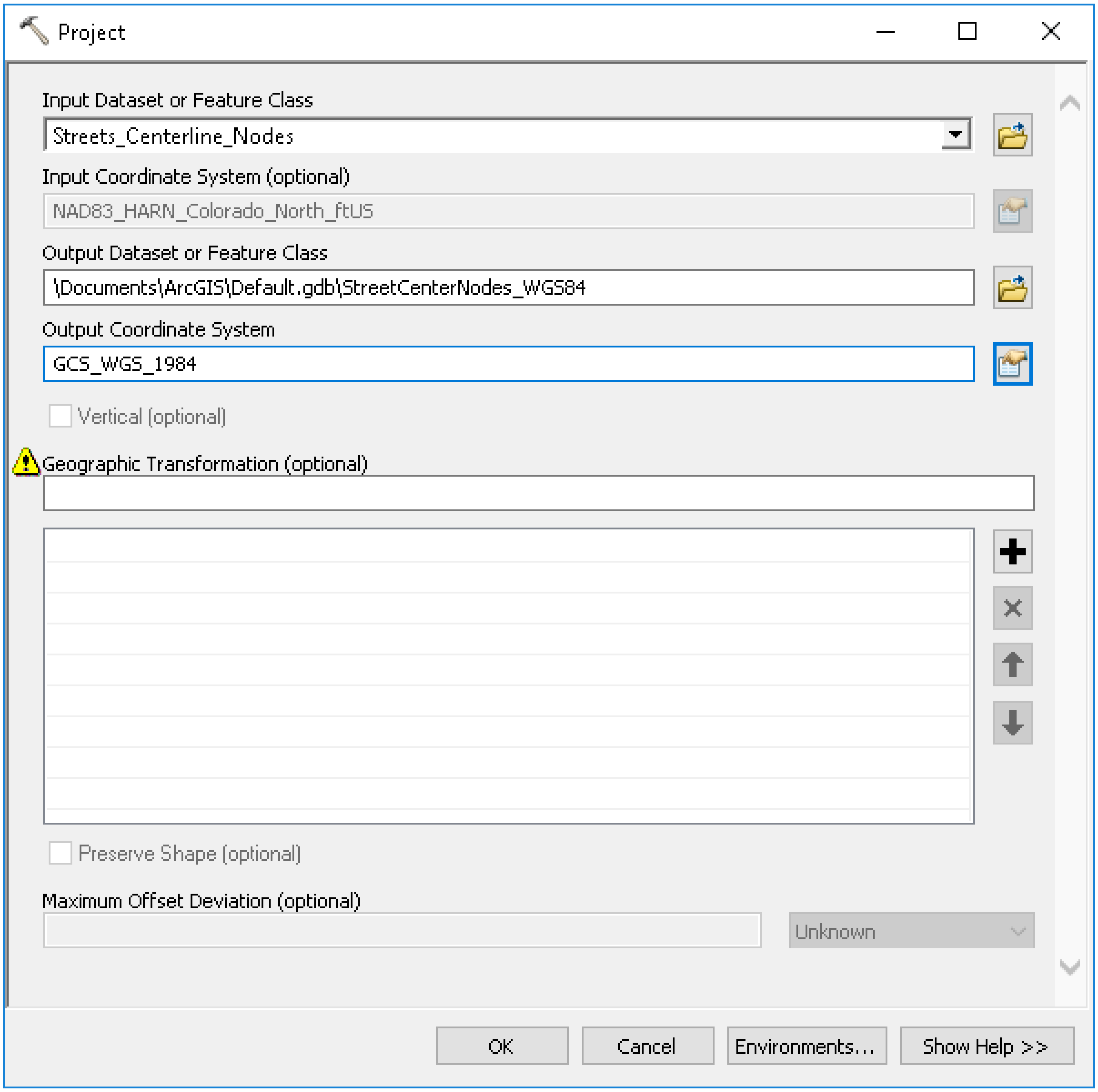
Project to GPS Coordinate System (WGS84)

Finally, project your data to the coordinate system that your GPS devices will be reporting in. This is most likely WGS 84.

https://pro.arcgis.com/en/pro-app/tool-reference/data-management/project.htm

- **Input Dataset**: Street_Centerline_Nodes
- **Output Dataset**: StreetCenterNodes_WGS84

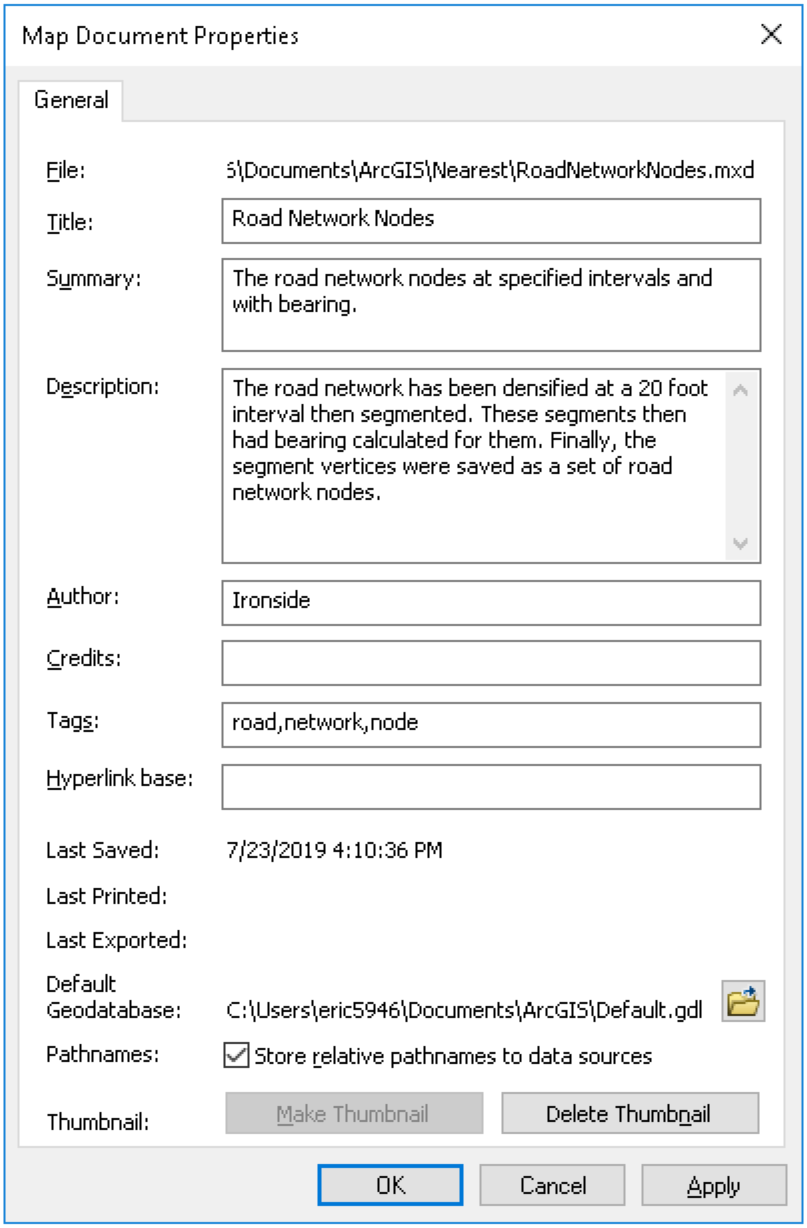
- **Output Coordinate System**: Geographic/World/GCS_WGS_1984
- **Geographic Transformation**: Supply the correct transformation for your data



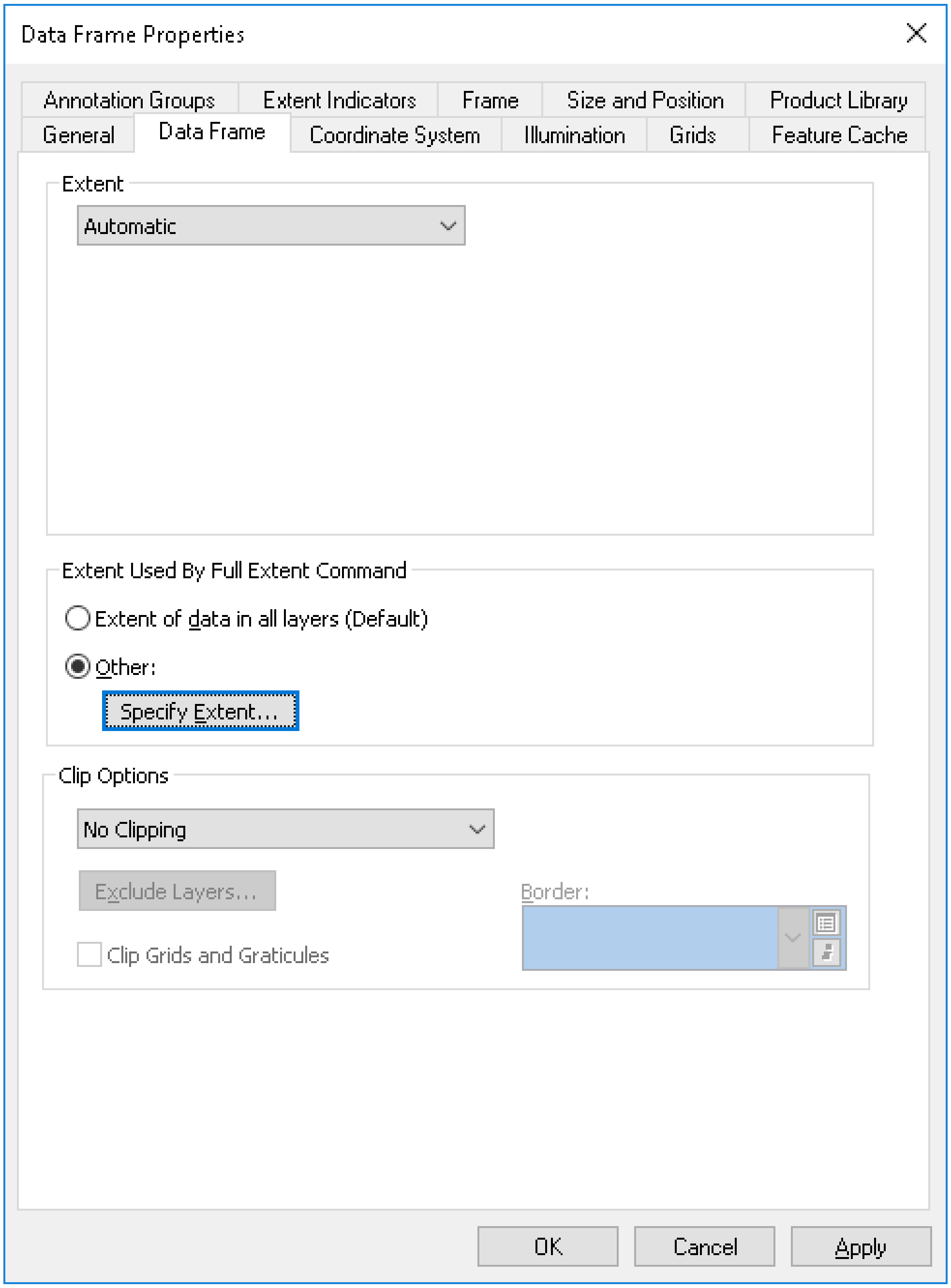
Publish a Feature Service

Now you will publish a feature service to allow GeoEvent to access the road network nodes.

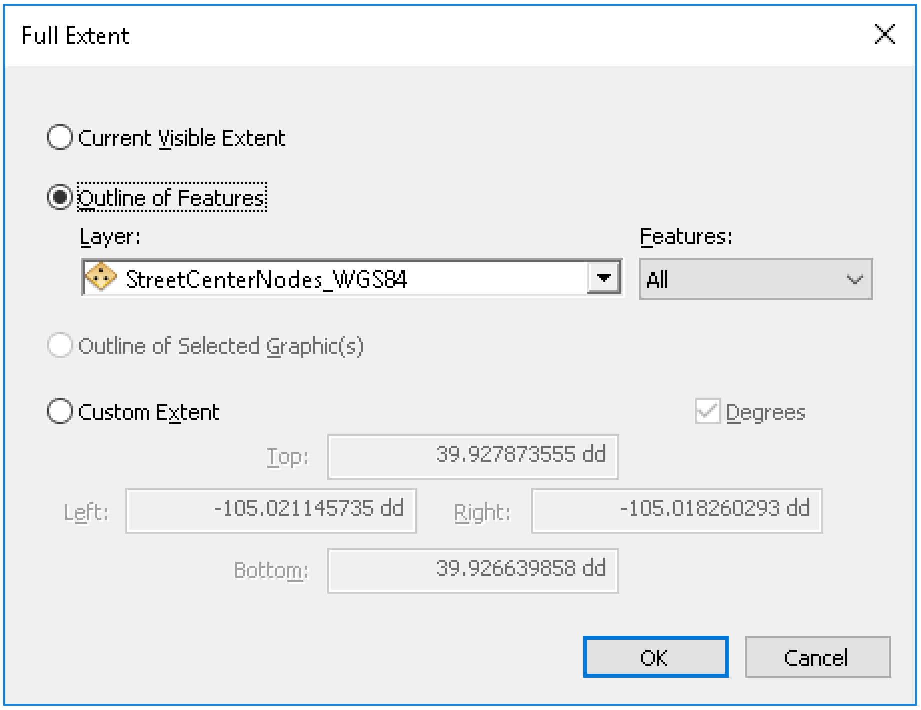
1. Open ArcMap and create a new map document.
2. Add the feature class StreetCenterNodes_WGS84.
3. From the File menu, edit the Map Document Properties and press OK.



4. Save the document as RoadNetworkNodes.mxd in the location of your choosing.
5. Double click on your Layers data frame
    a. Go to the Data Frame tab and select Other for **Extent Used by Full Extent Command**



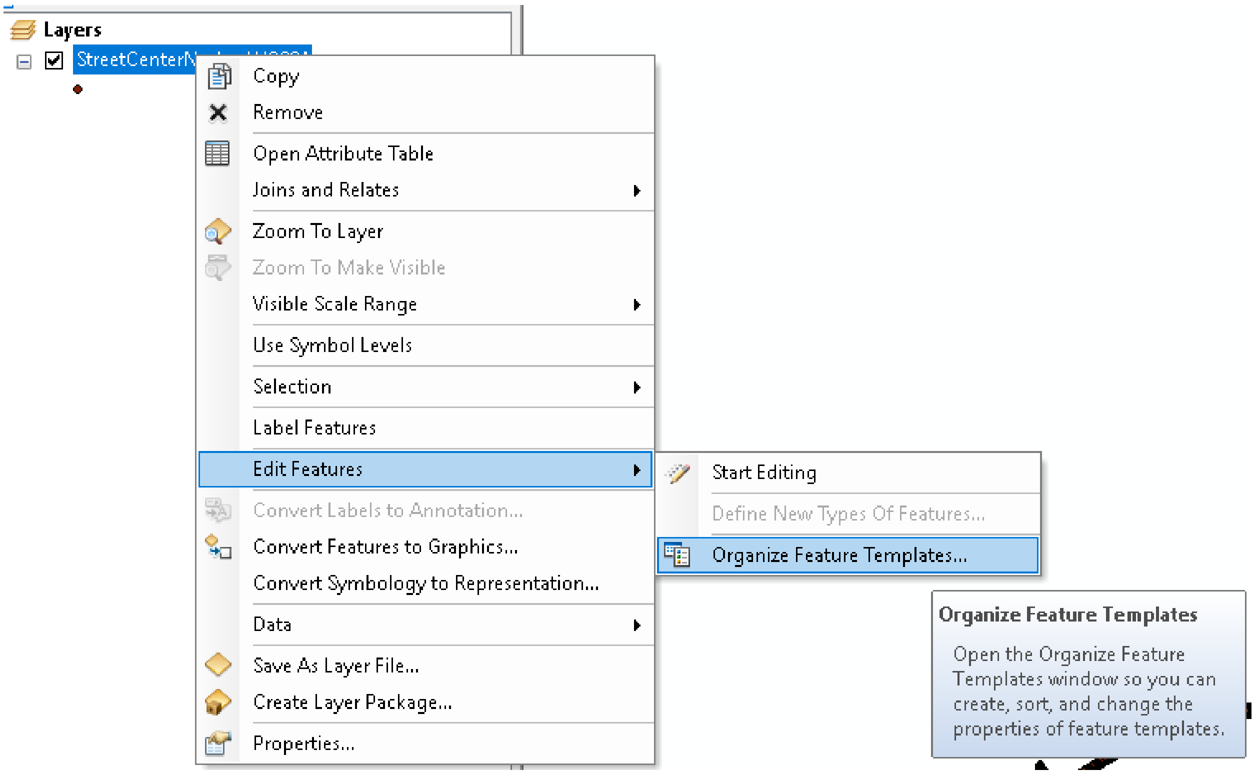
b. Press the ***Specify Extent*** button and select **Outline of Features** (be sure the node feature class is selected)



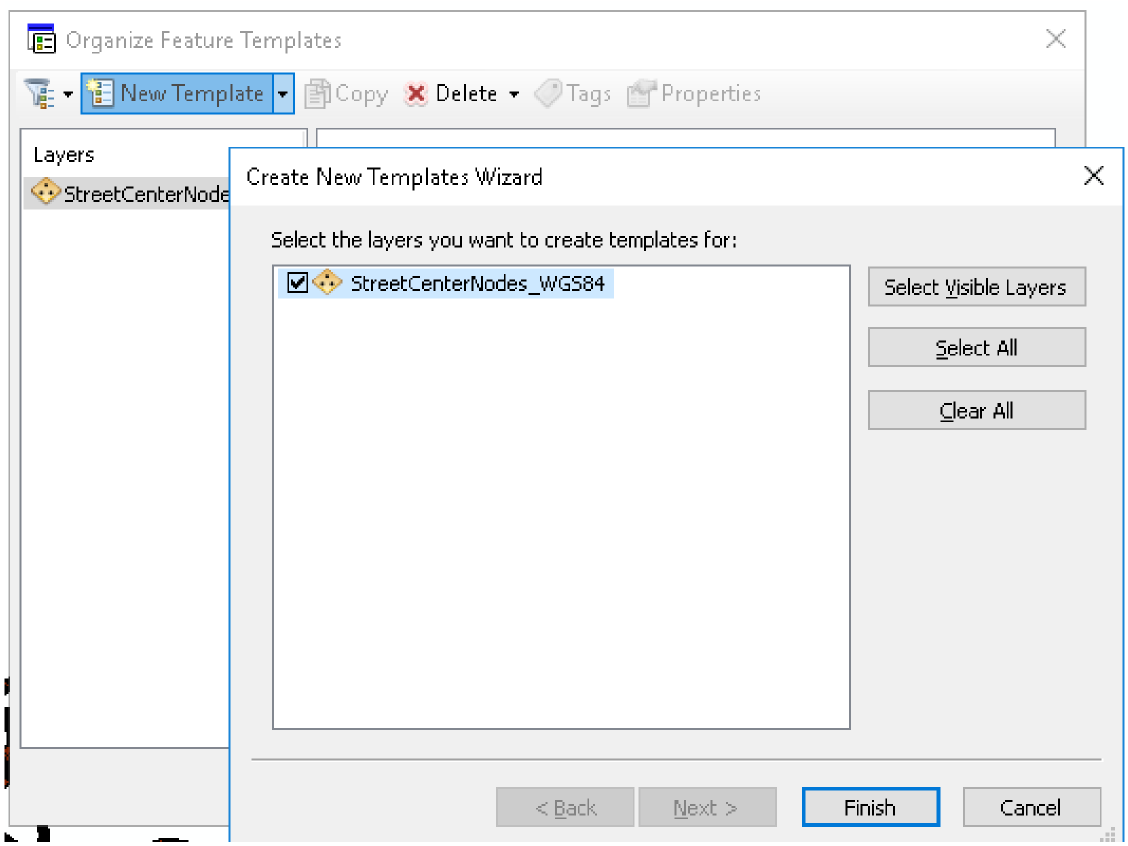
   c. Save your changes to the data frame.

6. Edit the properties of the **StreetCenterNodes_WGS84** feature layer (double click)
   a. On the **General Tab**, set the **Scale Range** to **Don't show layer when zoomed:**
      i. **Out beyond:** 1:591657528
      ii. **In beyond:** <None>
   b. Save your changes.
7. Right click on your **StreetCenterNodes_WGS84** feature layer and select **Edit Features**, **Organize Feature Templates**.

GeoEvent Server – Implementing a ‘Snap To’ Process



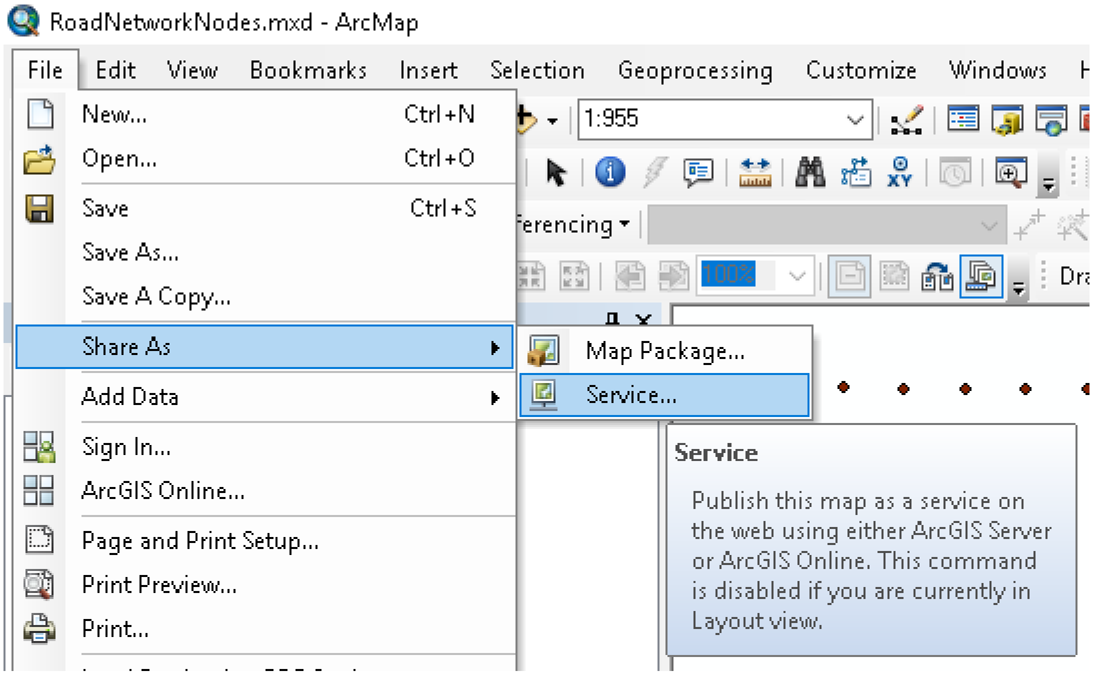
a. Select New Template, select all layers in the map, and press Finish.



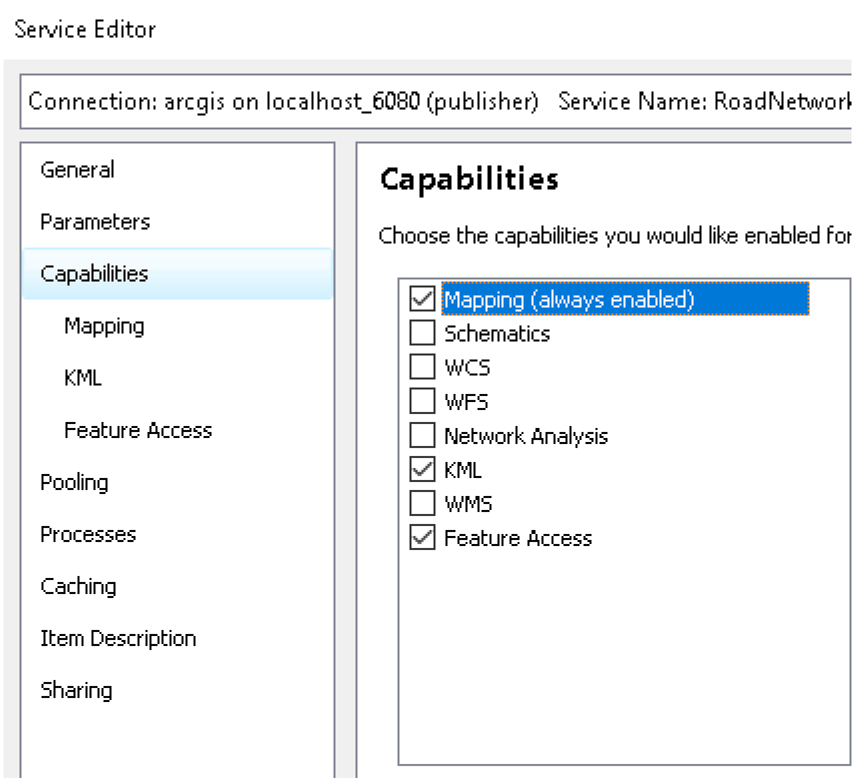
b. Press close to save your changes

8. Save your map document.

9. From the file menu, share the document as a Service…



   a. Publish a service
   b. **Service Name**: RaodNetworkNodes
   c. Use an existing or create a new folder.
   d. On the Service Editor, Capabilities tab check the Feature Access check box.



   e. Press the **Publish** button at the top right.

i. Your node data can be stored in a registered geodatabase or you can copy the data to the ArcGIS Server.

Working in GeoEvent

Adding the Spatial Field Enricher

A custom processor is used to query the road network nodes spatially. This new custom processor is called the Spatial Field Enricher. It utilizes the feature service's REST interface to query the data. However, instead of joining data using an attribute join, it uses the event's geometry to query the data.

1. Unzip the file that contains the custom field enricher into a temporary directory.
2. In **GeoEvent Manager**, go to **Site > Components > Processors** and **select Add Local Processor**
3. Press **Choose Files**, browse to the location of the **spatialfieldenricher-processor-10.6.1.jar** file and press **Open**.
4. Press the **Add** button to add the custom processor to your GeoEvent.

GeoEvent Manager

Switching to GeoEvent Manager, it is assumed that an input already exists for the GPS data and the output will be to the GeoEvent Logger (create the output if it does not already exist).

Create Merged GeoEvent Definition

The Field Enricher has the option to add new fields or modify existing fields. Best practice is to modify existing fields. Because of this, we need to create a GeoEvent Definition that represents the merge of our GPS data with our Road Network Node data.

It is assumed your GPS data already has a GeoEvent Definition within GeoEvent Manager. For the demonstration below, our GPS data has a relatively simple schema (yours may be more complex).

GeoEvent Definition Name: GPS_IN

Field Name	Type	Tag
UserID	String	TRACK_ID
FixTime	Date	TIME_START
Speed	Double	
Heading	Integer	
Geometry	Geometry	GEOMETRY

Add a new GeoEvent Definition that merges the fields from your GPS events and the data from the Road Network Nodes. You will want to make sure the field names for the Road Network Nodes match exactly (including case) the field names from the feature service. Rename any GPS field names as needed (e.g. the geometry field). An example follows:

GeoEvent Definition Name: GPS_RoadSnap

Source	Field Name	Type	Tag
GPS	GPSUserID	String	TRACK_ID

GPS	GPSFixTime	Date	TIME_START
GPS	GPSSpeed	Double	
GPS	GPSHeading	Integer	
GPS	GPSGeometry	Geometry	
Road Network Nodes	RoadID	String	
Road Network Nodes	Name	String	
Road Network Nodes	Bearing	Double	
Road Network Nodes	Geometry	Geometry	GEOMETRY

Create the GeoEvent Service

Now you will create the GeoEvent service that will process the data.

1. Create a new GeoEvent Service.
2. Add the GPS data input and the Logger output.
3. Add a Field Mapper Processor
    a. **Name**: Map to Merged Schema
    b. **Processor**: Field Mapper
    c. **Source GeoEvent Definition**: GPS_IN
    d. **Target GeoEvent Definition**: GPS_RoadName
    e. **Field Mappings**:

Source Fields	Target Fields
UserID	GPSUserID
FixTime	GPSFixTime
Speed	GPSSpeed
Heading	GPSHeading
Geometry	GPSGeometry
	RoadID
	Name
	Bearing
	Geometry

4. Add a Field Enricher (Spatial Query) Processor
    a. Name: Snap To
    b. Processor: Field Enricher (Spatial Query)
    c. ArcGIS Server Connection: Default
    d. Folder: Root
    e. Service: RoadNetworkNodes
    f. Layer: RoadNetworkNodes_WGS84
    g. Target Fields: Existing Fields
    h. Enrichment Fields: RoadID, Name, Bearing, Geometry
    i. GeoEvent Join Field: GPSGeometry
    j. Where Clause: Bearing > (${ GPSHeading } -10) and Bearing < (${GPSHeading} + 10)
    k. Buffer Distance: 20.0
    l. Buffer Units: Feet
    m. Spatial Relationship: Intersects

    n. Maximum Results Returned: 1
    o. Cache Refresh Time Interval: 1440
    p. Maximum Number of Feature Records: 1000
    q. Feature Layer Join Field: OBJECTID
5. Connect the input -> Field Mapper -> Field Enricher -> output
6. Press the Publish button.

At this point you should see data flowing into your GeoEvent Logger.

Using the Spatial Field Enricher

Each of the properties of the Spatial Field Enricher are discussed below.

- **ArcGIS Server Connection**: This is the GeoEvent Data Store

- **Folder**: Each data store may have one or more folders (root is the base folder)

- **Service**: This is the name of the feature service that contains the feature layers to query.

- **Layer**: This is the layer that contains the features to query.

- **Target Fields**: Existing Fields or New Fields. Existing fields expects the incoming GeoEvent Definition to contain the same fields (name and type) as the enrichment fields below. Field names are case sensitive. New Fields will create a copy the incoming GeoEvent Definition and add the new enriched fields to it. You must take care not to have duplicate field names.

- **Enrichment Fields**: These are the fields whose values you want to add to the incoming GeoEvent.

- **GeoEvent Join Field**: This is the geometry field to use to do the spatial query. You can select either the field name or the tag GEOMETRY. Since some GeoEvent Definitions contain more than one Geometry, this will help you choose which one to select.

- **Where Clause**: This where clause is applied to the data while it is being queried. You can use field substitution by enclosing the field name (case sensitive) in ${} notation. For example, the where clause we set above "Bearing > (${ GPSHeading } -10) and Bearing < (${GPSHeading} + 10)" is asking for data that has a Bearing value within +-10 degrees of the current GeoEvent's GPSHeading value. For example: If a GPS event has a heading of 90 degrees (due east), this where clause would evaluate to "Bearing > (90 -10) and Bearing < (90+10)" selecting any data whose Bearing value is between 80 and 100.

- **Buffer Distance**: This is the distance the GeoEvent's Geometry will be buffered. For a GPS point, the query will create a buffer polygon that is centered on the point and has a radius equal to this value. For our example, we are searching for road network nodes that are within a 20 foot radius (40 foot diameter). To improve the chances of snapping to a node, increase this value.

You might also utilize a GPS quality measure (like horizontal dilution of position) to generate a buffer zone. For example, if one measurement has a error equal to 60 feet, and the next has a error of 10 feet, you might utilize that as your buffer distance by supplying the error value as the Buffer Distance. Utilize the ${} notation to use a field value instead of a constant. For example: ${GPSError}

- **Buffer Units**: These are the units the query is going to use to buffer the incoming geometry.

- **Spatial Relationship**: There are a variety of spatial relationships supported by the Spatial Field Enricher. For this application, we are looking for road network nodes that intersect (or are within) our buffer geometry.

- **Maximum Results Returned**: This is a number that must be between 1 and the maximum number of features a feature service can return (typically 1000). If you choose a number greater than 1 and the query returns more than 1 feature, a new GeoEvent will be generated for each returned feature. For example, if you set Maximum Results Returned to 2 and there were 2 road network nodes within your buffer geometry with acceptable bearing values, 2 new GeoEvents would be forwarded. Each one would contain the original GPS data along with the enriched fields from each of the queried features from the service. In this application we want to snap to a single point, so we chose 1 for this value. It is up to the feature service to decide which feature is returned. So if there are more than 1 road network nodes within the buffer geometry that meet the bearing criteria, only one will be returned (at random).

- **Cache Refresh Time Interval**: This processor will cache the results from each query and will not call out to the feature service if the cache contains data. The cache key is a string representation of the GeoEvent geometry. Thus, if two events have an identical geometry, the second event will be enriched from the cache. This is done for performance reasons.

  In this application, our road network nodes are static. It is assumed they will not change frequently and the cache doesn't need to be cleared out very often (1440 minutes is 24 hours).

- **Maximum Number of Feature Records**: The cache is stored in memory and can only hold so much. This property defines how many cached objects can be maintained in memory.

- **Feature Layer Join Field**: This property is not used and can be ignored.

Testing on a Map

Create a stream service from your GPS_IN GeoEvent Definition. Copy the GPS_RoadSnap GeoEvent Definition, call it GPS_RaodSnap_OUT, and remove the GPSGeometry field (only 1 geometry field). Create a second stream service from the GPS_RoadSnap_OUT GeoEvent Definition.

Add these Stream Service outputs to your GeoEvent Service. Connect the Input to the GPS_IN stream service. Connect the Field Enricher to a field mapper and map from the GPS_RoadSnap definitionto the

GPS_RoadSnap_OUT definition. Connect the second field mapper to the the GPS_RoadSnap_OUT Stream service.

In a web map, add the two stream services and verify they are both receiving data. Review the snapped points to be sure they are working as expected. Note that some points may not snap because of the tolerance set above (20 feet). You can increase or decrease this tolerance as needed.