

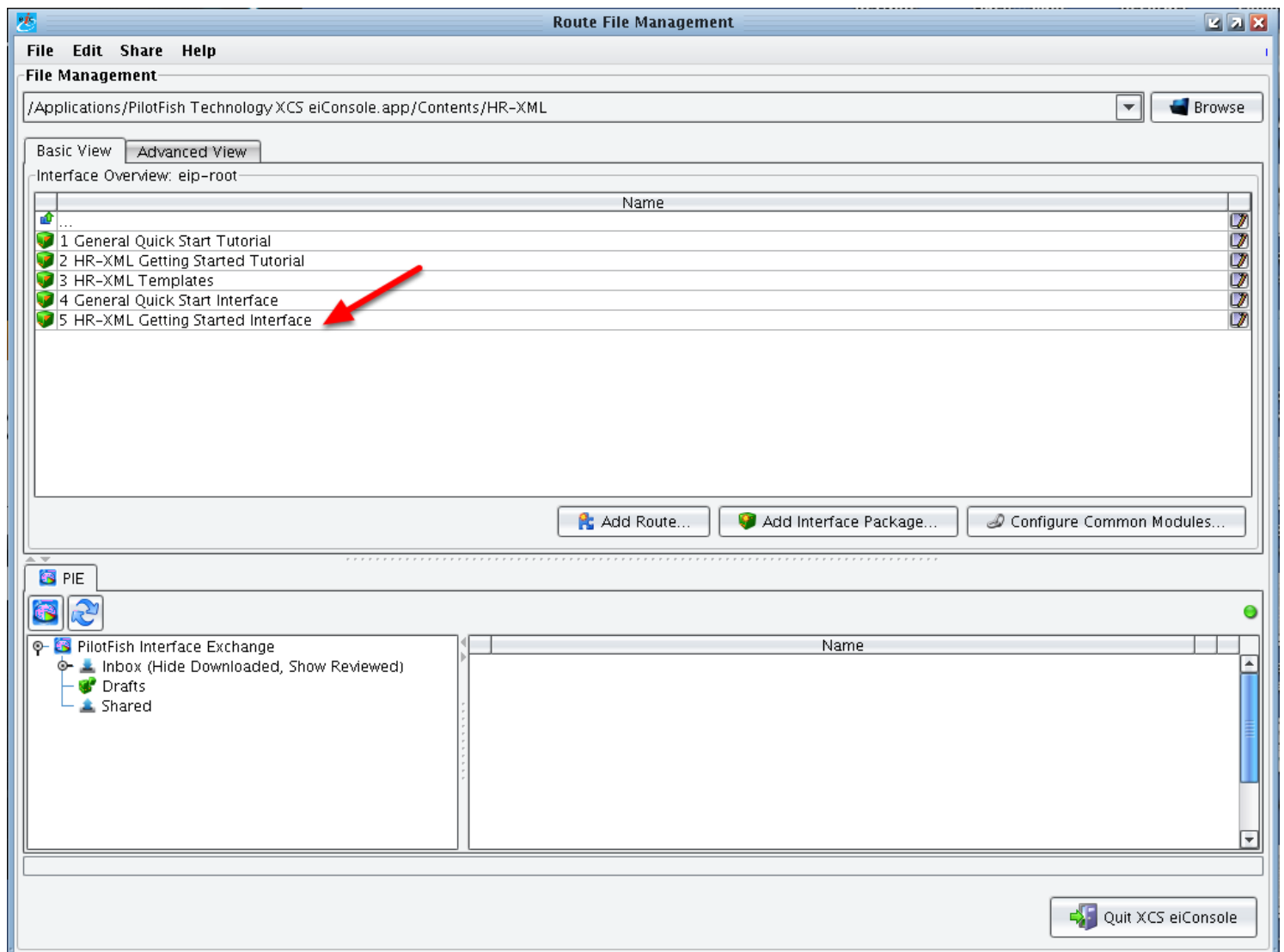
## 2 HR-XML Getting Started Tutorial

---

Welcome to the eiConsole for HR-XML Getting Started Tutorial. New users should complete the [General Quick Start Tutorial](#) to learn the basic principles and terminology used in the eiConsole before proceeding to this more advanced lesson. The HR-XML Getting Started Tutorial builds on concepts learned in the previous tutorial that are not repeated here and are required to complete this tutorial.

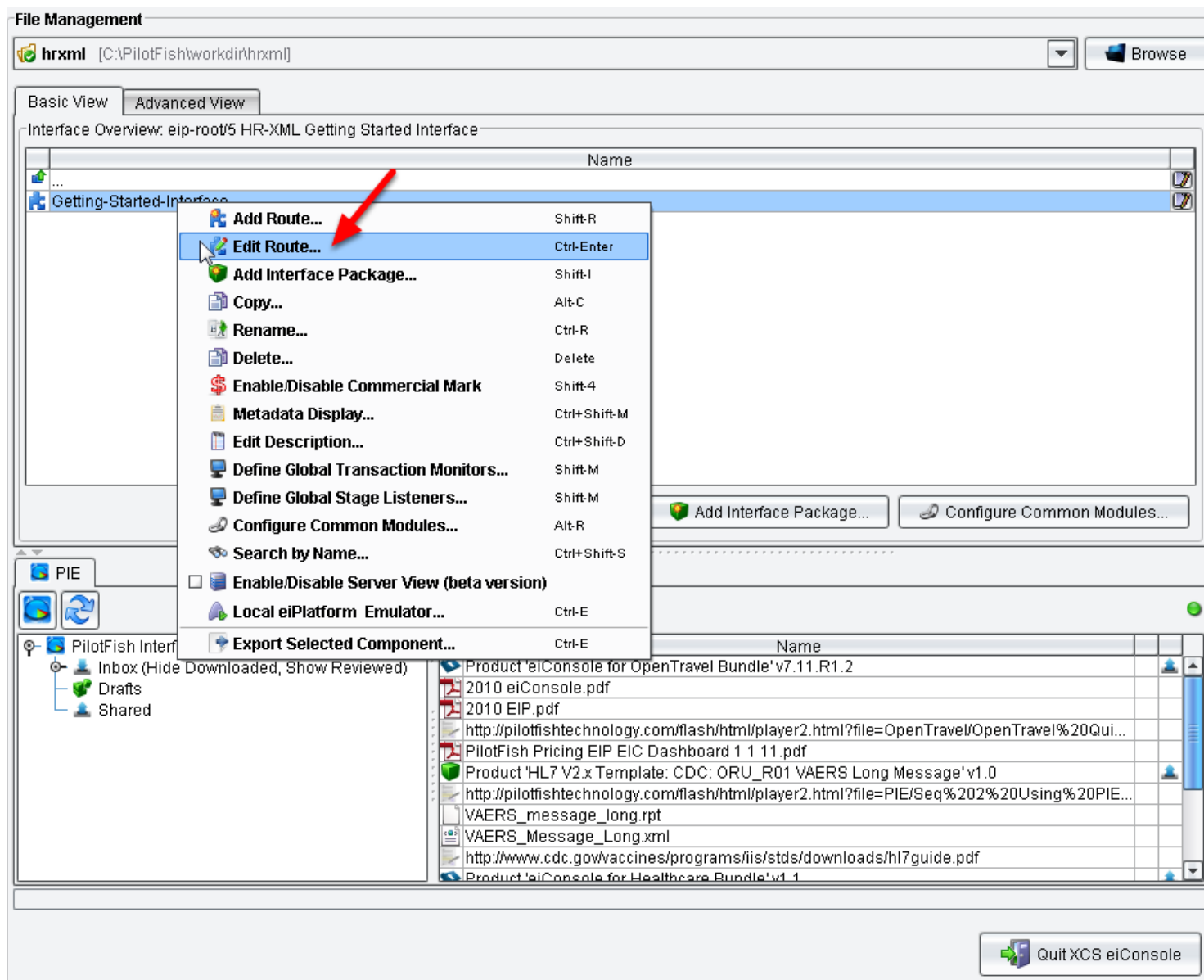
In this tutorial you will build a simple Process Screening Order interface. This interface will accept a list of candidates in a csv file and create a corresponding request for a background and/or credit check in the HR-XML 3.0 format. This interface will take the average user about 1 hour to complete.

Be sure to click on the row, 3 HR-XML Templates, to view the extensive selection of BOD templates which are included with this bundle. These can be used as the starting point in the eiConsole's paint-by-numbers process for developing interfaces.

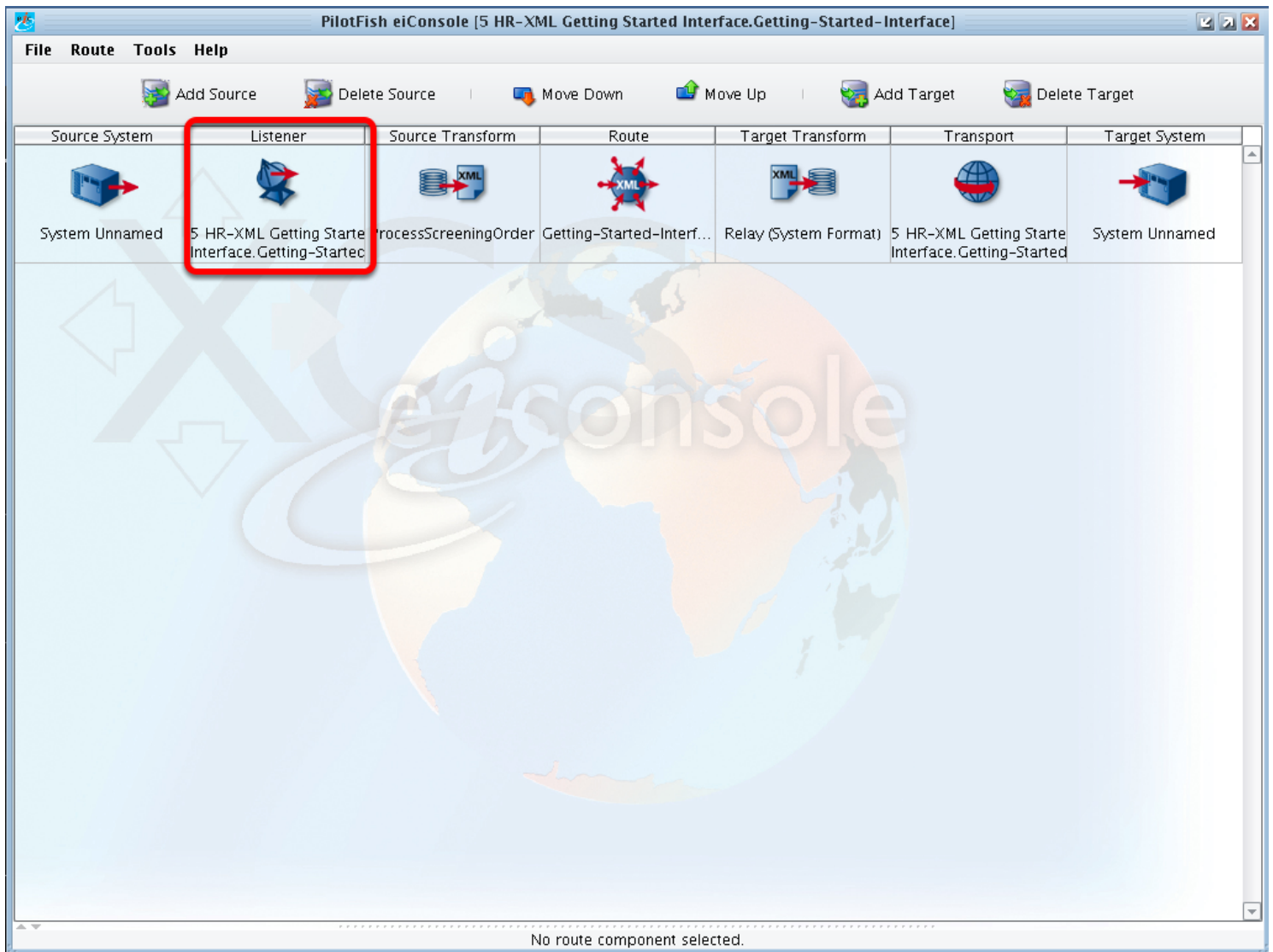


When first opening the eiConsole for HR-XML bundle, you will see 4 Interface Packages in the Route File Management window. These are: 1 General Quick Start Tutorial, 2 HR-XML Getting Started Tutorial, 3 HR-XML Templates and 5 HR-XML Getting Started Interface. If you completed the Quick Start Tutorial you should also have the 4 General Quick Start Interface Package.

Let's get started! Double click the **5 HR-XML Getting Started Interface** row.



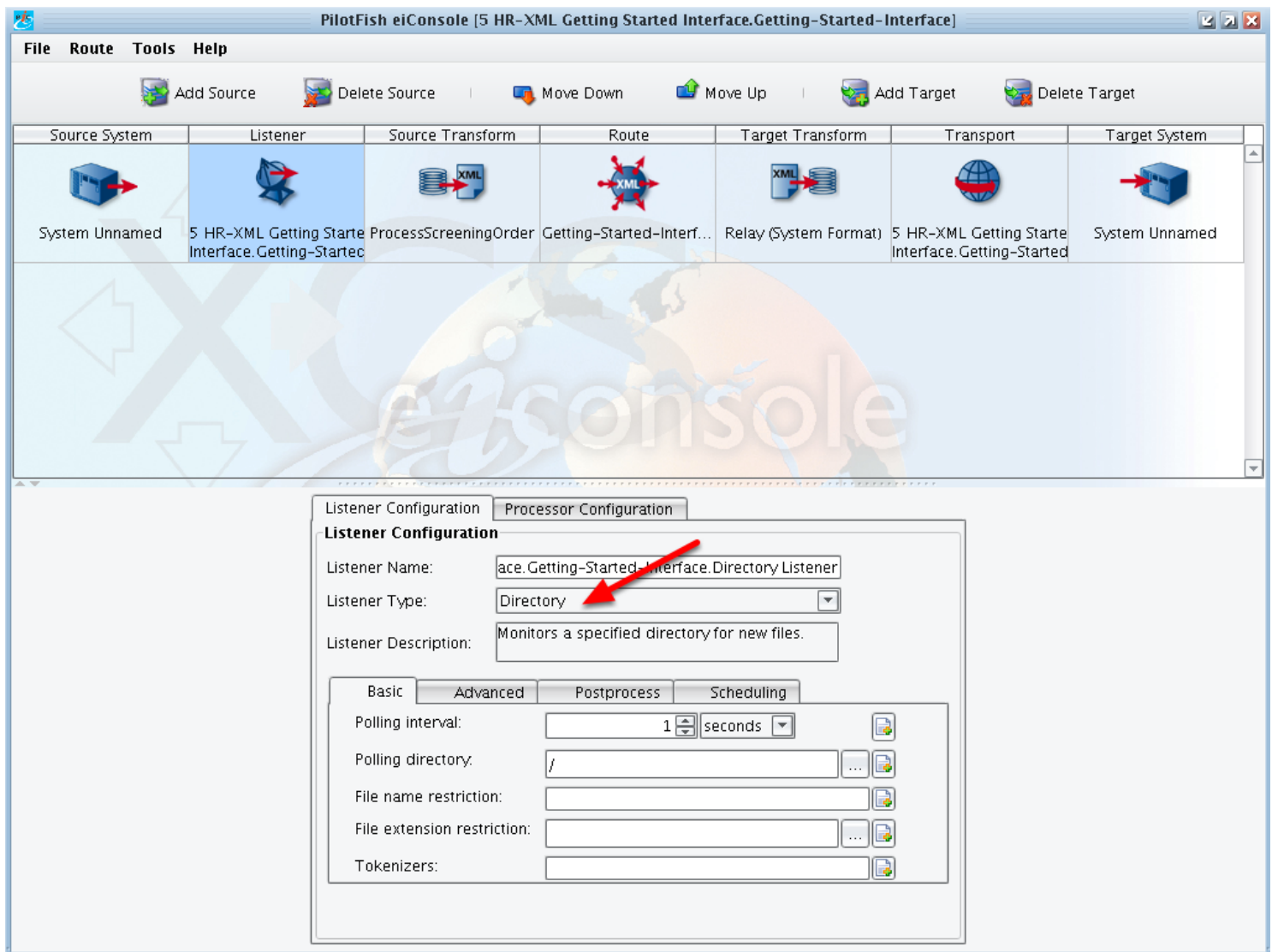
When you open the package you will see a configured Route/Interface. For this tutorial you'll be tweaking this Route, the **Getting Started Interface** message. Right click on the row and select **Edit Route** from the dropdown.



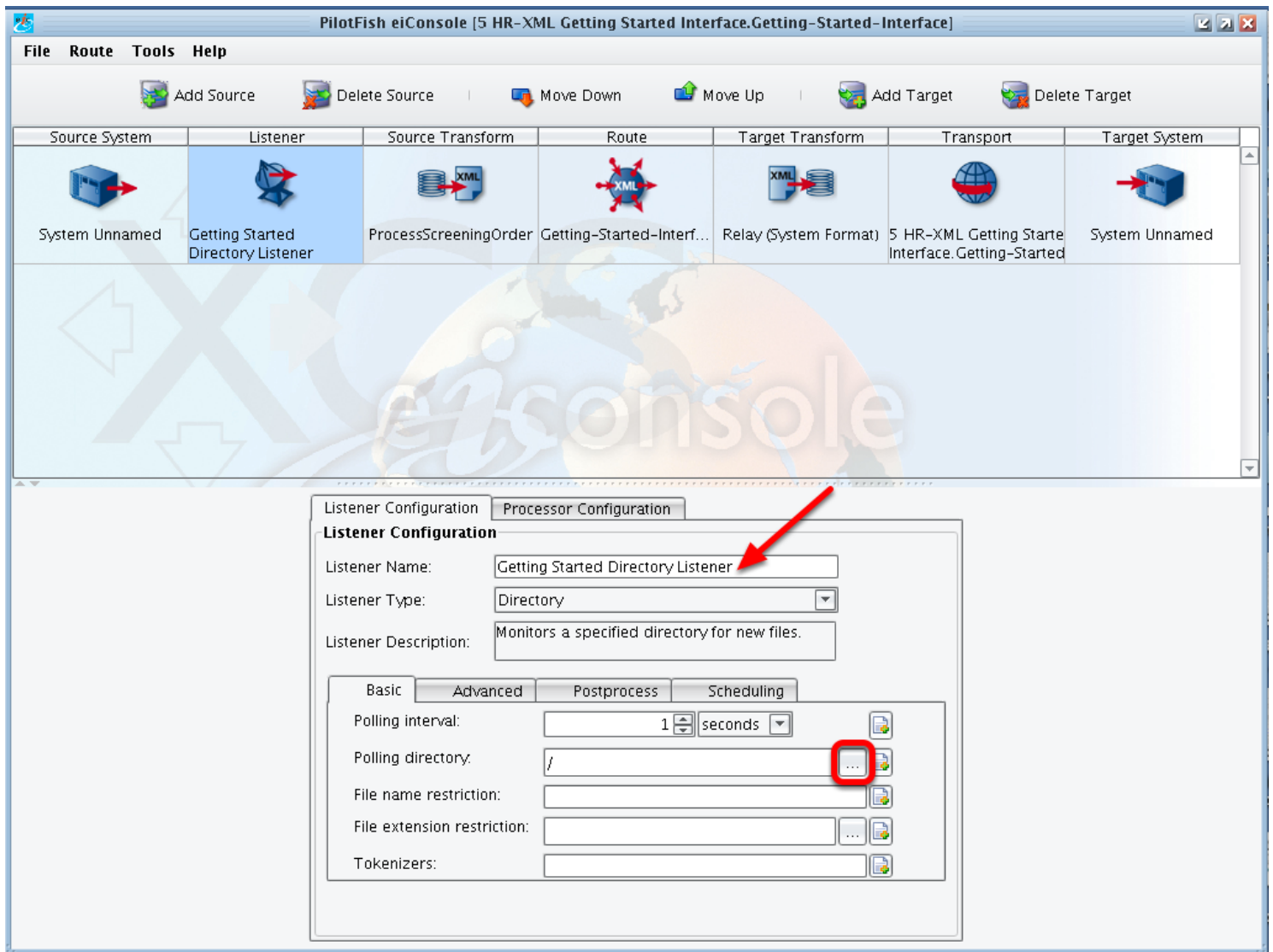
This will bring you to the Main Route grid. The main route grid depicts the flow of data from a Source System to a Target System. Your job will be to configure each one of the stages in between including the:

- Listener: handles the connectivity to the Source System;
- Source Transform: from a proprietary format into your HR-XML canonical model;
- Route: any routing rules, if you had multiple targets you could make a decision as to which data should be sent to which Target systems;
- Target Transform: this allows you to further transform data into a format that the Target System can consume;
- Transport: this allows you to handle the connectivity to the Target System. You'll begin with the Listener.

Click the **Listener** stage in the main route grid.



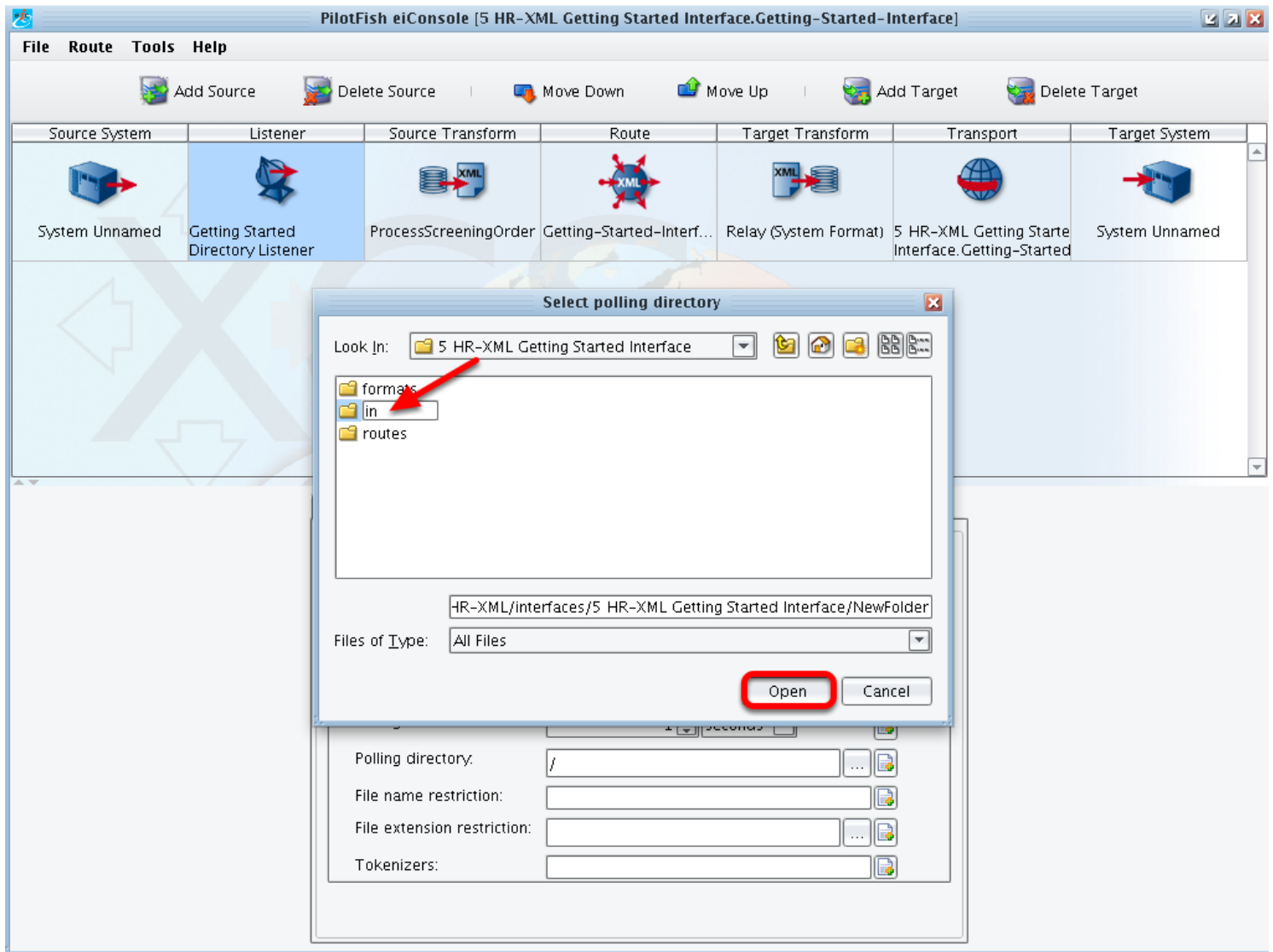
Each time a stage in the upper grid is selected, a set of configuration panels will appear below. With the Listener selected, you'll see the Listener Configuration area. This is where you choose how you expect the Source System to send data. You would do this by selecting the appropriate Listener type from the Listener Type dropdown. Here you see this template has already specified a Directory Listener. This means that the system will poll a directory at a specified interval and look for a particular file.



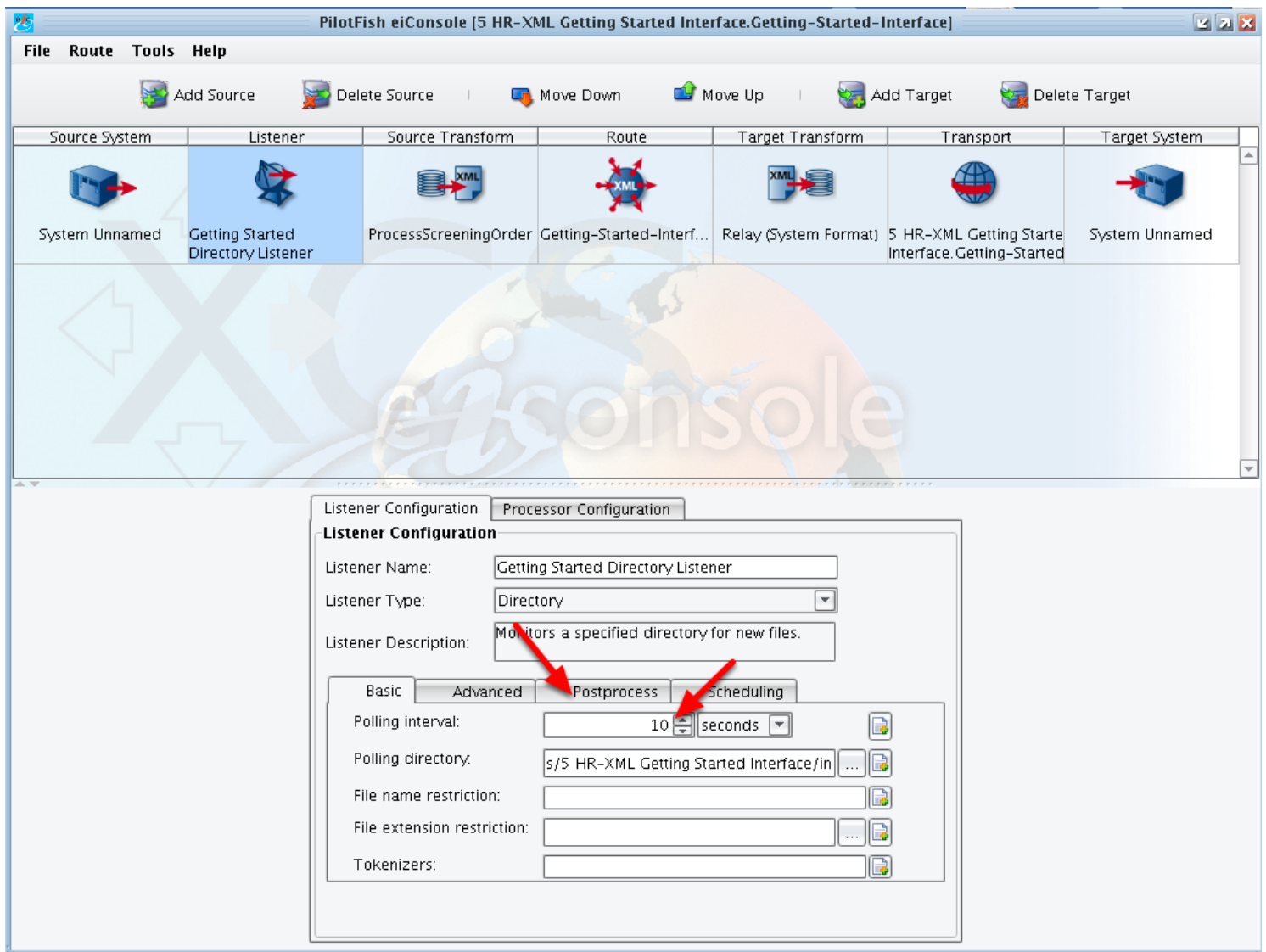
Once the configuration panel opens edit the **Listener Name** to read **"Getting Started Directory Listener"** or give it some other meaningful name.

**Note: within a Working Directory you cannot use the same Listener Name for two different routes. This creates ambiguity in the system. So for example, if you tried to name your Listener "Source Directory Listener" you would be prompted to enter another name because Source Directory Listener is already being used in the 2 HR-XML Getting Started Tutorial sample interface.**

In this case you'll keep the Listener as a Directory Listener, but you'll change the Polling directory. To do this, first click the red **Elipsis** button next to the **Polling directory** configuration item.

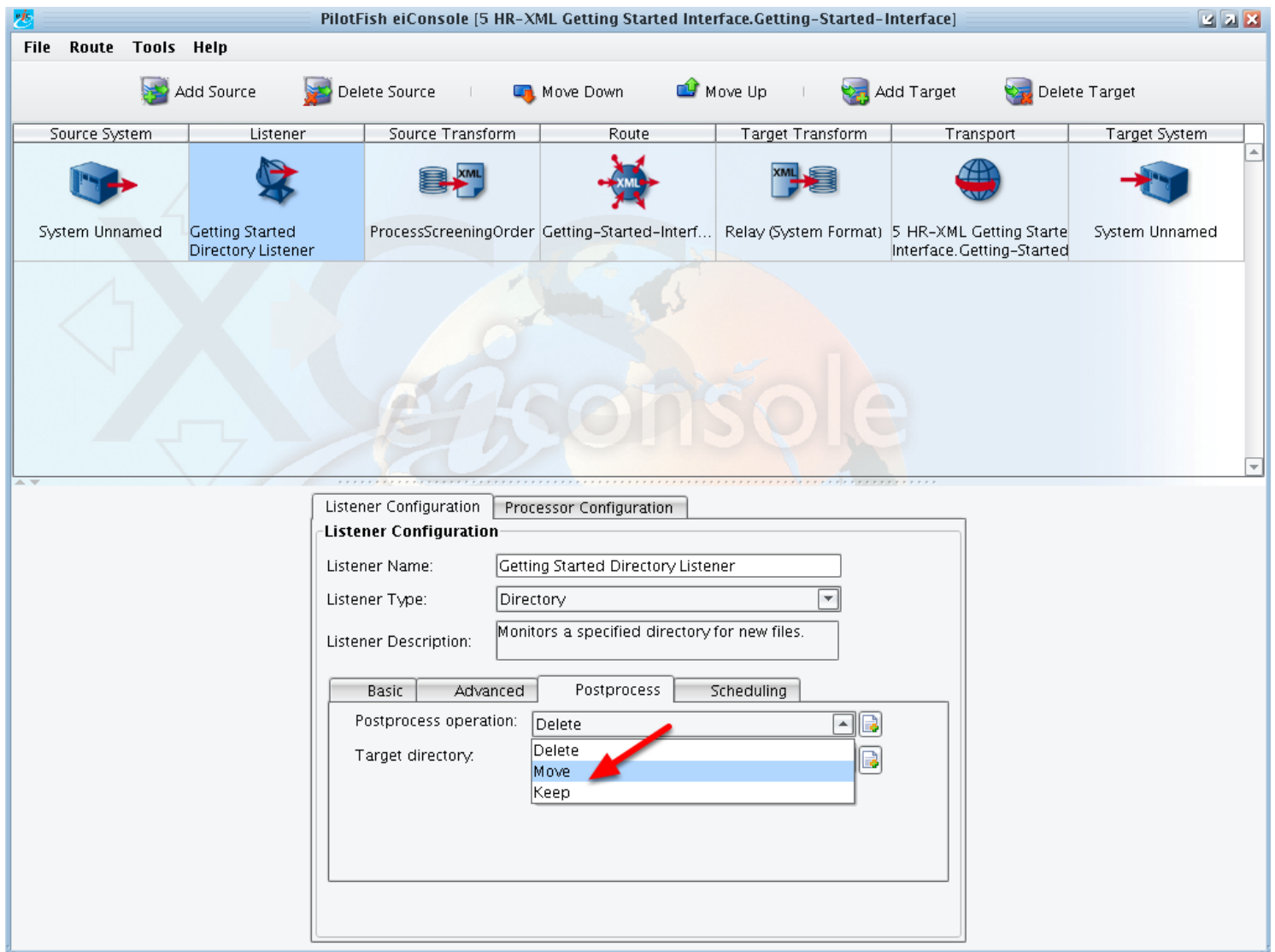


Which will bring up a file selection dialogue. Navigate to the **5 HR-XML Getting Started Interface** folder. Create a new folder called **"in"**, select it and click **Open**.

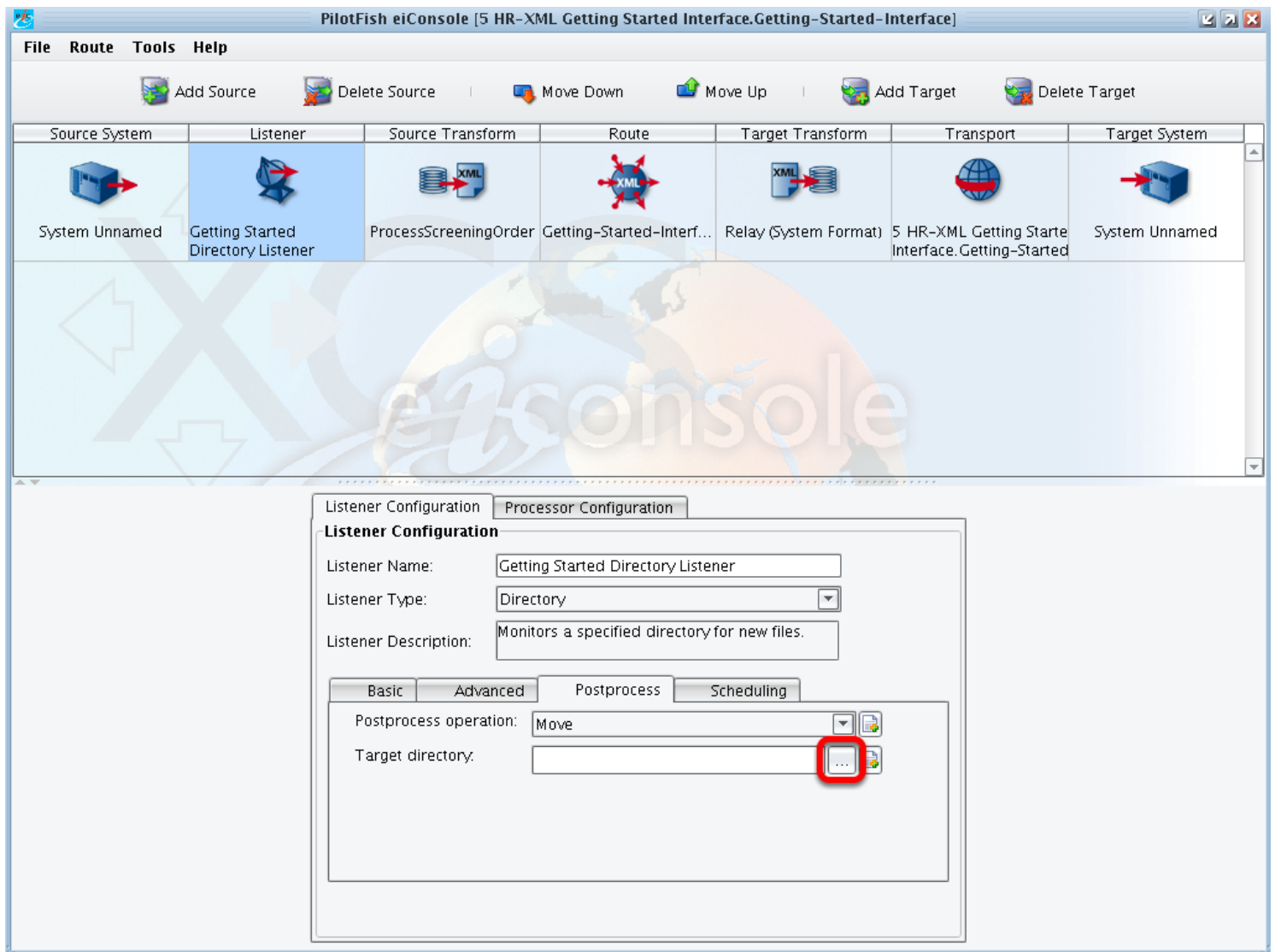


Now, when this Listener runs, it will poll the newly created drop directory every 1 second for any input file that you provide. Next change the polling interval. Select the current **Polling interval** of 1 second and enter the value **"10"**. You will also want to configure how the system behaves after it picks up the file. To do this, click on the **Postprocess** tab in the configuration area.

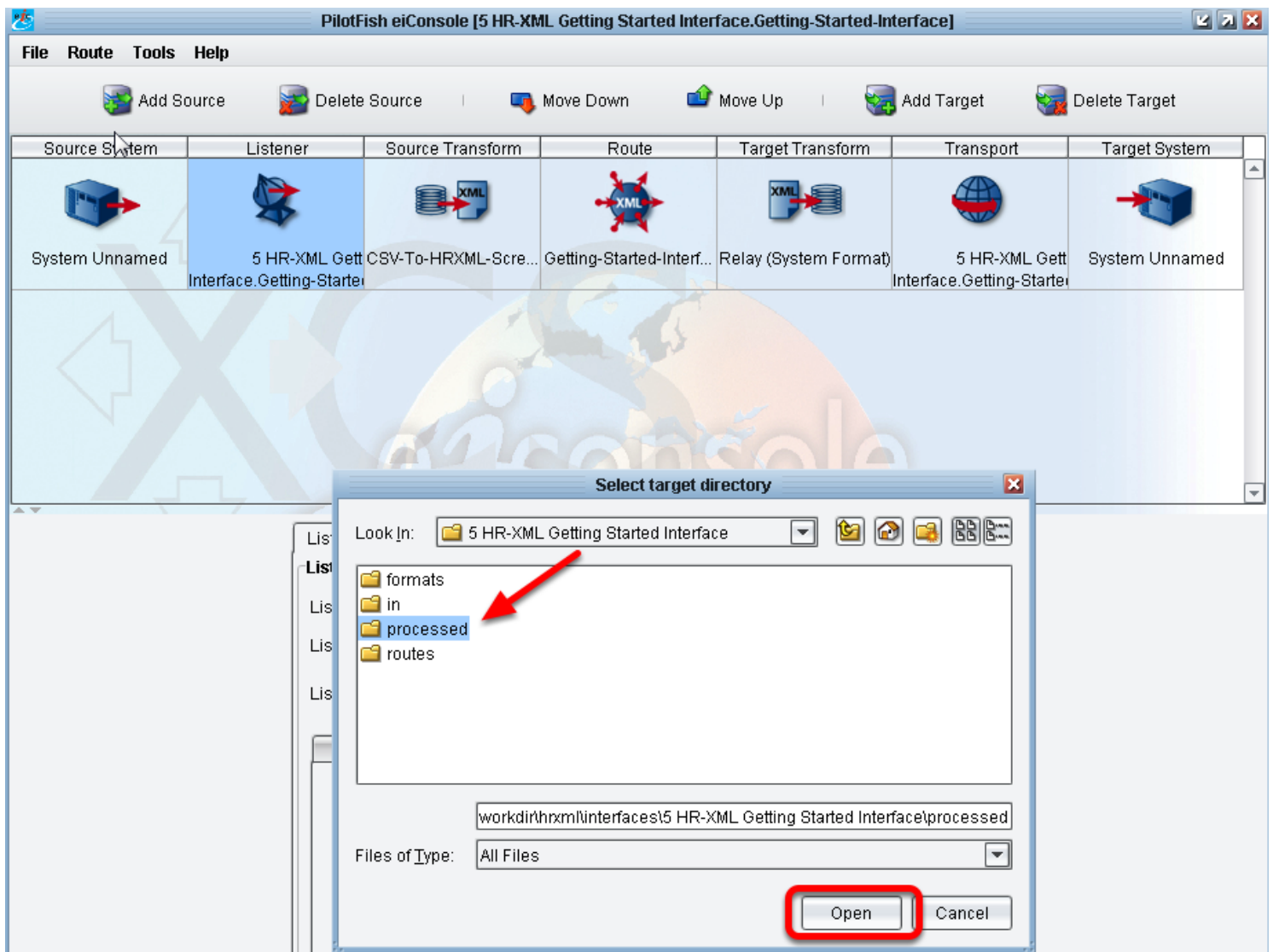




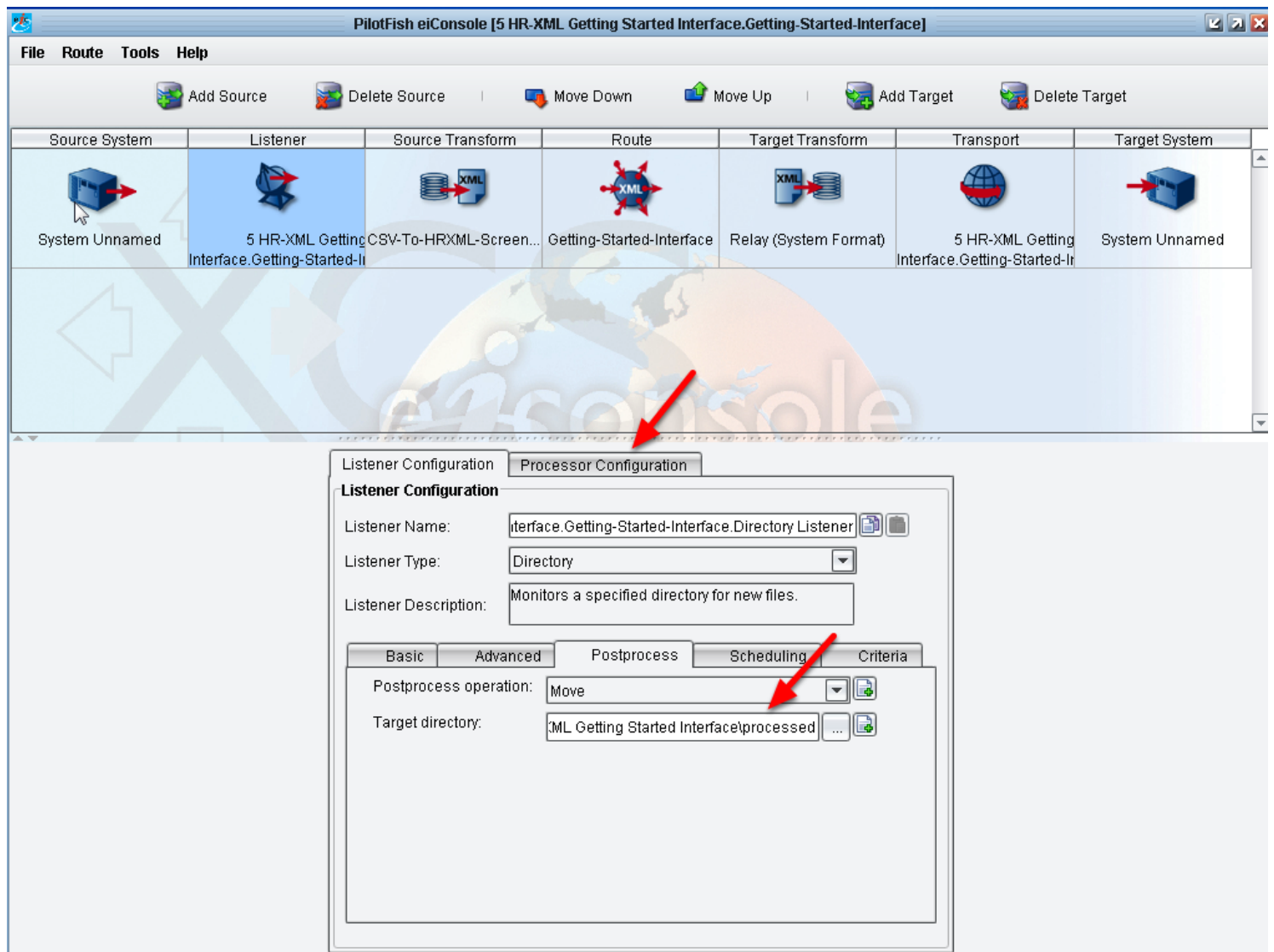
You'll note that the current Postprocess operation is to Delete the file. However, let's say you wanted to keep a backup, so you'd select **Keep**. Here, we will want to select **Move** from the Postprocess operation dropdown.



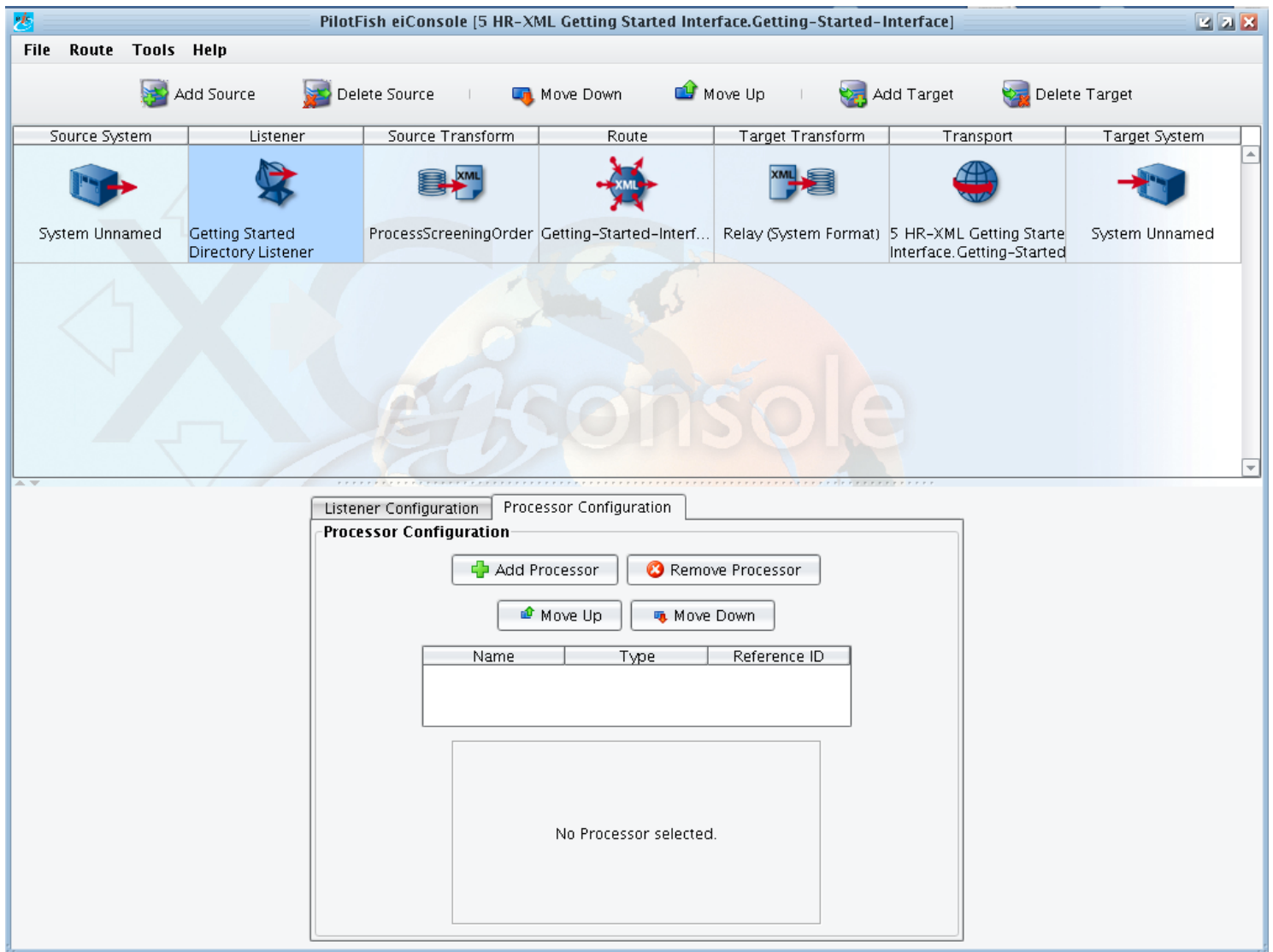
Now we will want to specify the Target Directory. To do so, click the **Elipsis** button.



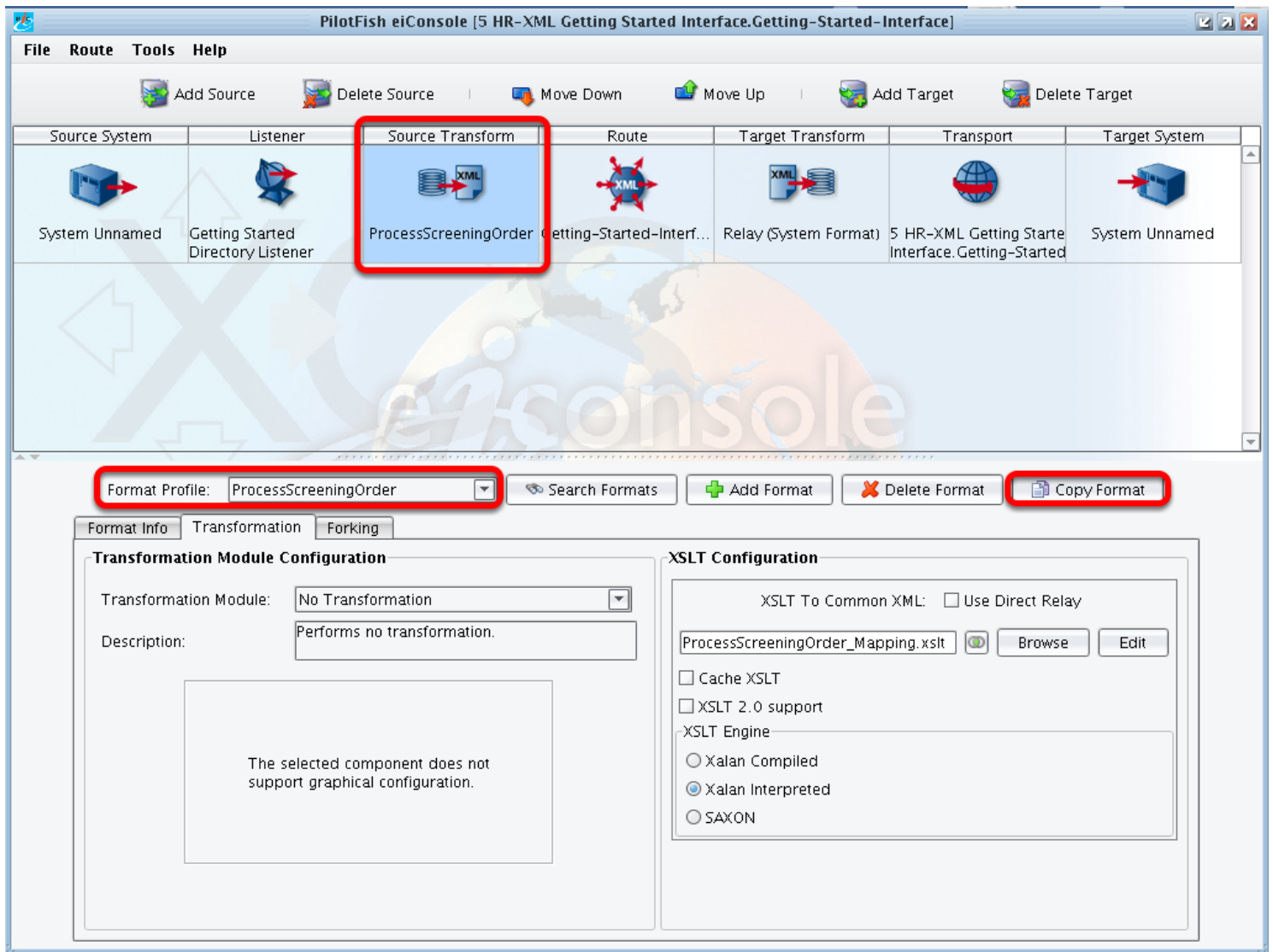
Once again this opens up a file selection dialogue. In the same folder as before create a new folder and name it **"processed"**, then click **Open**.



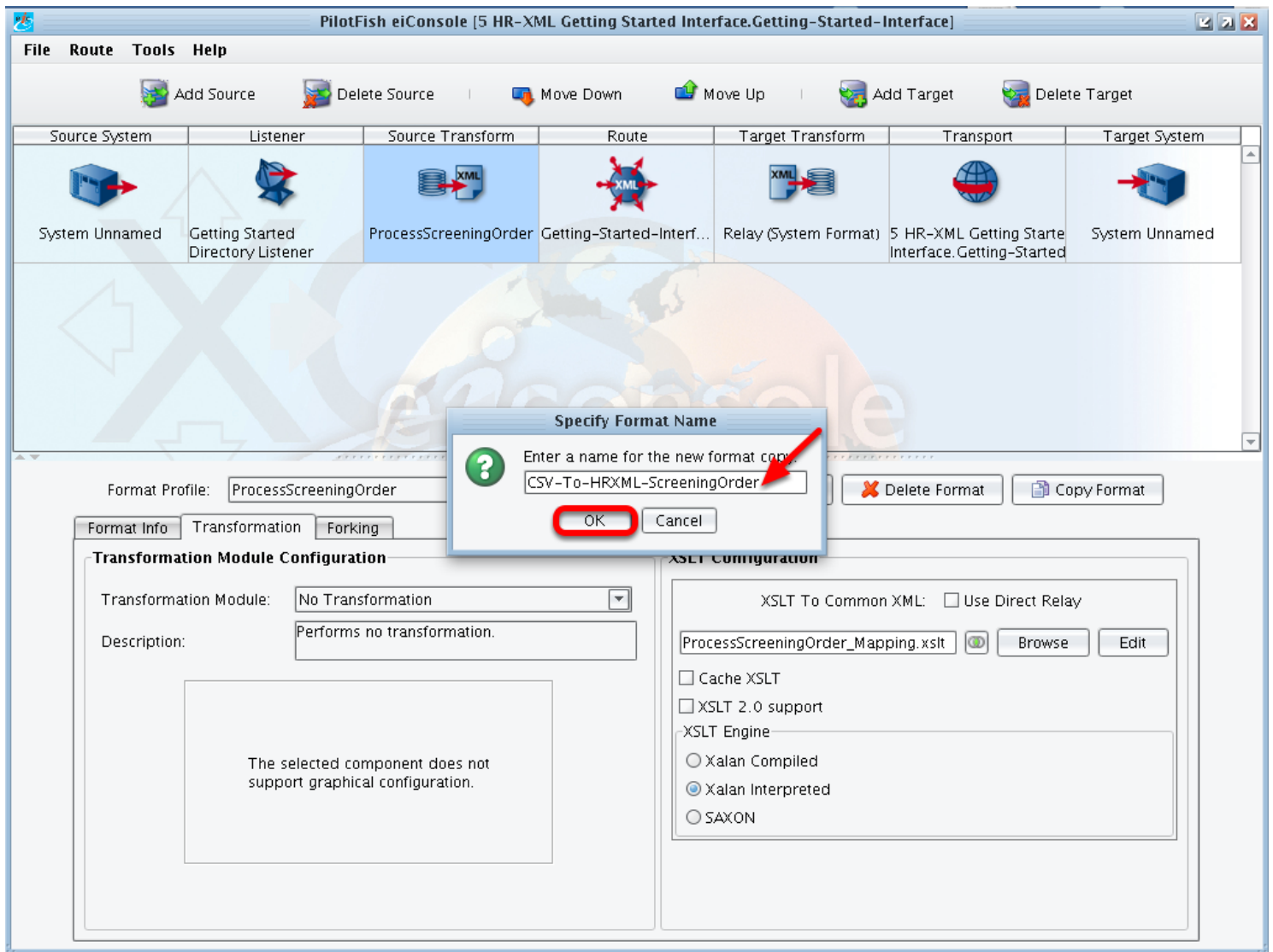
Now, as files are processed out of your in folder, a copy will be moved to the out folder. Let's move on. Select the tab next to the Listener Configuration tab, the **Processor** Configuration tab.



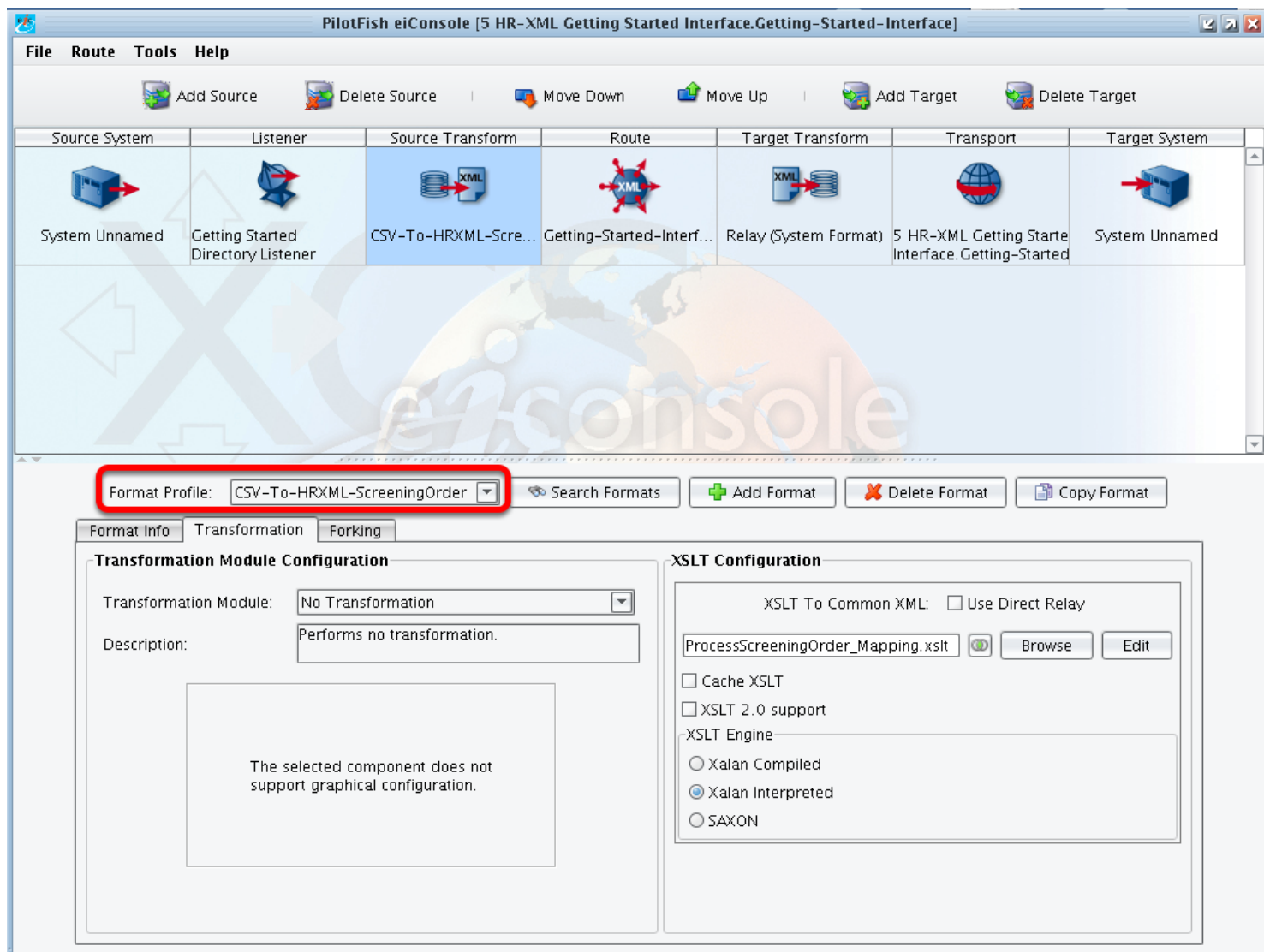
Processors allow you to do general work with the data stream using a collection of widgets shipped with the eiConsole. In this Route, none of this processing is necessary and we will move on to the next stage.



Now click on the **Source Transform** stage in the main route grid. You'll see below that the format ProfileProcessScreening order has been created. This ProfileProcessScreening order format profile is a reference to the existing template for the screening order. Because you don't want to modify the template, but instead want to work off of a copy, you'll need to create a copy of this format. Click the **Copy Format** button to the right of the Format Profile dropdown.

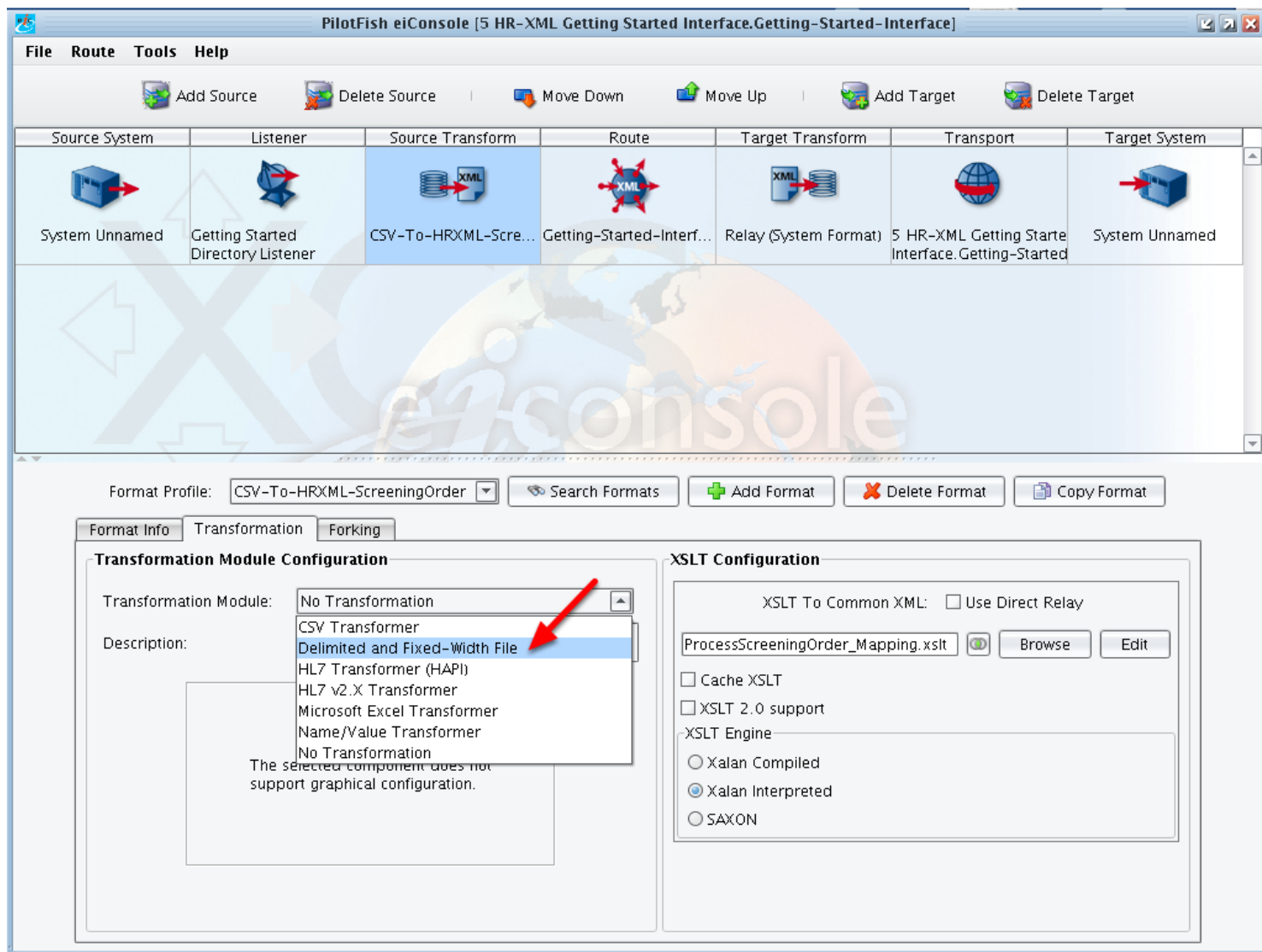


When prompted to provide a name we will call it **"CSV-To-HRXML-ScreeningOrder"** and then click **OK**.



You'll note that the Format Profile selected will change to the newly created format. In the Transformation tab you'll see 2 subsections. The Transformation Module Configuration and the XSLT Configuration. The Transformation Module is used to convert the data from its inbound format into XML if it hasn't yet been translated into an XML representation. The eiConsole supports this translation through a variety of Transformation Modules. The most commonly used, and the one you'll use in this case, is the Delimited and Fixed-Width File Transformation Module.



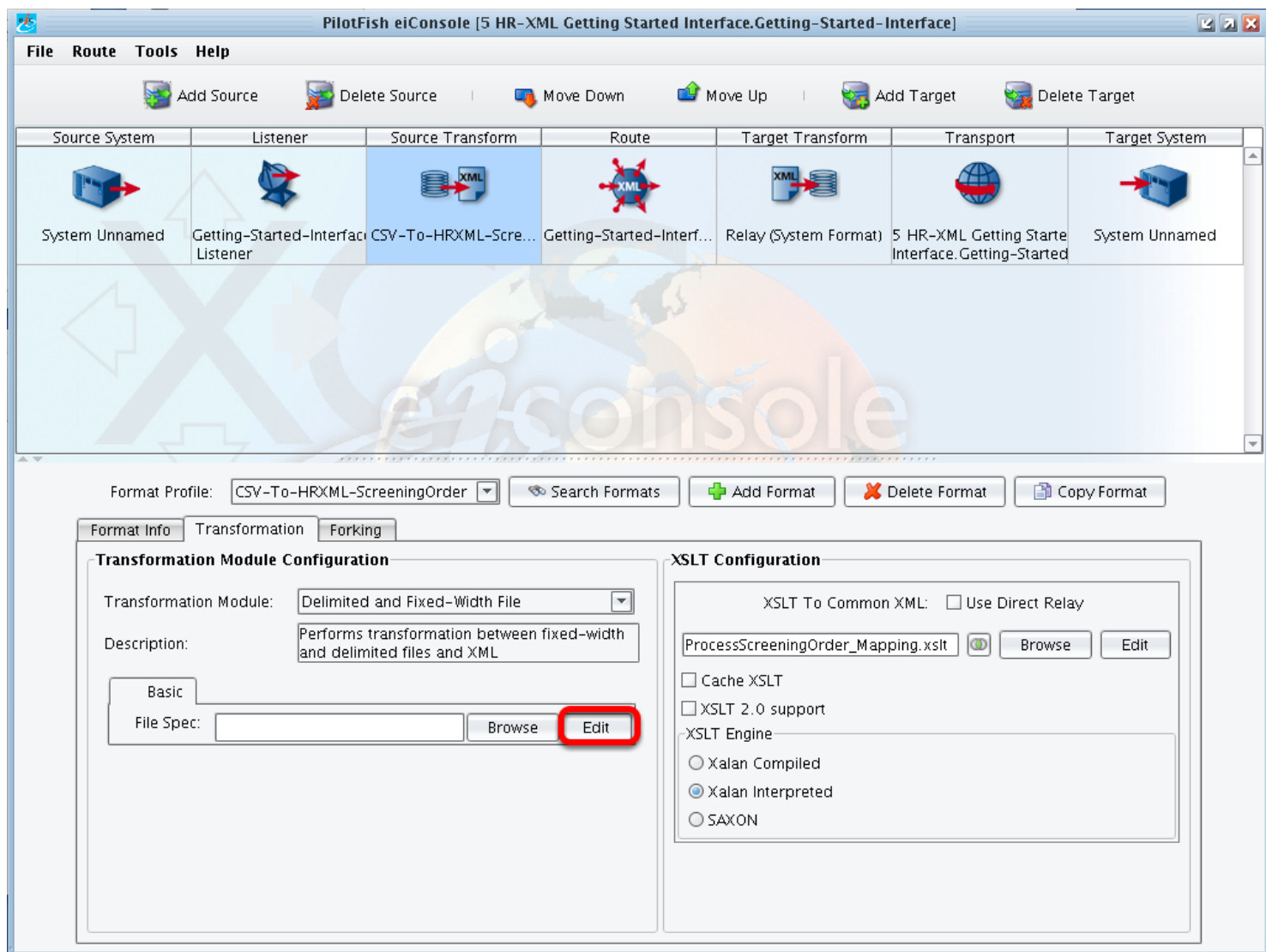


So you'll want to select the **Delimited and Fixed-Width File** Transformation Module from the dropdown.

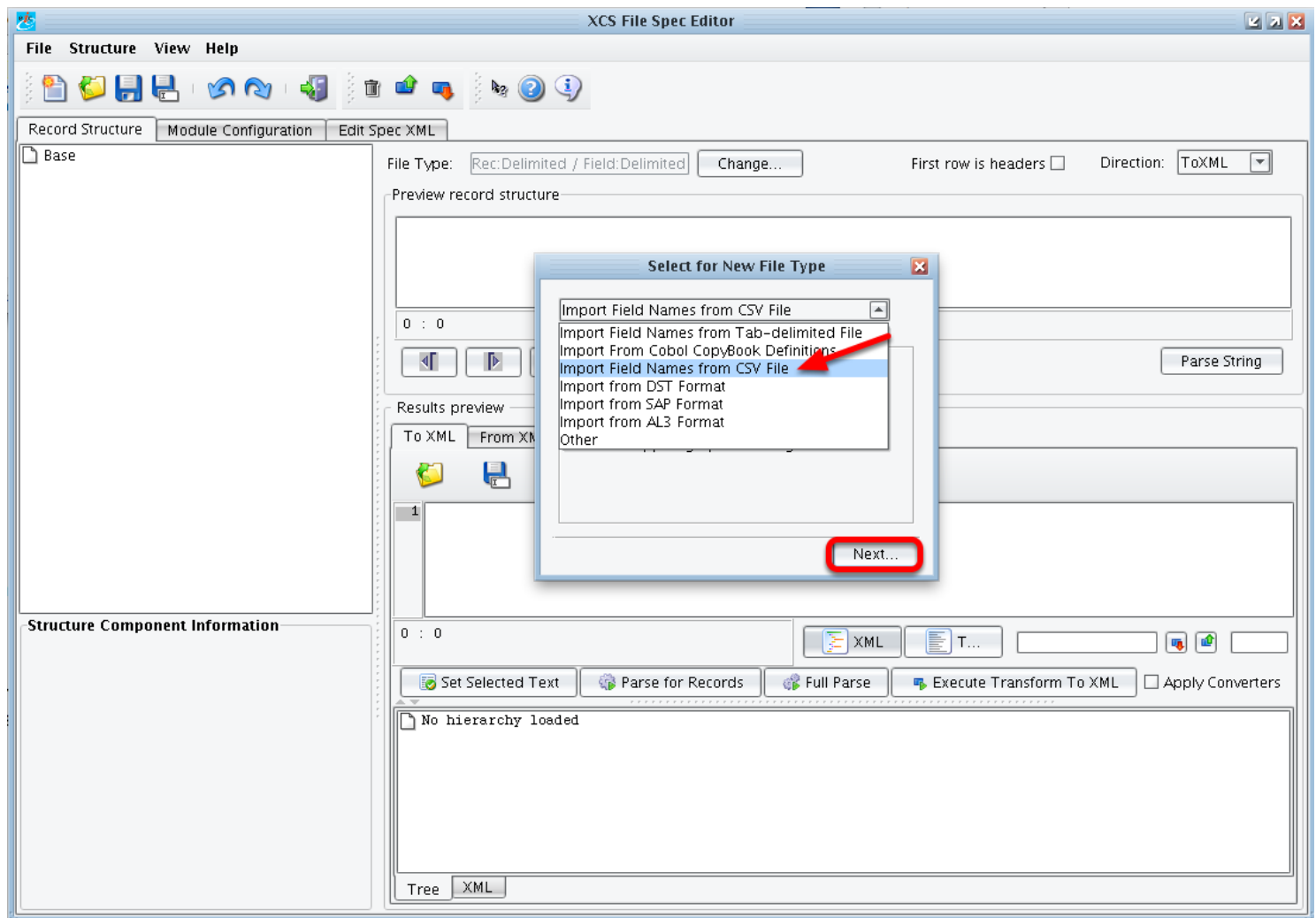
```
File Path ▾ : /Applications/PilotFish Technology XCS eiConsole.app/Contents/HR-XML/interfaces/2 HR-XML Getting Started Tutorial/in/candidates.csv
candidates.csv
1 | FirstName,MiddleName,LastName,Addr1,Addr2,City,State,ZIP,DOB,Credit,Criminal
2 | Mick,E,Moose,1 Castle Court,,Orlando,FL,32801,11/18/1928,X,
3 | Nick,N,Pole,1225 Main Street,,Santa Claus,IN,42759,2/17/1928,,X
4 |
1 1 | (none) ▾ Unicode (UTF-8) ▾ Windows (CRLF) ▾ Last saved: 6/26/13 9:44:36 AM 201 / 38 / 4
```

The Delimited and Fixed-Width File Transformation Module will allow you to convert data from a delimited or fixed-width file format into an XML representation that can be further transformed. In this case, the input will be a csv file. Before you go any further, let's take a look at that file. You'll find the file **candidates.csv** in the sample tutorial, **2 HR-XML Getting Started Tutorial** within the "in" folder or in the 3 HR-XML Templates (\interfaces\3 HR-XML Templates\data\tutorial\input). Open this in a text editor.

When you open this file you'll see it's relatively simple. It's a csv file where the first row indicates column headers and each row thereafter is a data row including some basic information about the candidate. The last two columns are used to indicate whether you want to perform a credit check or criminal background check for each of these candidates.

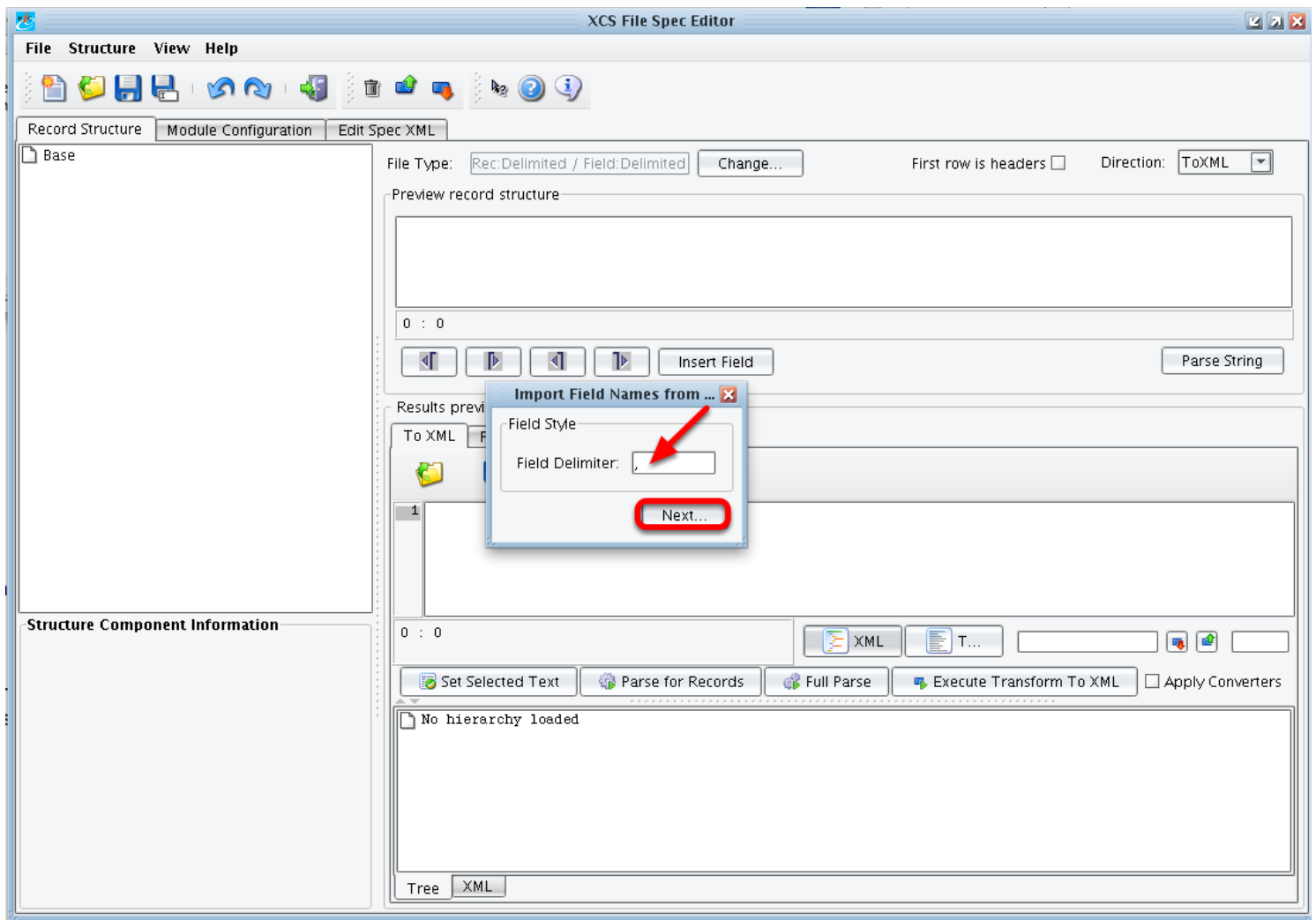


To describe the structure of the file to the eiConsole click **Edit** next to the File Spec configuration item in the Transformation Module Configuration section.

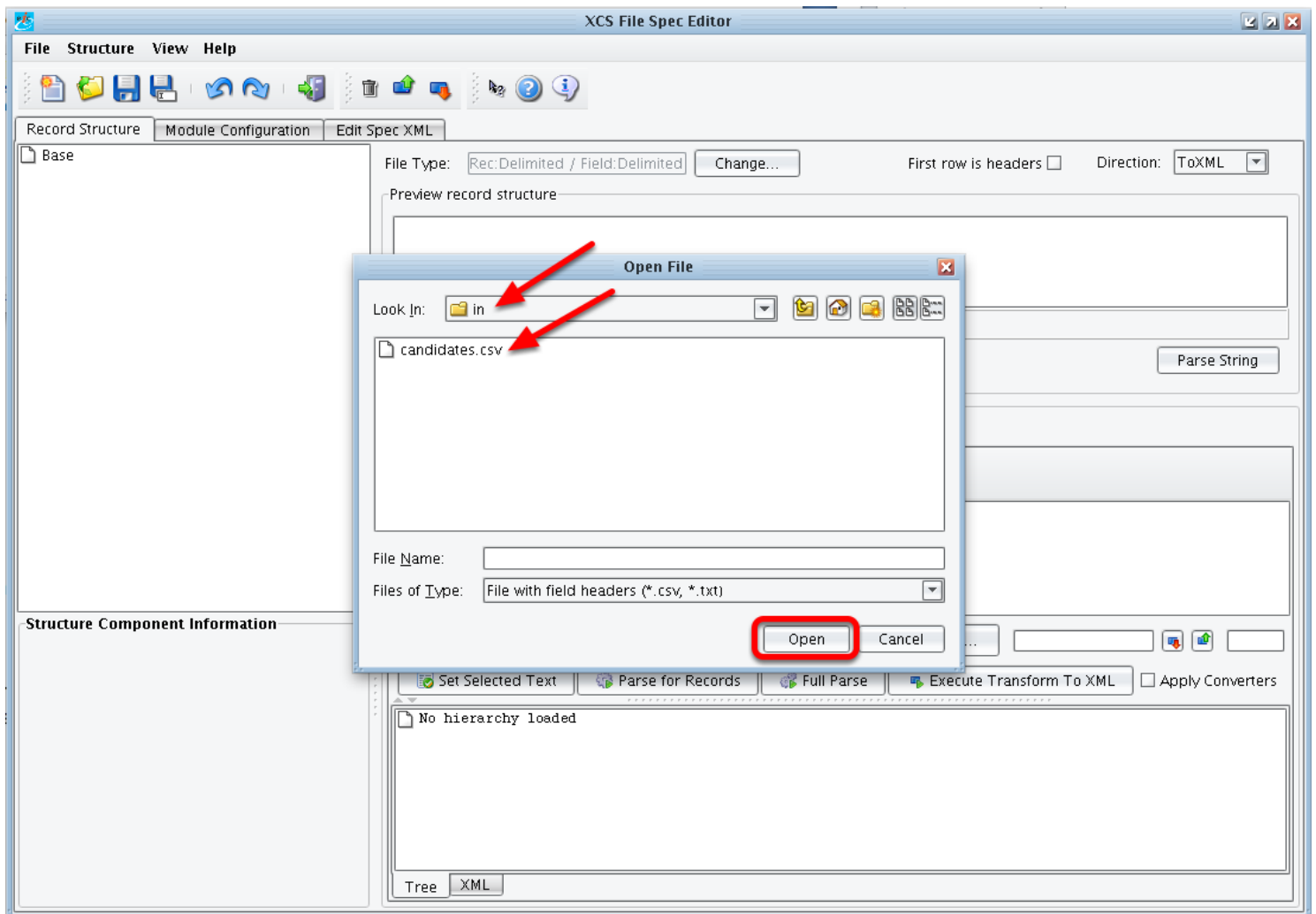


This will launch the XCS File Specification Editor. It will also launch the new file specification wizard. In the Select for New File Type dropdown you'll need to select **Import Field Names from CSV File**.

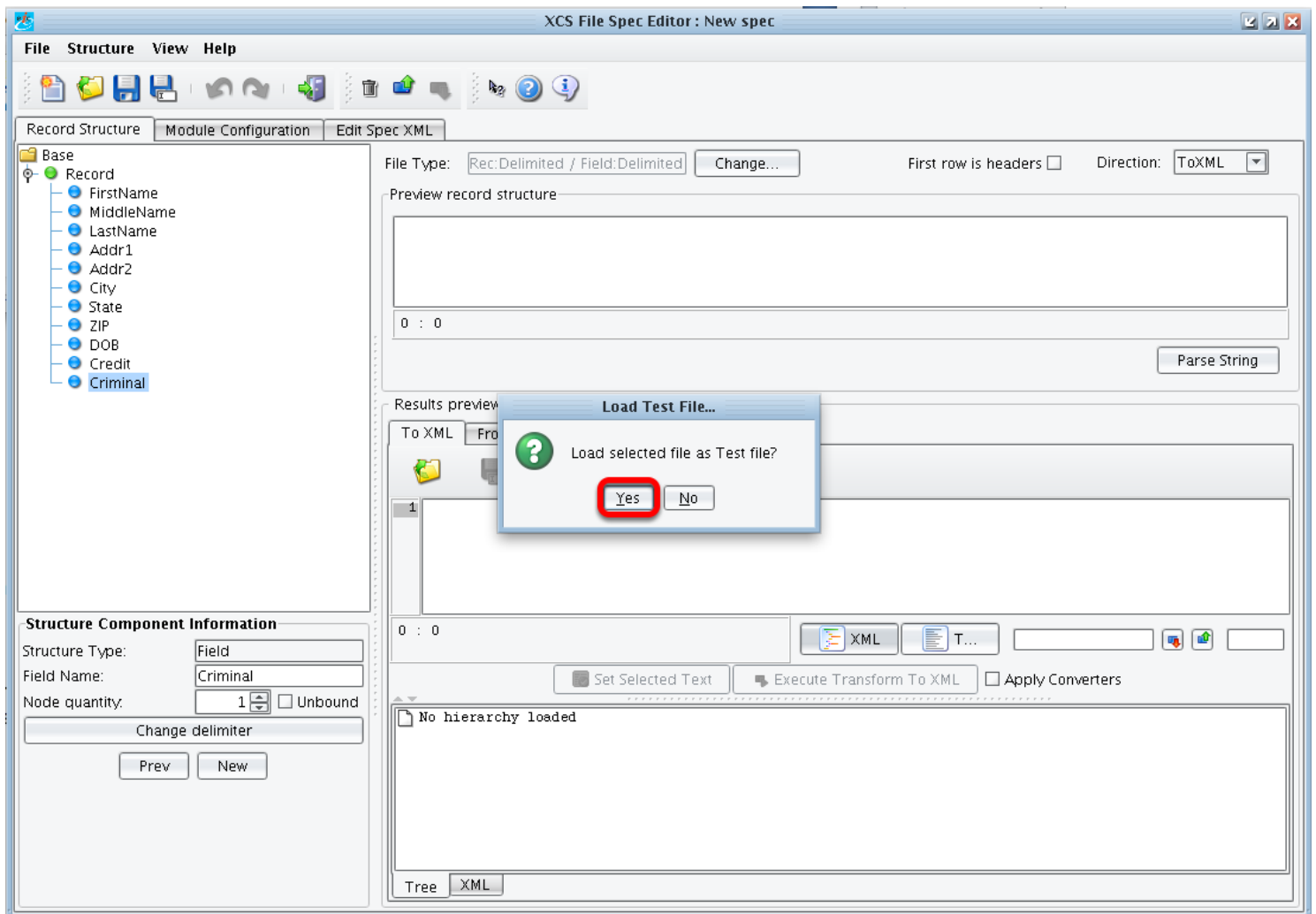
Click that entry in the dropdown and then click the **Next** button.



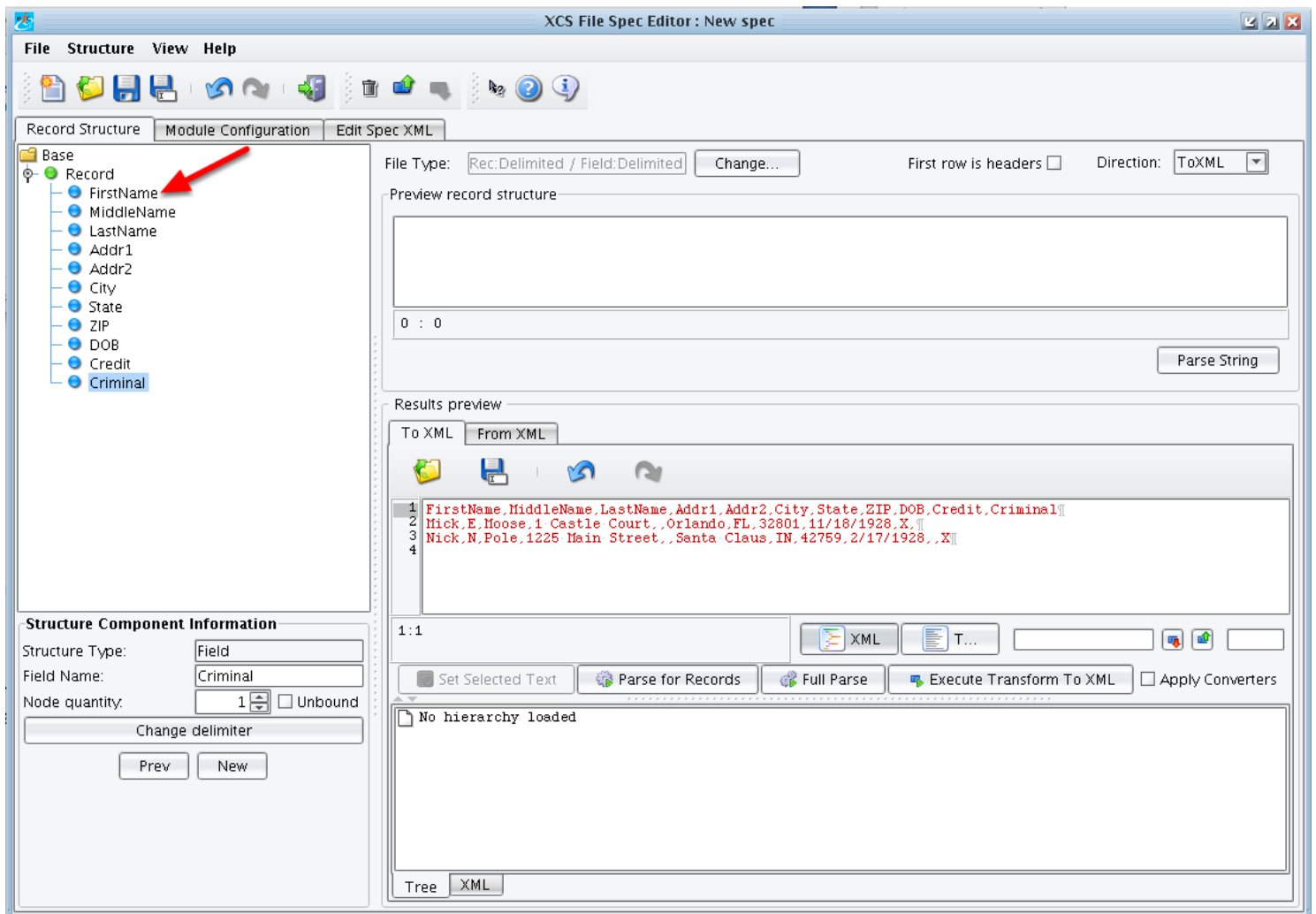
You'll see that the Field Delimiter is correctly set to a comma, so then click **Next** again.



Next you'll need to provide the sample file from which it will read the format. Once again, navigate to the **2 HR-XML Getting Started Tutorial** open the **"in"** folder and then select the **candidates.csv** file and click **Open**.



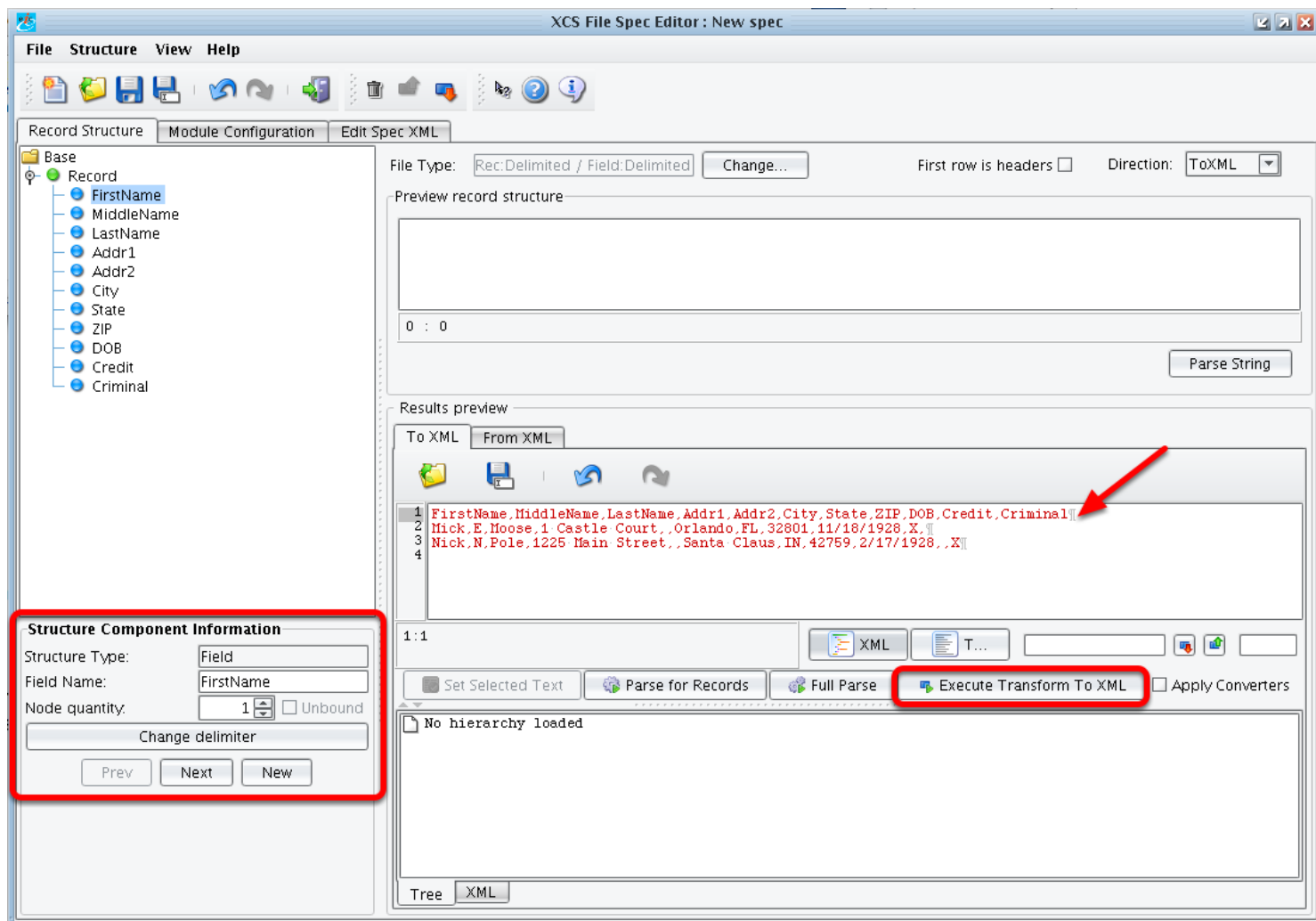
When prompted to **Load Test File** as the Test file, click **Yes**.



You'll notice that the left hand side of the screen is now populated with a tree that represents the format of that csv file. You have one Record type denoted by a green node that's name has been defaulted to record. You then have a number of different fields, each represented as a blue node in the list.

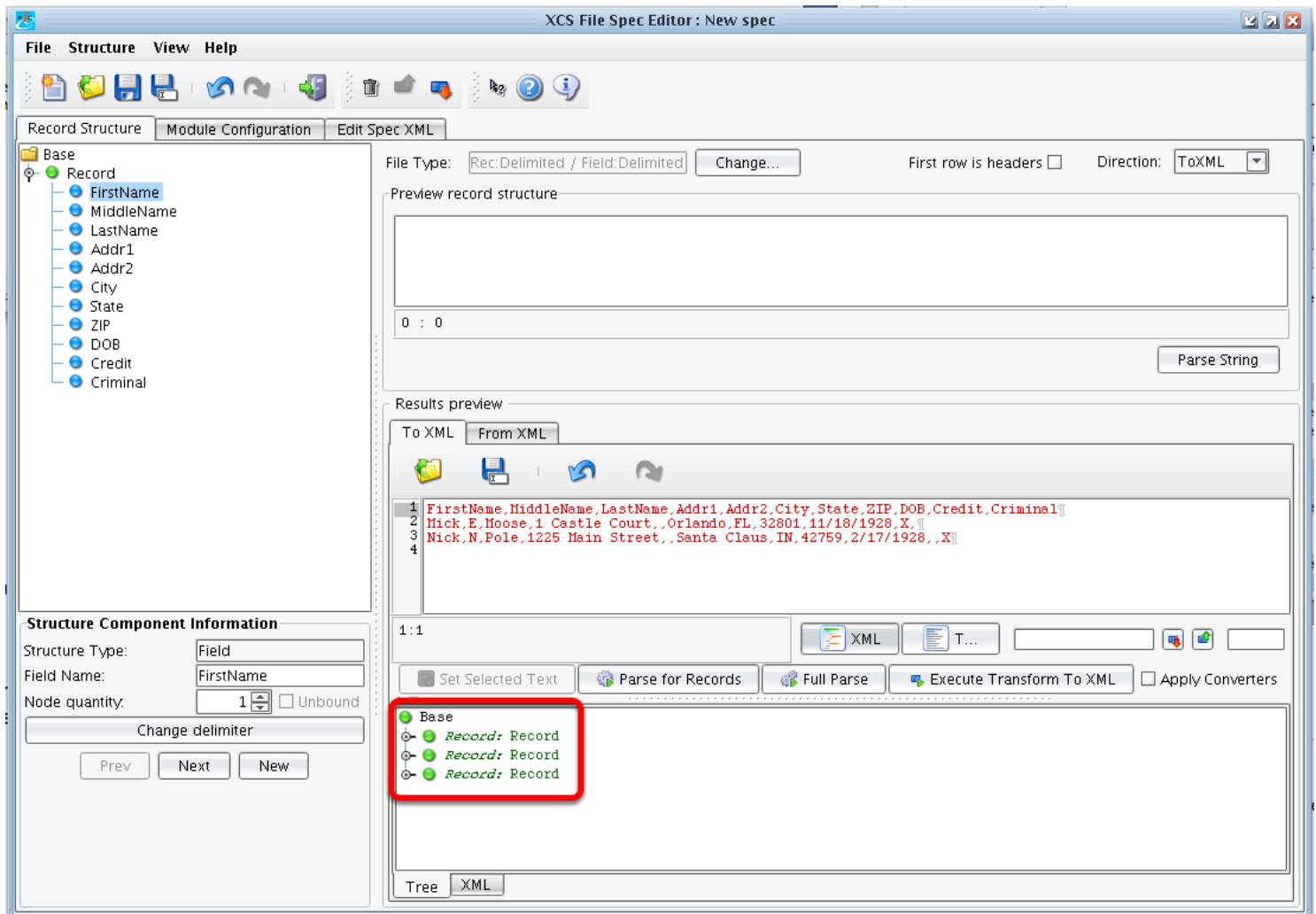
Select the **First Name** node.



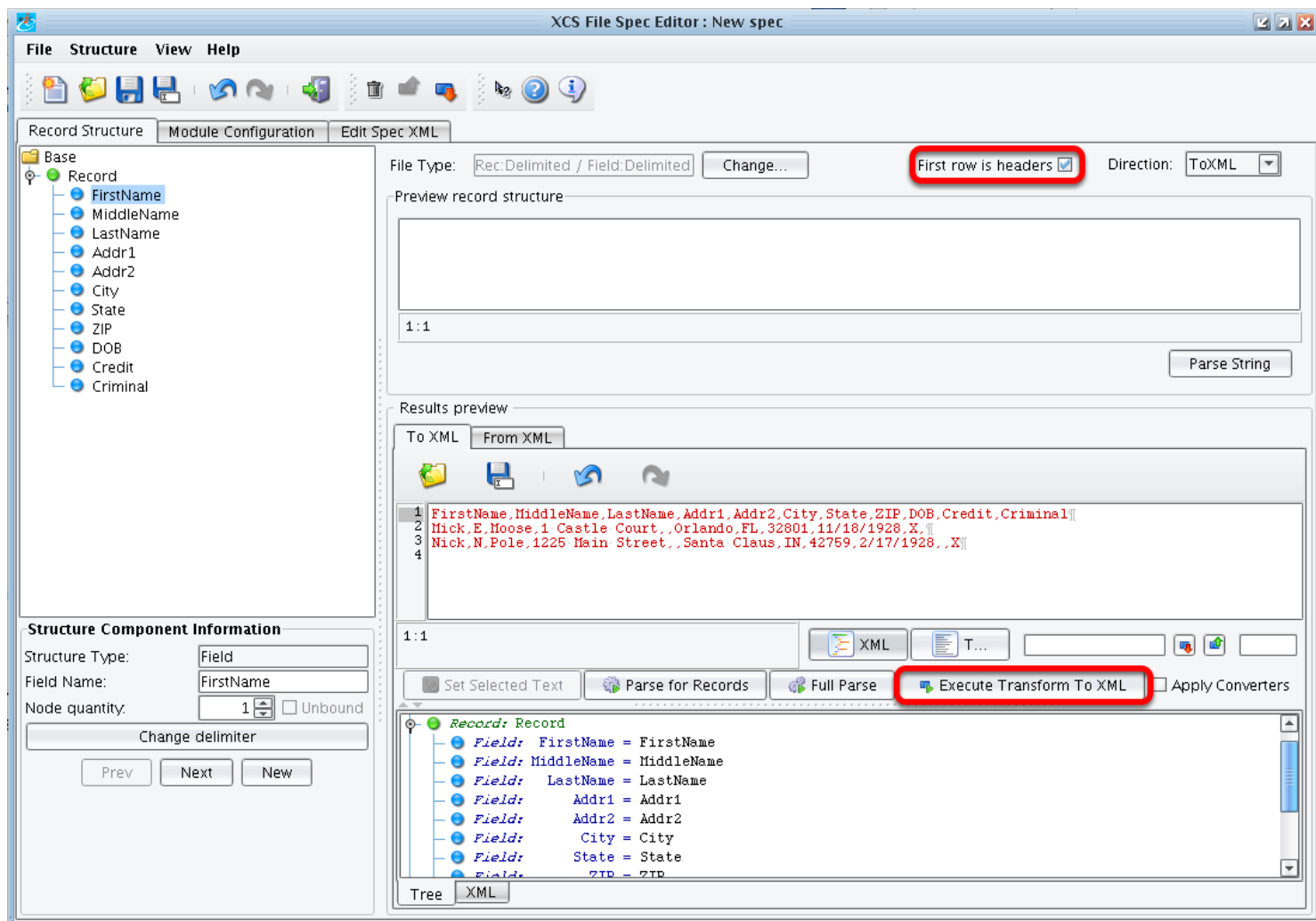


Details about the field appear in the panel underneath the tree.

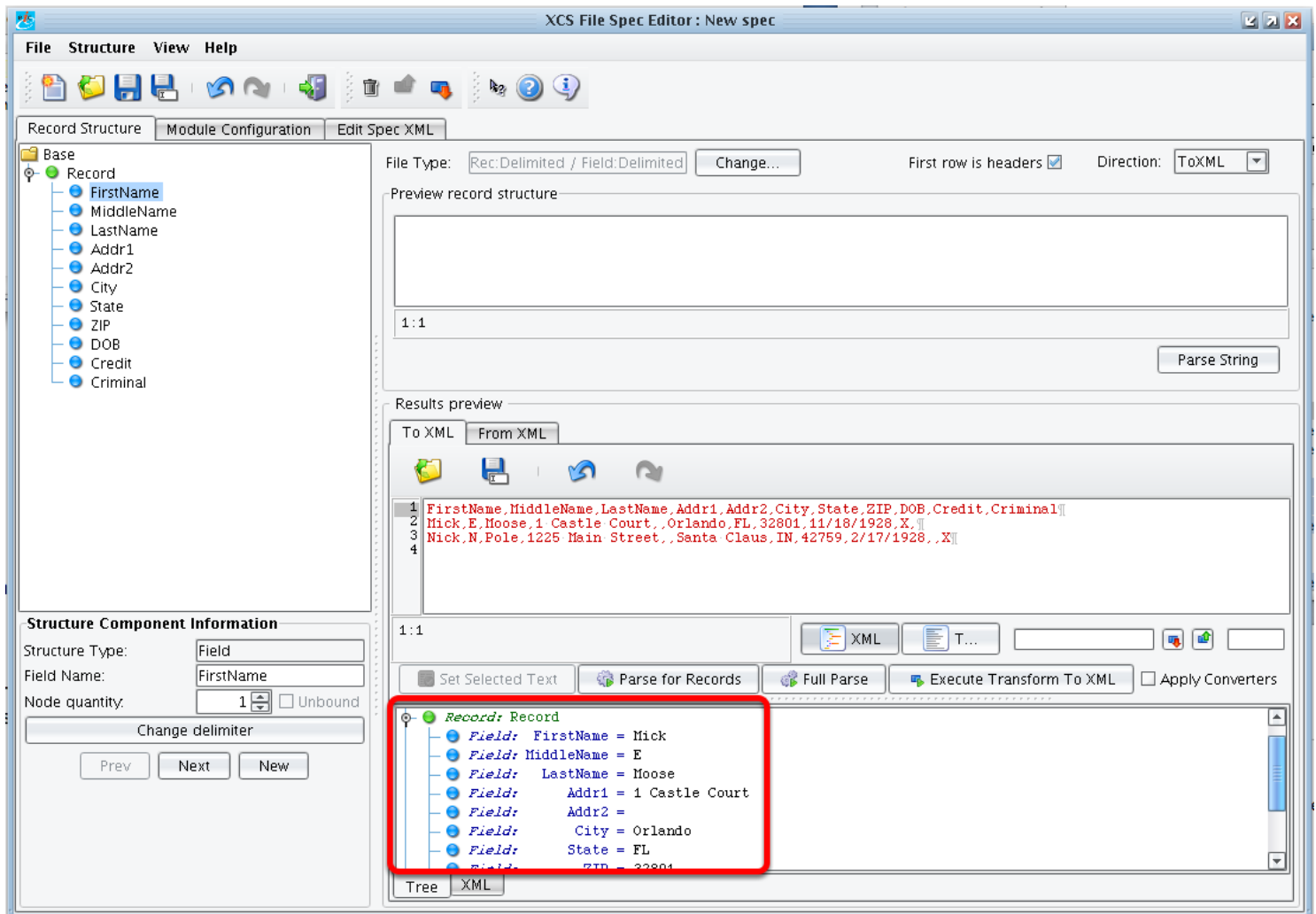
The sample file appears in your **Results preview** window. To preview the transformation of this data into a parsed structure, click the **Execute Transform to XML** button.



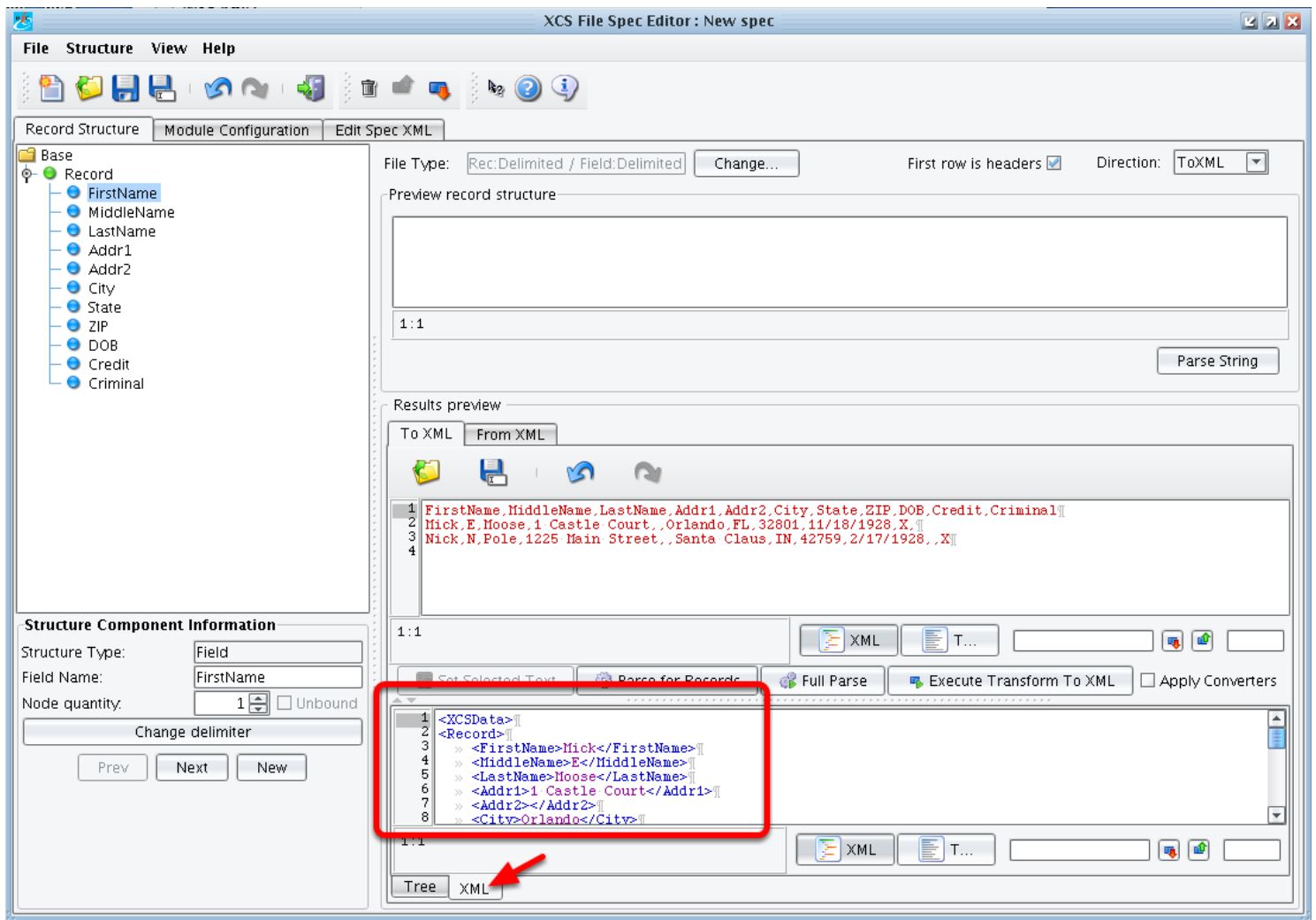
You'll see a new tree appear directly underneath the Results preview area. This tree can be expanded to show how the system parsed the sample file above. Click the nodes to expand the tree.



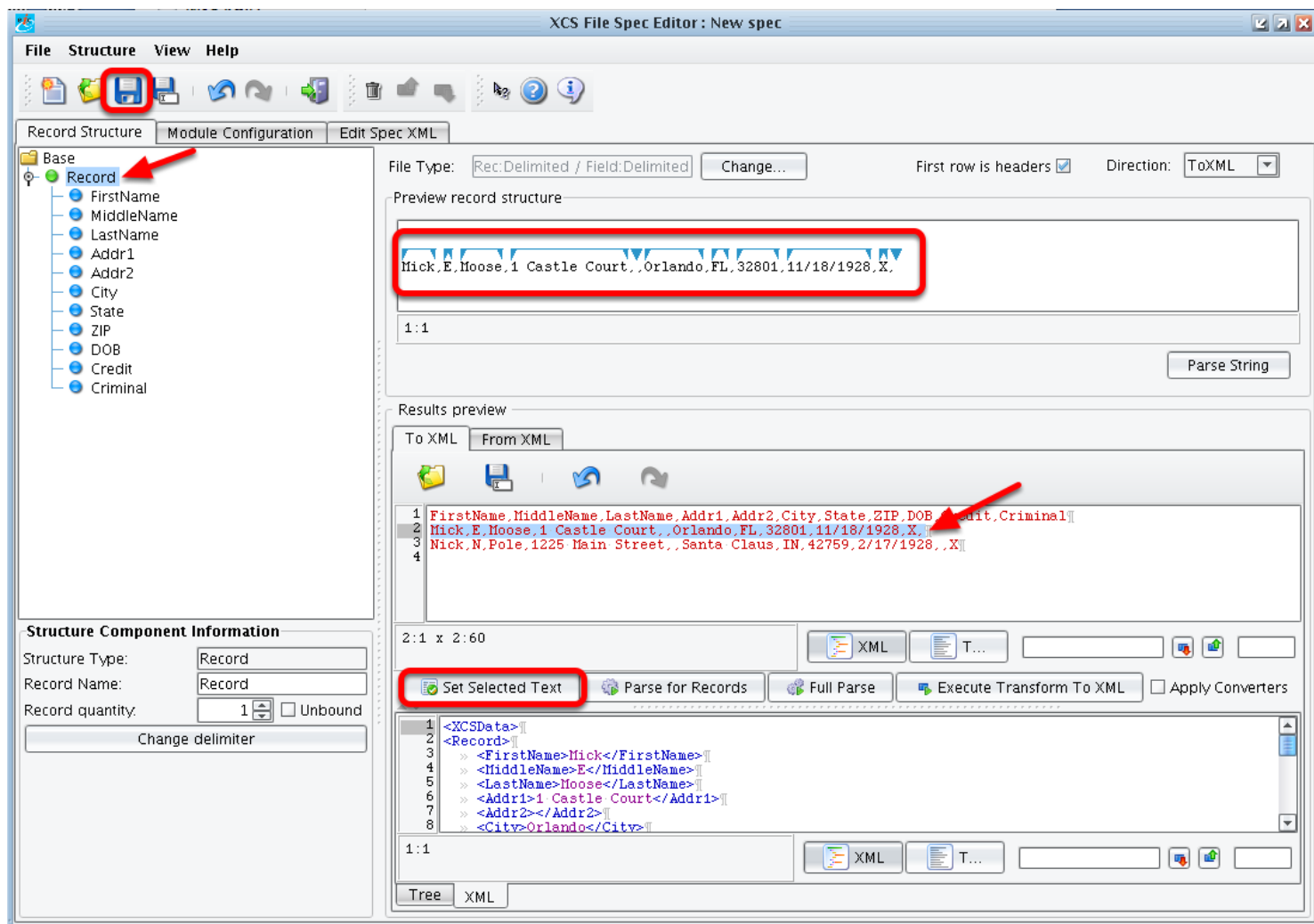
You'll see that the system has incorrectly identified the first row here as data. Instead, this should be ignored as it contains field headers. Check the **First row is headers** checkbox at the top of the screen. And then click **Execute Transform to XML** again.



Click the nodes to expand the tree. You'll see that now the system has correctly found and parsed the two records contained in your data file.

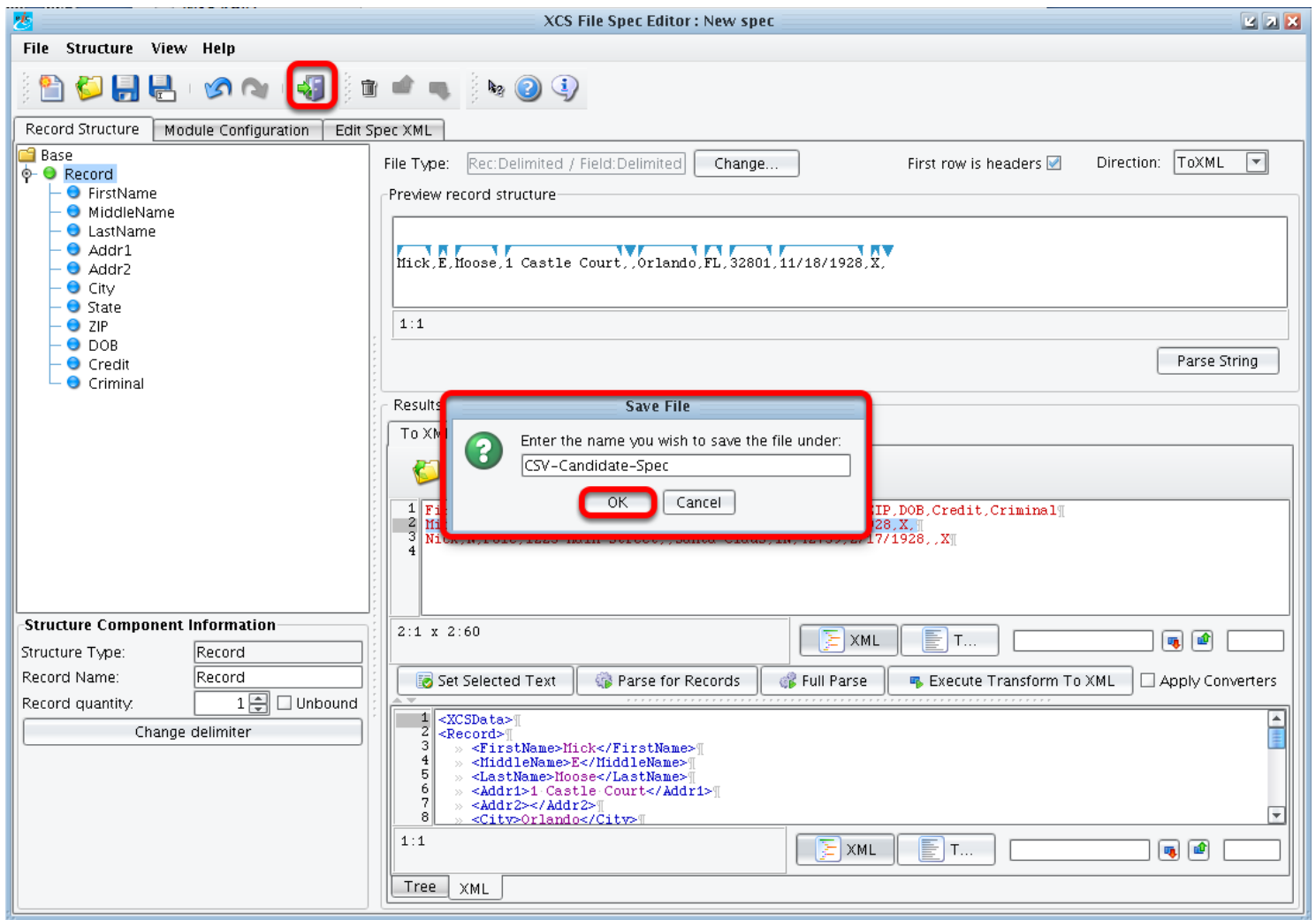


If you click on the **XML** tab, you'll see the XML structure that the File Specification will generate.



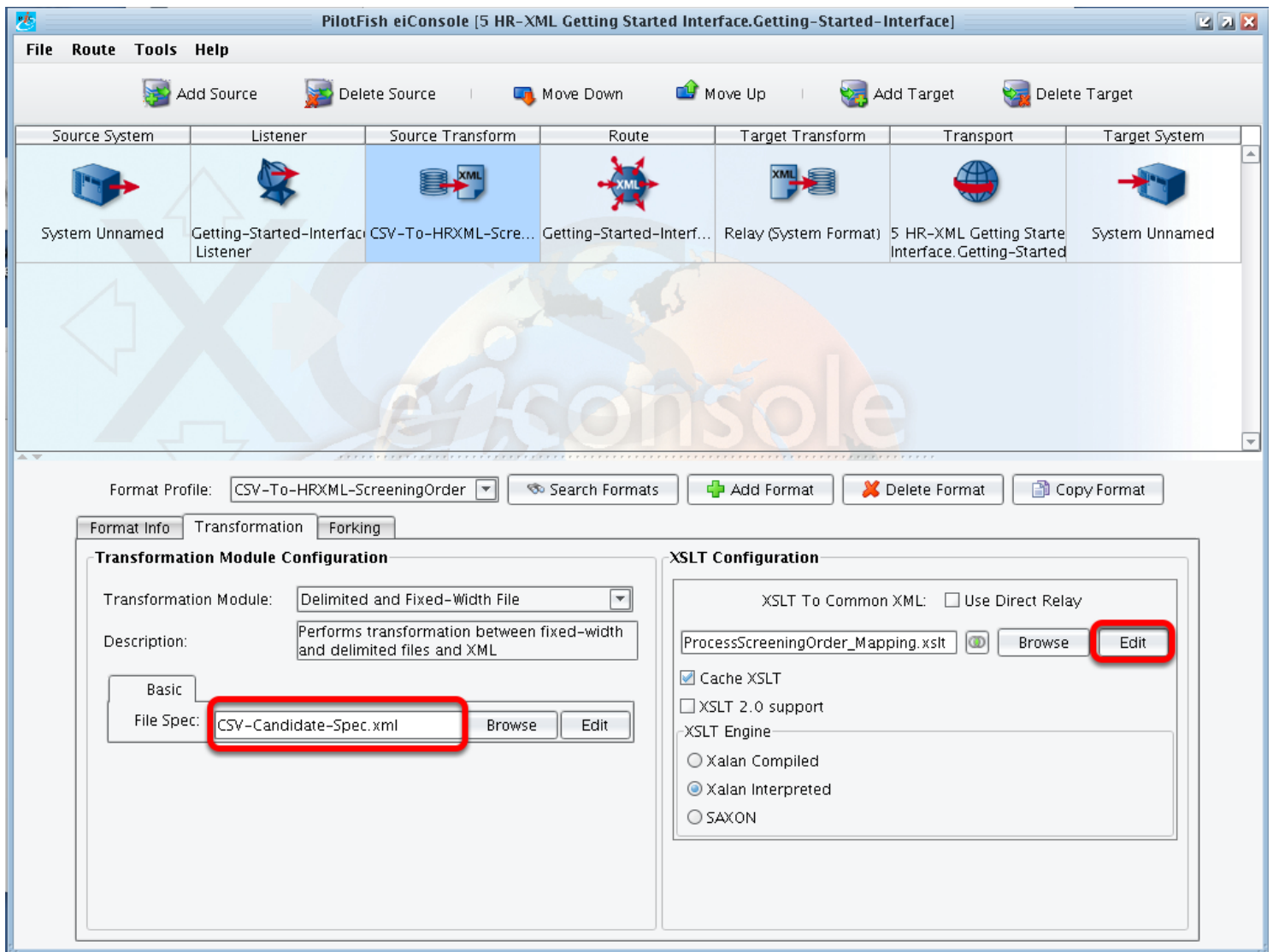
You can also validate how a particular line within the file is parsed. To do this, select a line of the file, select a corresponding Record on the left, and click the **Set Selected Text** button.

If all looks in order you can save your File Specification and move on. Click the **Save** icon in the upper left menu bar.



And give the File Specification a name. Let's call it **"CSV-Candidate-Spec"** and click **OK**.

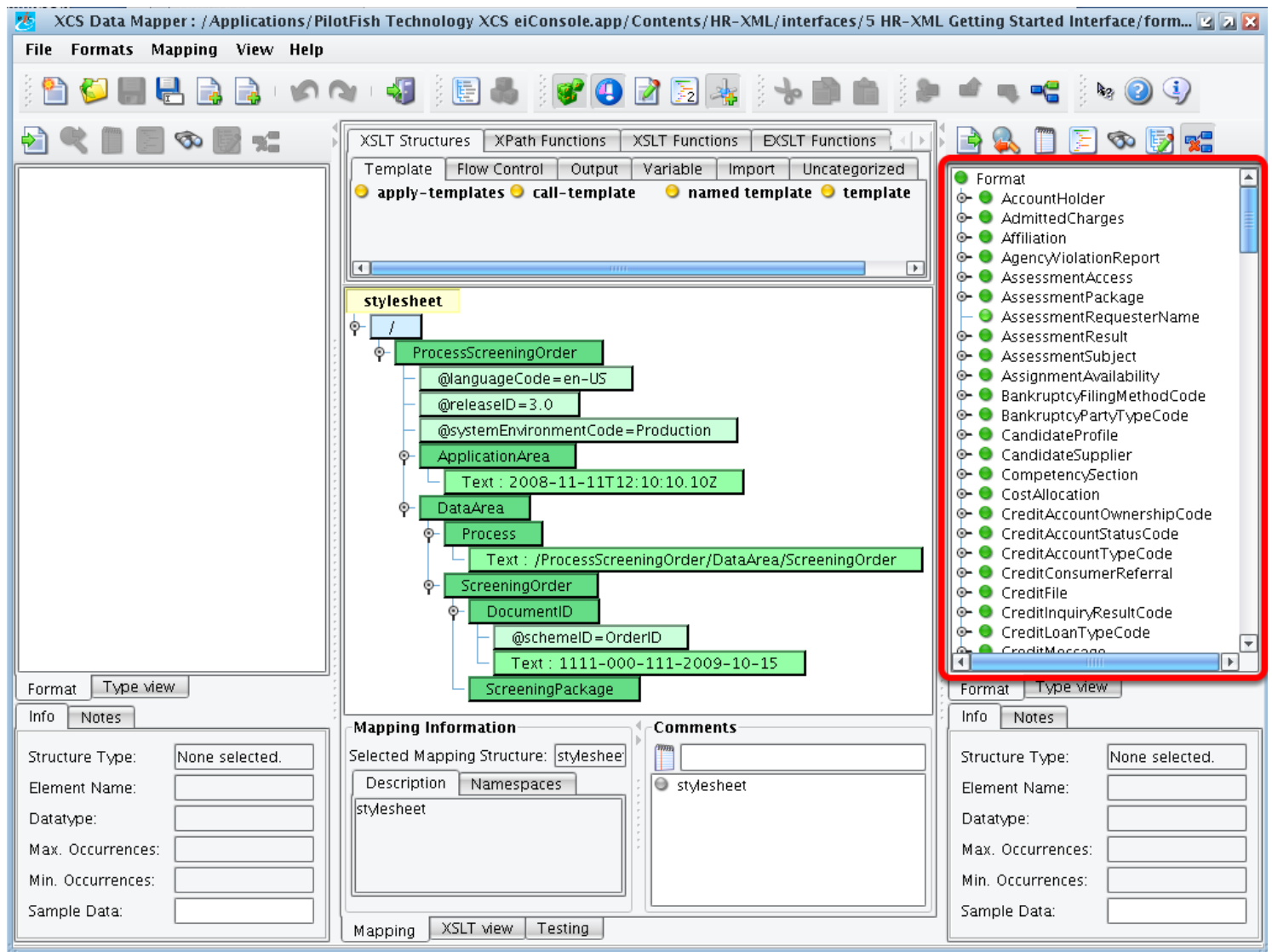
Then click the **Return to Console** button in the menu bar.



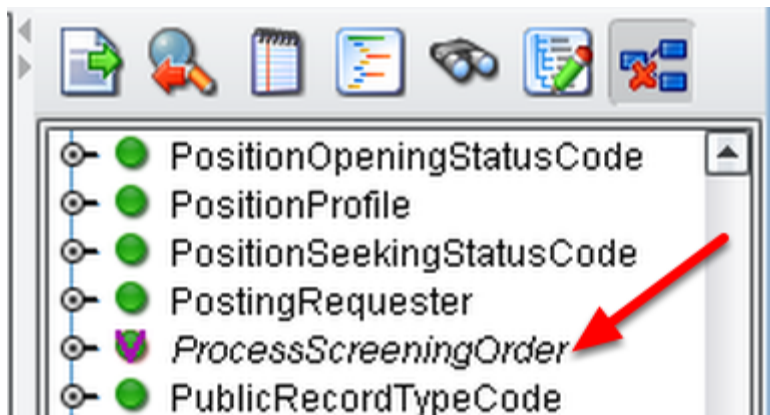
You've now returned to the main route grid where you see that the File Specification has been populated with XML file generated by the File Specification Editor.

Now that the CSV file can be converted into XML, your next step is to generate the XSLT that will map that XML onto the ProcessScreeningOrder HR-XML BOD. To do this, click the **Edit** button next to the XSLT name in the **XSLT Configuration** dialogue.

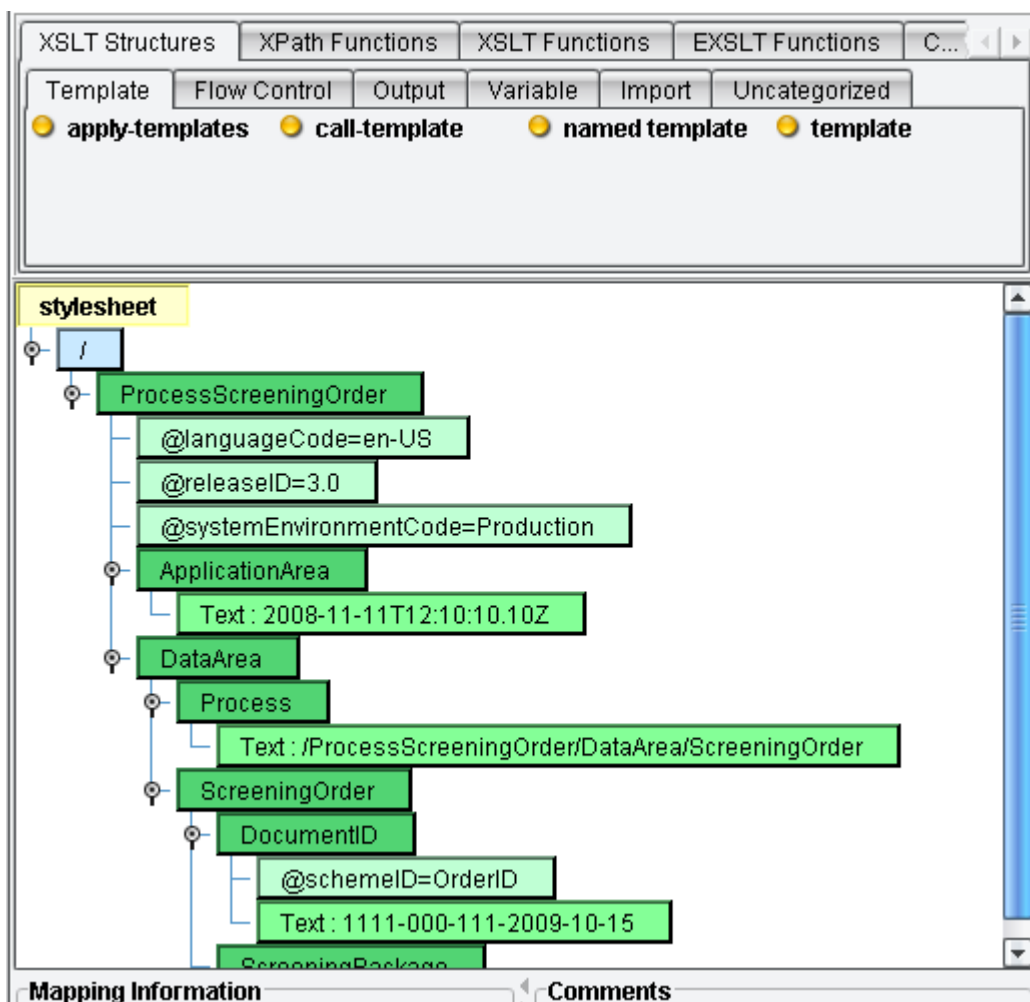




This launches the XCS Data Mapper. The XCS Data Mapper is the tool used within the eiConsole to create logical mappings between 2 different data formats. You'll see here that when you load a template, the panel on the left hand side is blank. This will need to be populated with the format of the data that you wish to map from. The tree on the right hand side has been populated with a tree representing the structure of the HR-XML model.



You'll note that the *ProcessScreeningOrder* node has been italicized and has a checkmark. This indicates that this is the particular element, or one of the elements, that has been used in the mapping.

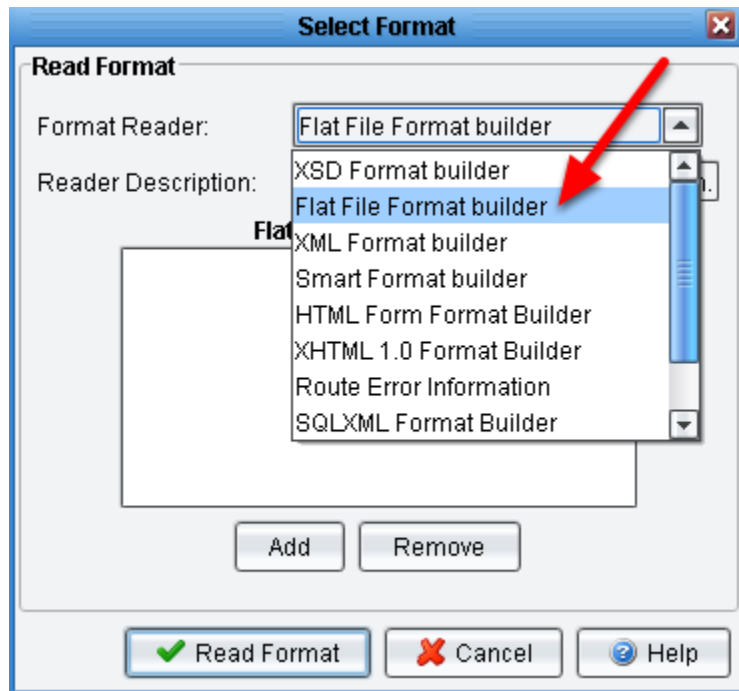


The middle panel contains a set of green nodes, and will contain a set of blue nodes as well. The green nodes indicate XML elements that will be in your output. Here you see the bare bones requirements for the *ProcessScreeningOrder* message. There are currently no blue nodes because no values from the Source

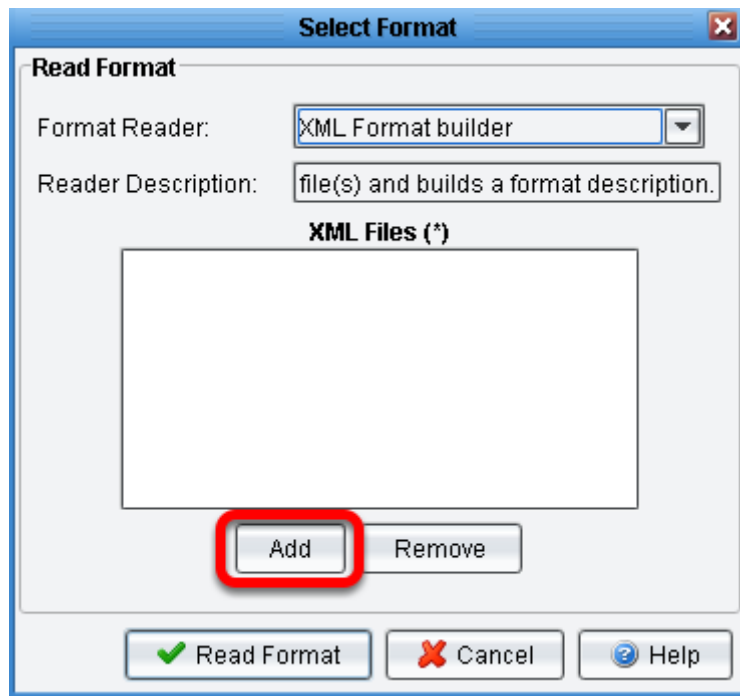
side have been mapped in yet. Your job will be to load the format of the Source and then map those onto the correct ProcessScreeningOrder template.



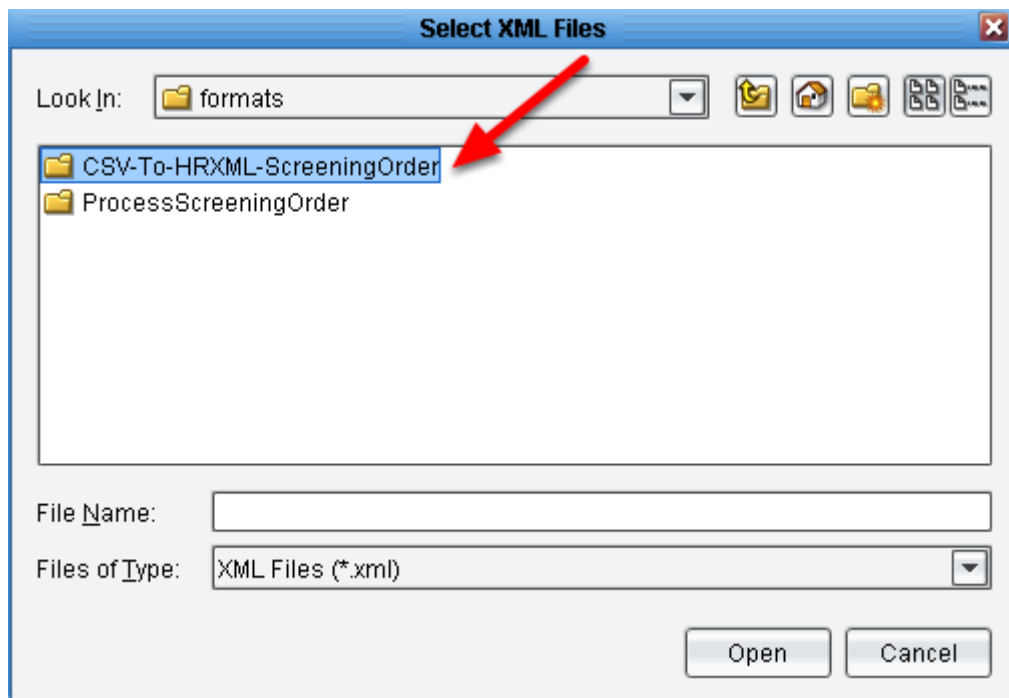
To begin this process, load the File Specification as your Source format. To do this, click the **Open Source Format** button in the top menu bar.



When the Select Format dialogue appears, choose the **Flat File Format Builder**.

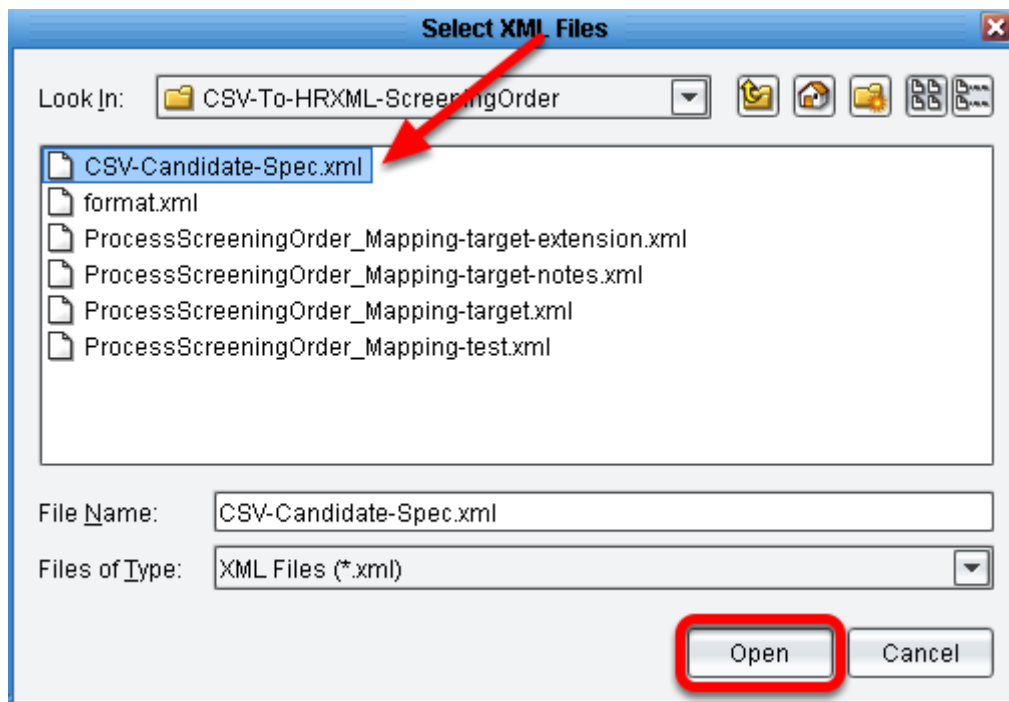


Once selected from the Format Reader dropdown you'll need to select the specific File Specification file. To do that, click the **Add** button underneath the Flat file format files area.

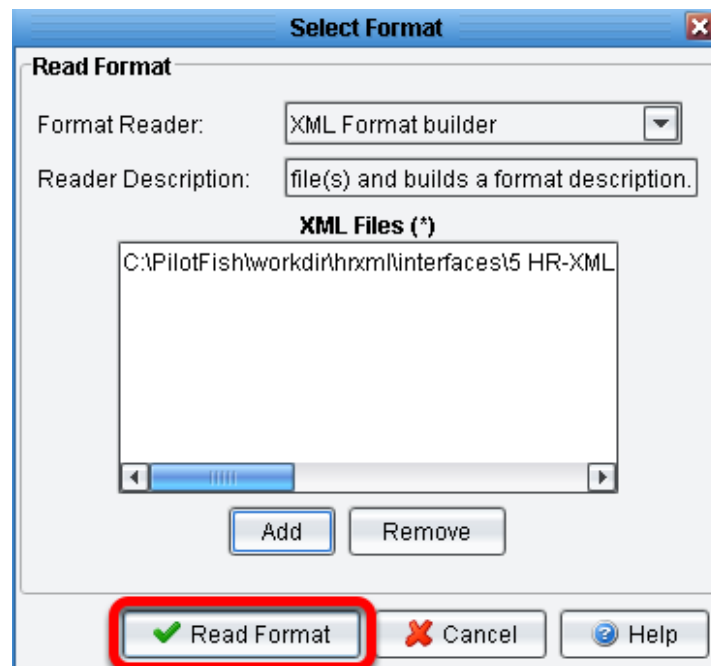


You'll now need to navigate to the file specification you created in the previous step. To do this, locate the formats folder underneath your distribution. A new format folder will have been created to match the name that

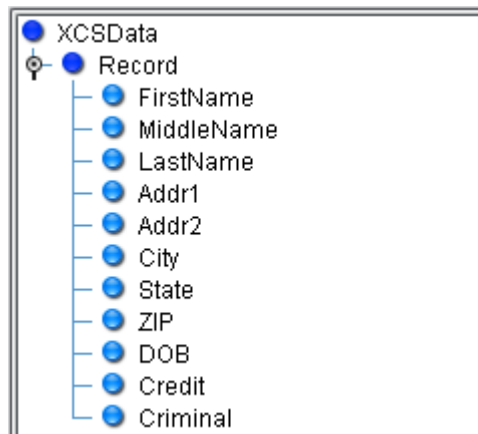
you provided when you created a copy of the ProcessScreeningOrder template. Find that folder called **CSV-To-HRXML-ScreeningOrder** and double click it.



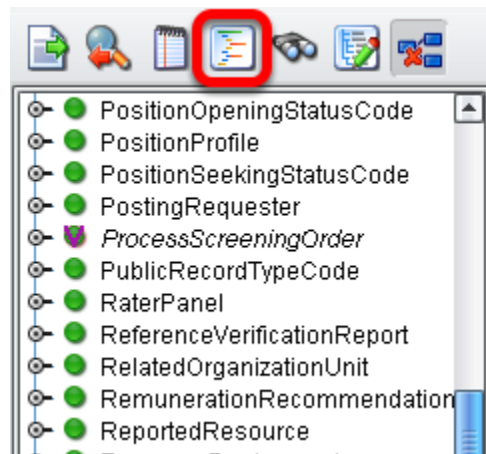
Inside this folder you should see the File Specification that you created, the **CSV-Candidate-Spec**. Select this file and click Open.



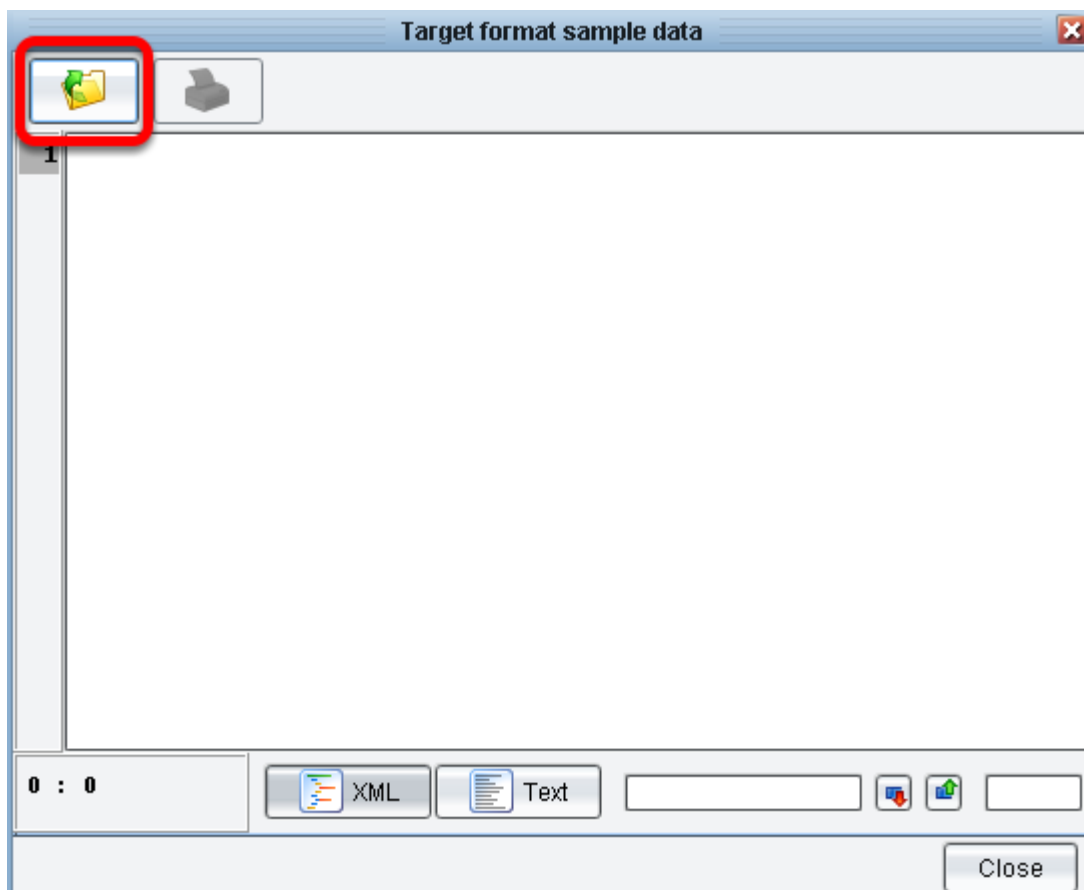
Then click **Read Format**.



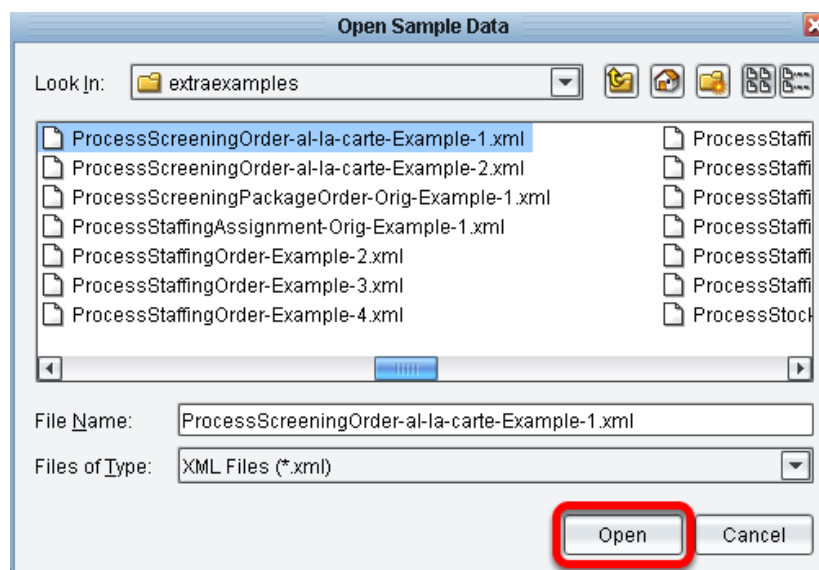
The left hand panel will be populated with a tree very similar to the one you just saw in the File Specification Editor. The Root node is XCSData. Each Record is represented with a Record node. Each individual field is a leaf node.



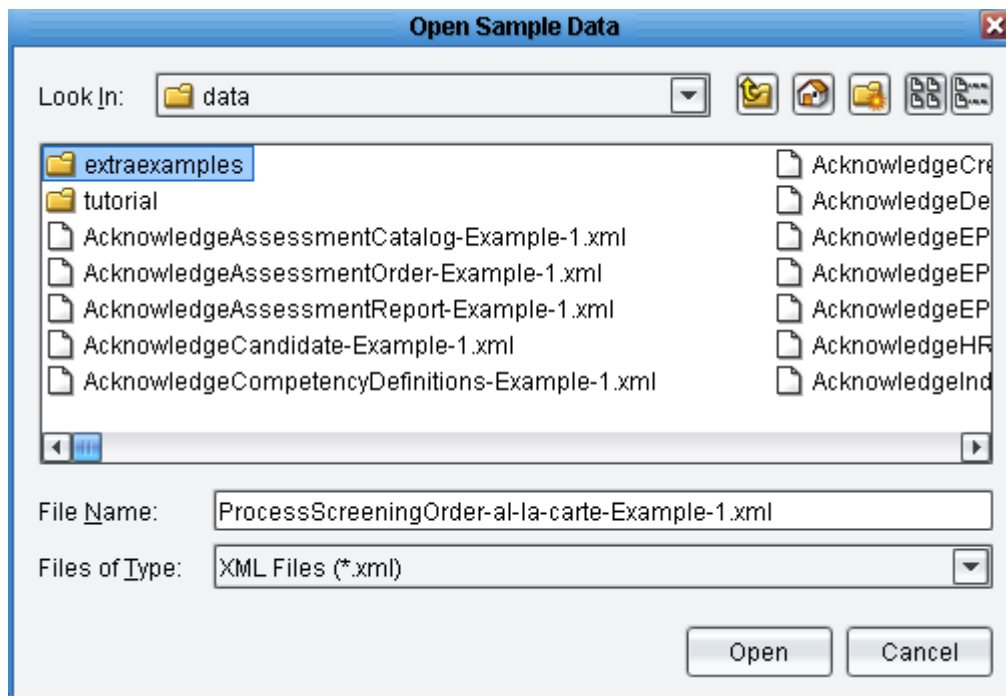
The next thing you'll want to do is load in a sample of the ProcessScreeningOrder template that you want to create. Instead of using the lightweight template that's there, you'll want to build this from an HR-XML provided sample file. To do this, click the **Use Sample Data** Icon above the Target Format tree.



This will launch the Target format sample data dialogue. Click the **folder** icon in the upper left, which will allow you to load sample data from a file.

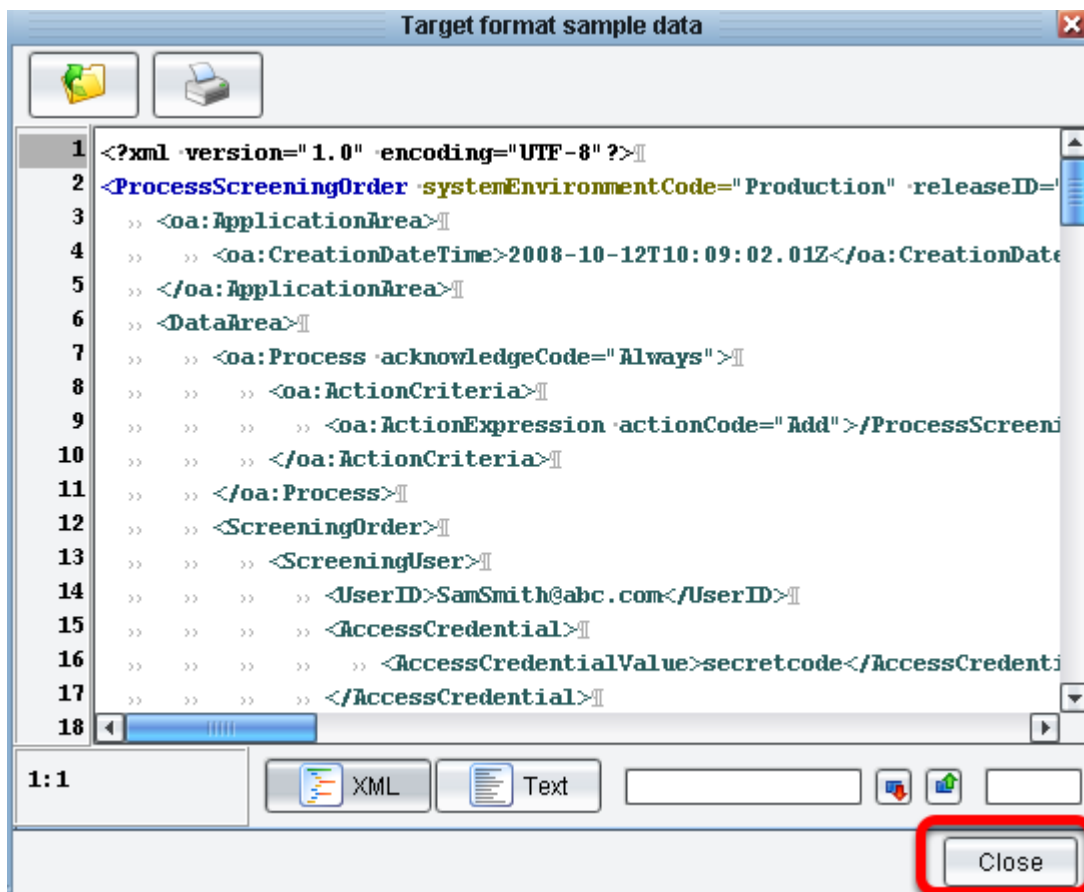


For this tutorial you'll use the **ProcessScreeningOrder-al-la-carte-Example-1.xml**. Select this file and click **Open**.

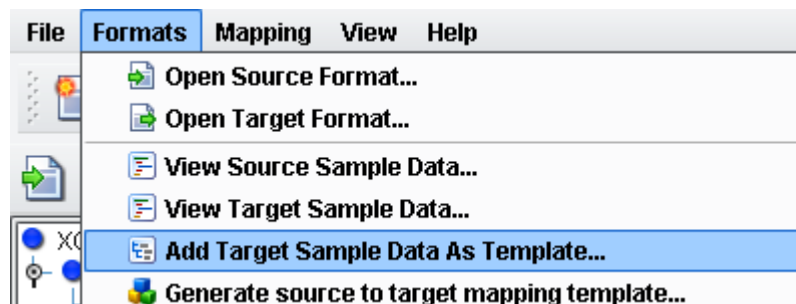


Navigate to the **extraexamples** folder of your distribution and double click it. It can be at "\\interfaces\\3 HR-XML Templates\\data\\extraexamples\\".

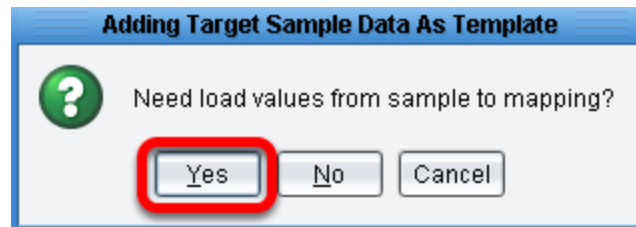




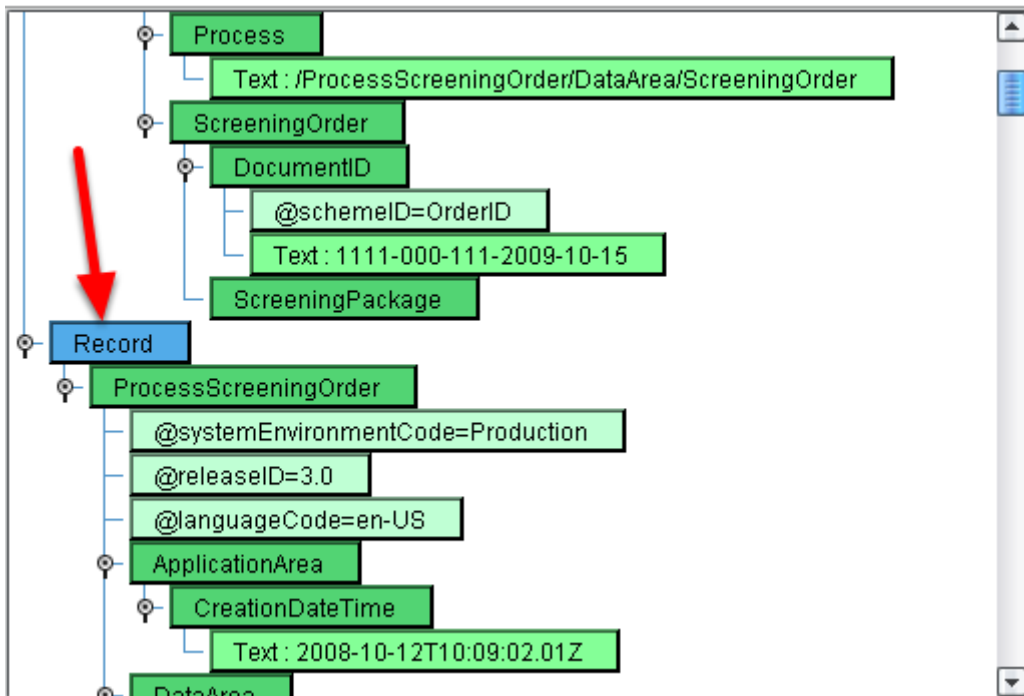
The contents of the file will appear in the Target format sample data area. Next click the **Close** button.



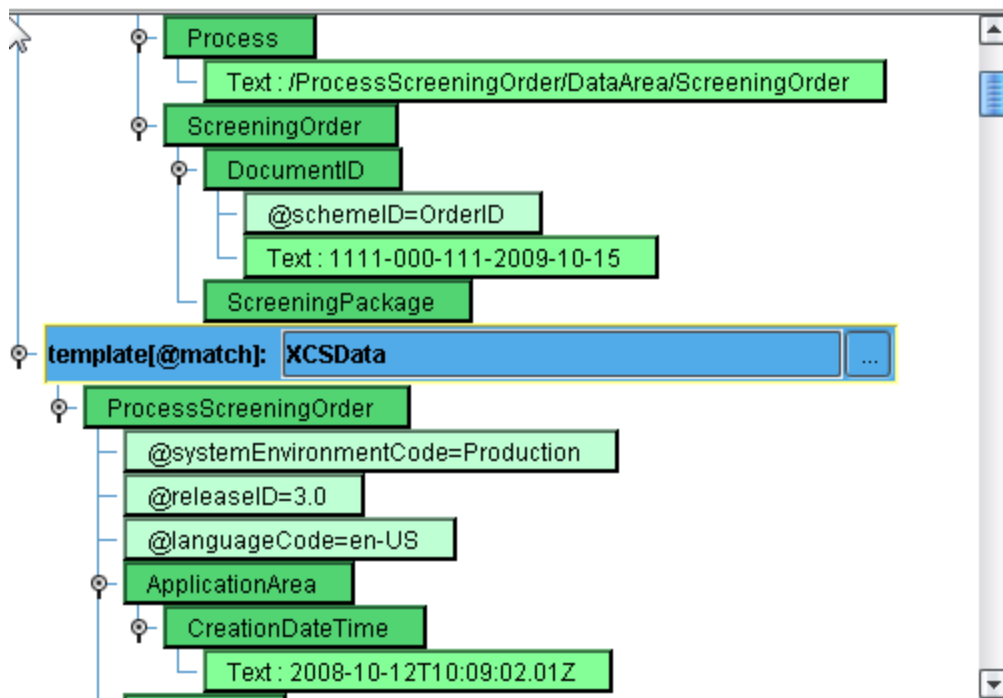
You'll use this template to provide information about the fields that you'll map to. To do this, in the Formats menu select **Add Target Sample Data As Template**.



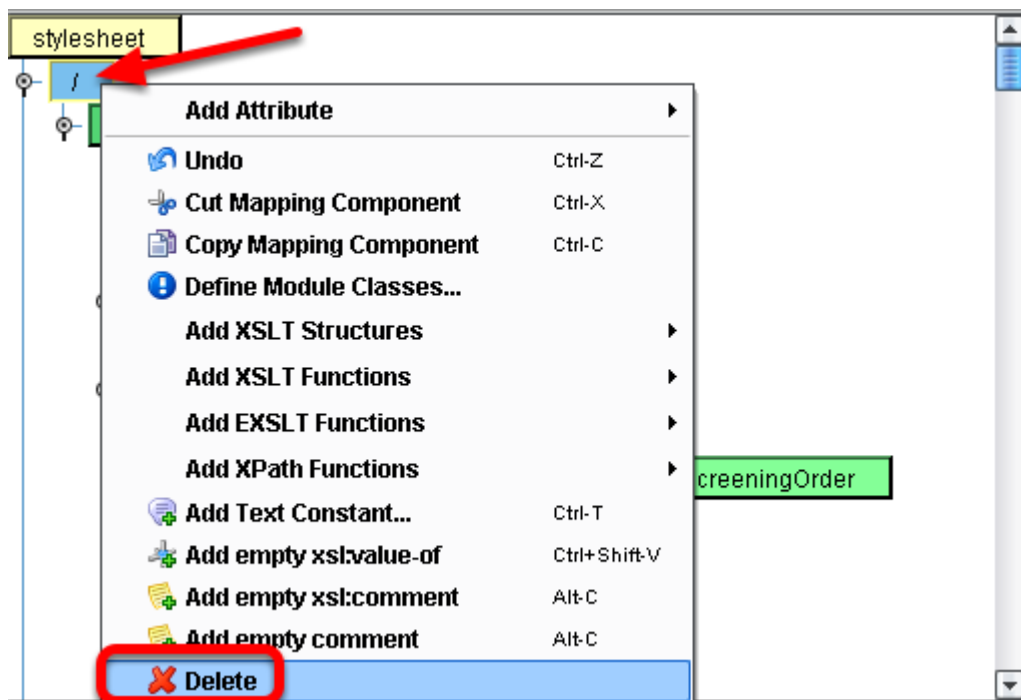
When prompted to load values from the sample to the mapping click **Yes**.



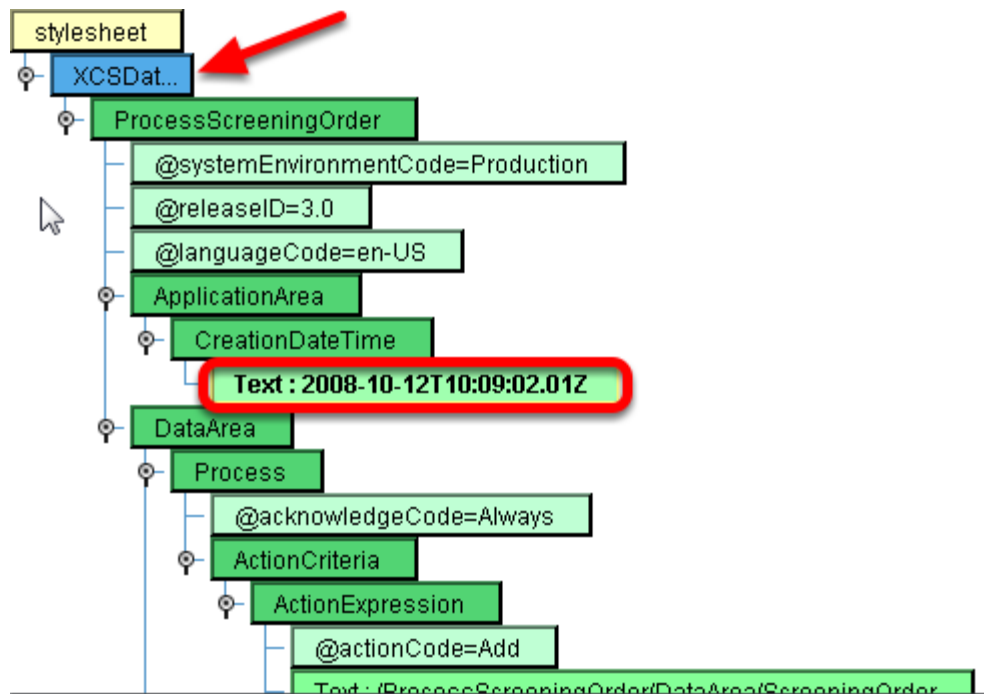
A series of new green nodes will appear in the center panel. Scrolling to the top of that, you'll see a blue node indicating Record.



Now change Record to **XCSDData**. Input it and hit **Enter**. We've got the template you'll wish to map to.

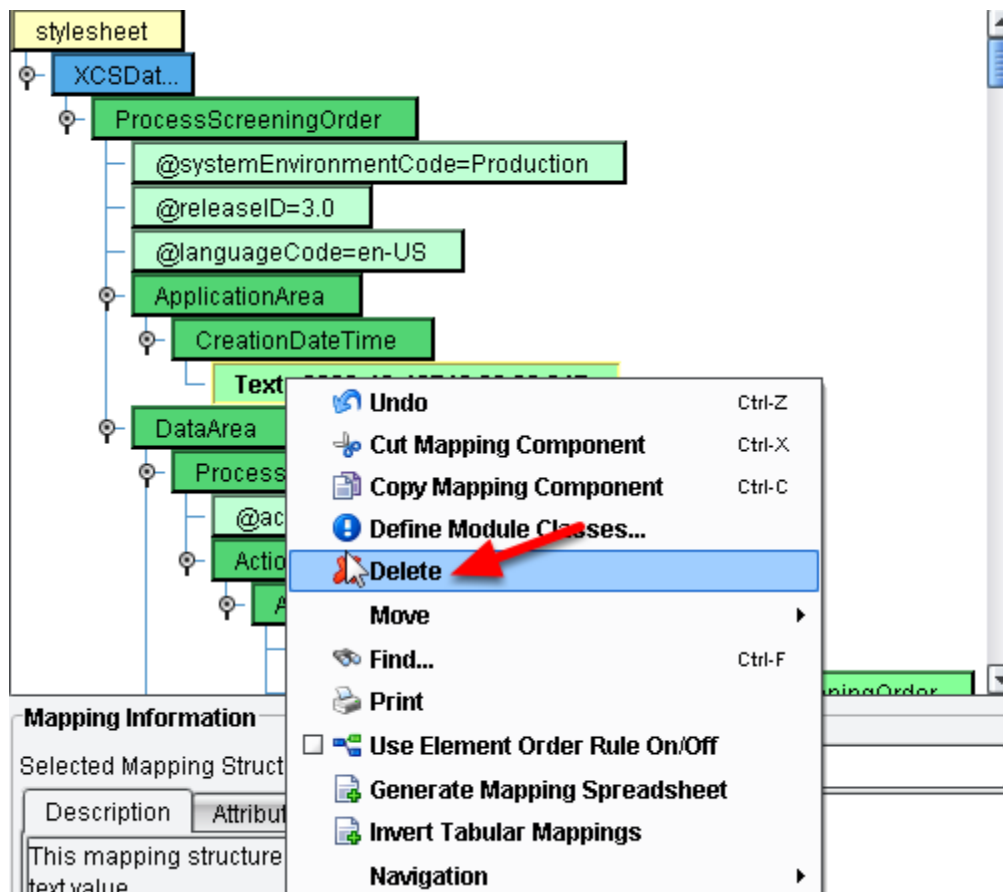


In order to delete the previous template, select / (the blue node) at the very top of the center panel. Right click and choose **Delete**.

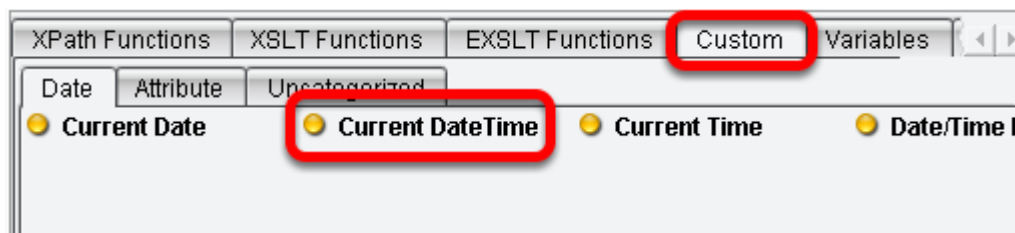


**XCSData** will now appear as the only blue node in the mapping. Your next step is to work your way through the center panel, mapping fields from the left into the center as appropriate.

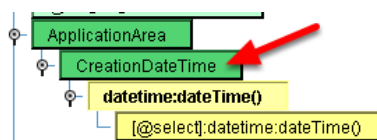
Let's begin by populating the Creation Date and Time. Select the **CreationDateTime** green node underneath ProcessScreeningOrder in the ApplicationArea. You'll see that it's currently populated with some hard coded text that was pulled from the sample file. You'll want to replace this with the current date and time.



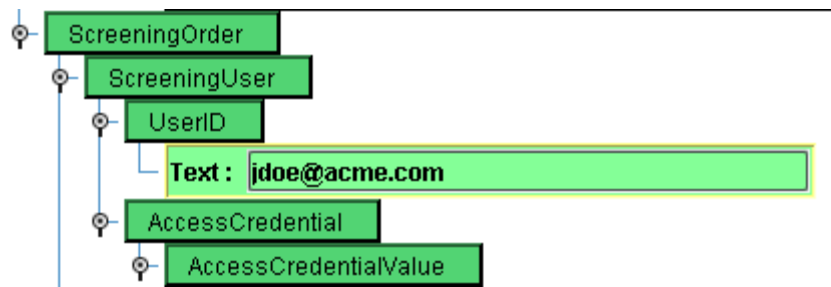
To do this, left click on the text node. Then right click and choose **Delete**.



Now, from the pallet above the main mapping area, select the **Custom** tab and the **Date** sub-tab. Click on the **Current DateTime** node and drag that on top of the **CreationDateTime** node in your main grid.

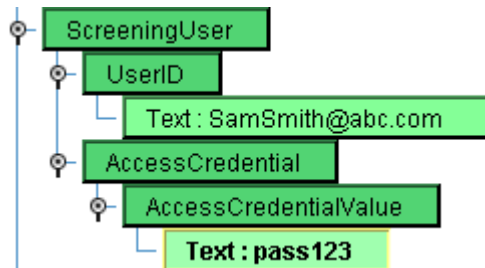


When the CreationDateTime node is highlighted with a yellow border, release the mouse and a new yellow node will be created which will populate the CreationDateTime element with the current Date and Time.

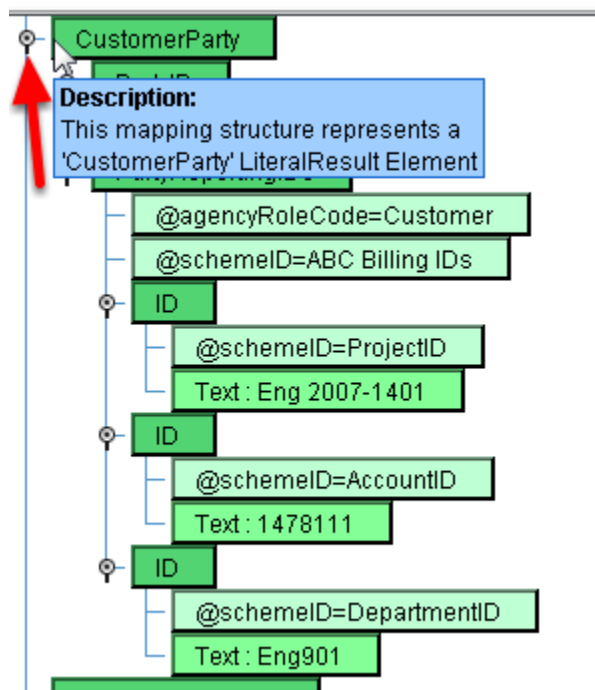


Scrolling further through the mapping you'll leave the DataArea – Process alone and move into the ScreeningOrder. You'll see there's a ScreeningUser section with a UserID and an AccessCredentialValue hard coded. Let's assume you want to keep this hard coded, but change it to populate it with your own value. To do this, double click first on the Text node underneath UserID. The text will become editable. Enter in "jdoe@acme.com". Enter the text and then hit the **Enter** key.

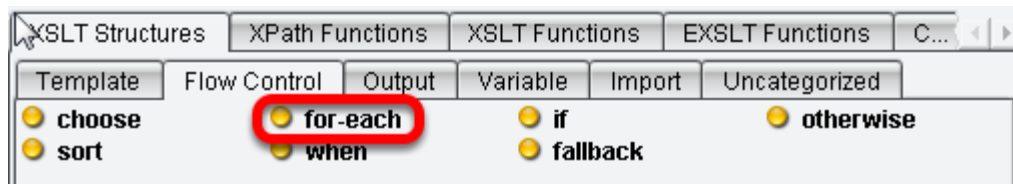
**Note:** it's important when using the Console to hit Enter after editing the nodes value. Hitting Esc or simply clicking off of the node will reset the node to its original value.



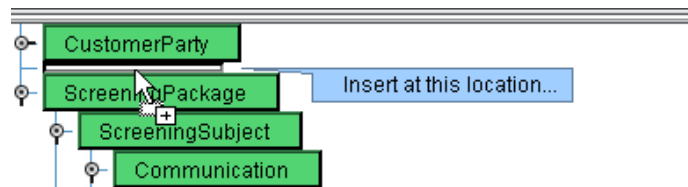
Repeat this process with the AccessCredentialValue. Changing secretcode to the simple text "**pass123**" and click **Enter**.



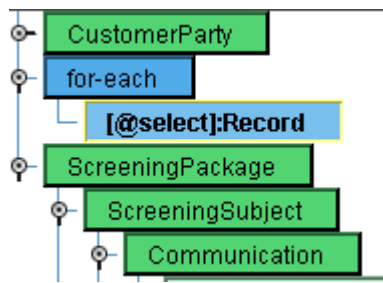
For the purposes of this demonstration, leave the CustomerParty section alone. To collapse this node click the collapse icon next to the CustomerParty. This will roll up all of the children underneath CustomerParty to hide them.



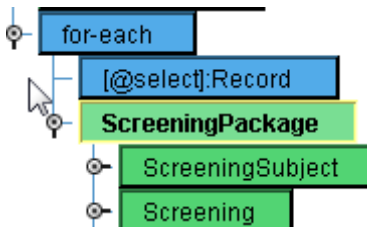
Next, move into the ScreeningPackage area. The ScreeningPackage has two child nodes, ScreeningSubject and then a set of Screening nodes. Because you may have multiple records in a file representing multiple candidates, let's assume there could be multiple ScreeningPackages within this message. You'll want to create one of them for each record that you encounter. To do this, underneath the XSLT Structures tab, select the Flow Control sub-tab. Then click the **for-each** node...



And drag it carefully above the ScreeningPackage. You'll want the gray line to appear between CustomerParty and ScreeningPackage. ScreeningPackage should not be highlighted.

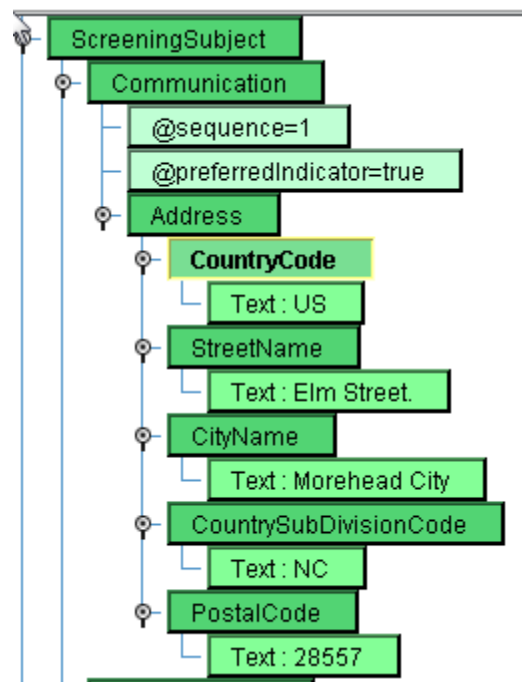


When you release the mouse a blue for-each node should appear. You'll want to create a ScreeningPackage for each record that you encounter. To choose what you want to iterate over, choose the Record node underneath XCSDData from the Source tree and drag it on top the for-each. When the for-each is highlighted with the yellow border release the mouse. You'll now see that anything you create underneath the for-each will occur each time you encounter a Record in the Source.

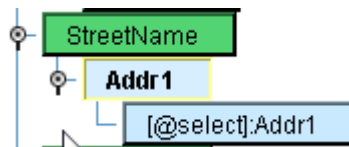


You'd like to create a ScreeningPackage in this scenario. To do this you'll make ScreeningPackage a child of the for-each node. To accomplish this, click on the ScreeningPackage node and drag in on top of the for-each. When you release the mouse, ScreeningPackage will now be a child of the Record.

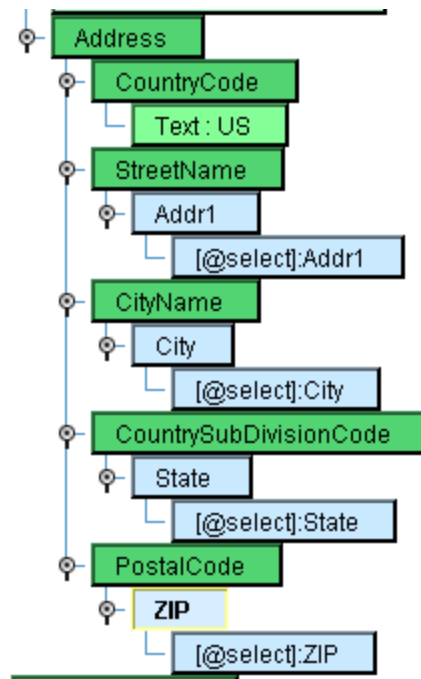




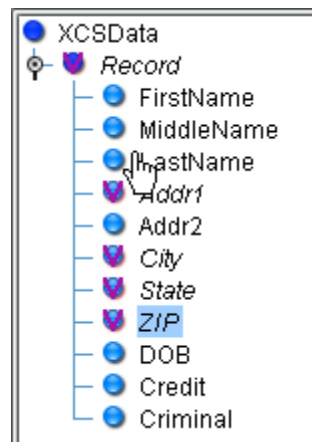
Now, you'll work on the ScreeningSubject. Let's map the address area. You'll see that you have a set of hard coded values for the CountryCode, StreetName, CityName, CountrySubDivisionCode, and PostalCode.



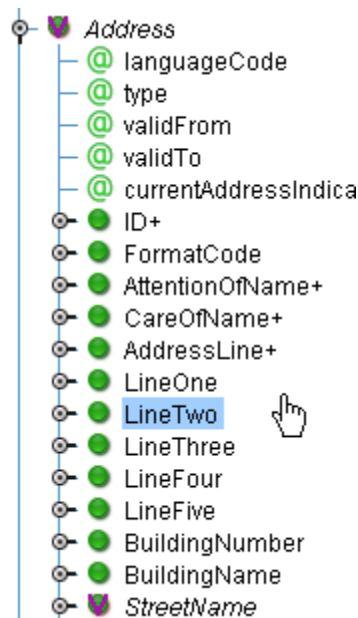
Examining your Source, you'll see for address information you have fields called Addr1, Addr2, City, State, and Zip. Assume this is a US based candidate, so keep the default CountryCode as US. However, delete the StreetName text Elm Street and replace that with Addr1 from the sample file. Click on the Text node underneath StreetName and right click for the Delete key. Then drag Addr1 under the Record node on top of StreetName. You've now created a one-to-one mapping between the Addr1 node from the Source and the StreetName element in your Result.



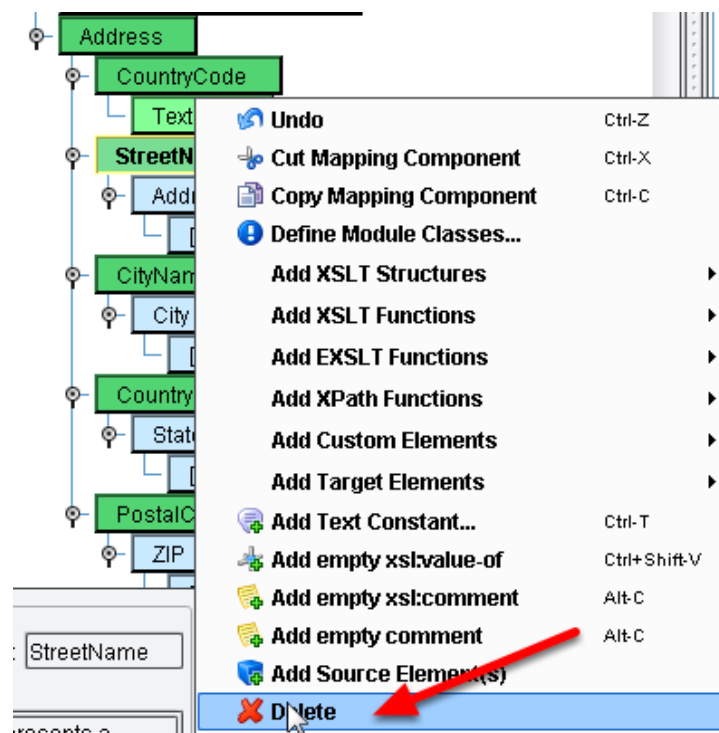
Repeat this process with CityName and CountrySubDivisionCode. Lastly, do the same thing with PostalCode.



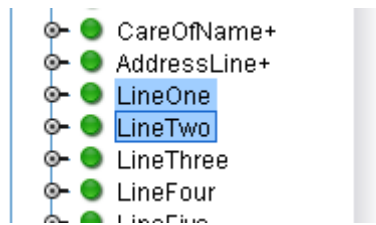
You'll note that Addr2 doesn't have a check mark. It hasn't yet been mapped. So you'll want to find an additional element legally in the HR-XML specification that you can add to the address.



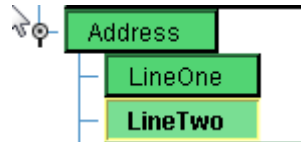
To do this, navigate to the Address area in the tree (Format/ ProcessScreeningOrder/ DataArea/ ScreeningOrder+/ ScreeningPackage+/ ScreeningSubject+/ Communication+/ Address/) on the right hand side. Once you've done that you'll see a list of all of the allowable children with those that have been used checked off. You'll note that StreetName has been mapped; however, LineOne and LineTwo would be appropriate for mapping Addr1 and Addr2.



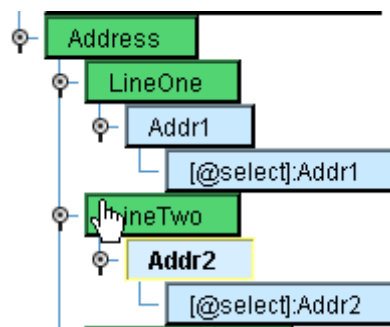
Let's delete the StreetName mapping. Click on the StreetName node. Right click and choose **Delete**.



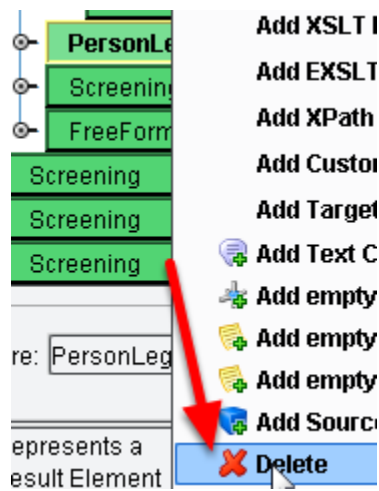
Let's replace that with LineOne and LineTwo. Multi-select LineOne and LineTwo by selecting each while holding the Control key.



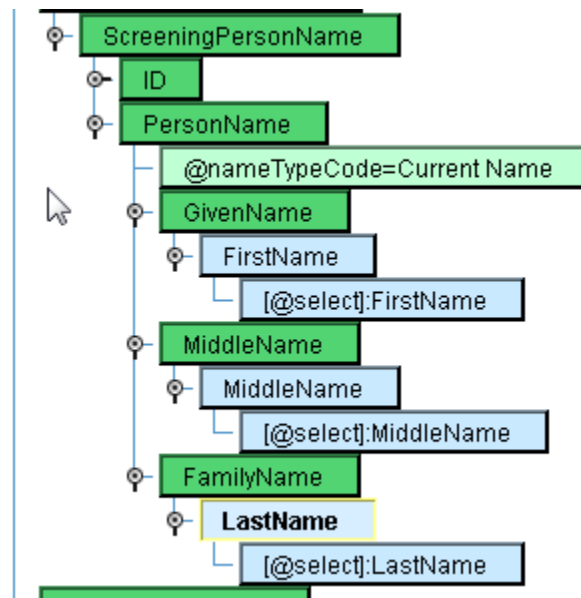
Drag these nodes on top of their parent the Address node. You'll see that LineOne and LineTwo green nodes now appear in the correct element order underneath Address.



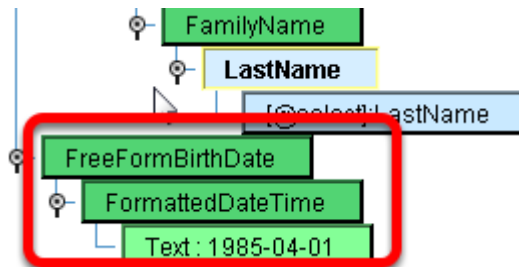
Now, map Addr1 and Addr2 to those nodes respectively.



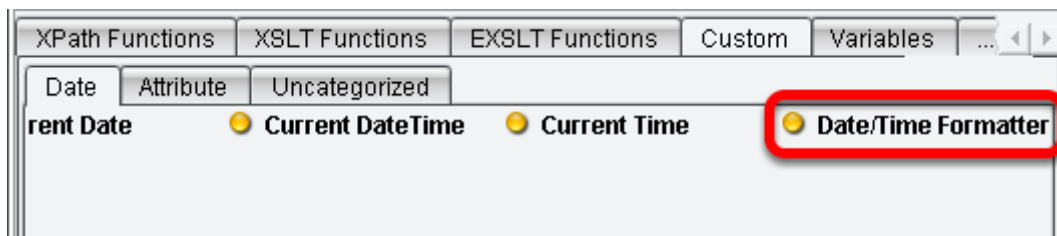
Having done that let's move on. You don't have social security number information, so you're going to remove the PersonLegalID. Select that node, right click and hit **Delete**.



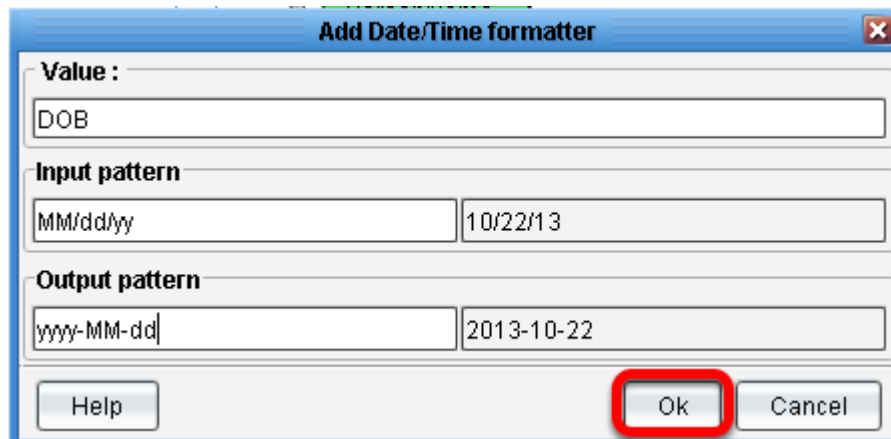
In ScreeningSubjectName, you'll only include the first ScreeningPersonName. Delete all other child nodes. Using the same process as above you'll map FirstName to GivenName, MiddleName to MiddleName, and LastName to FamilyName. Delete each of the existing text nodes and then replace them by dragging and dropping the corresponding fields from the Source.



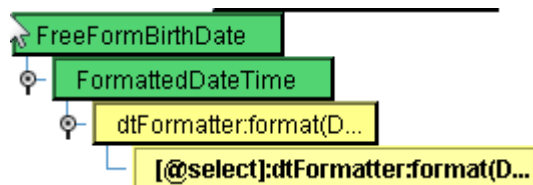
Save your work. Click the **Save** disk icon in the upper left. This will save your XSLT and allow you to continue. Now work on the birth date. Under the FreeFormBirthDate node you'll find the FormattedDateTime node containing a hard coded birth date. Delete this as you've done several times before and drag the DOB element onto the FormattedDateTime node.



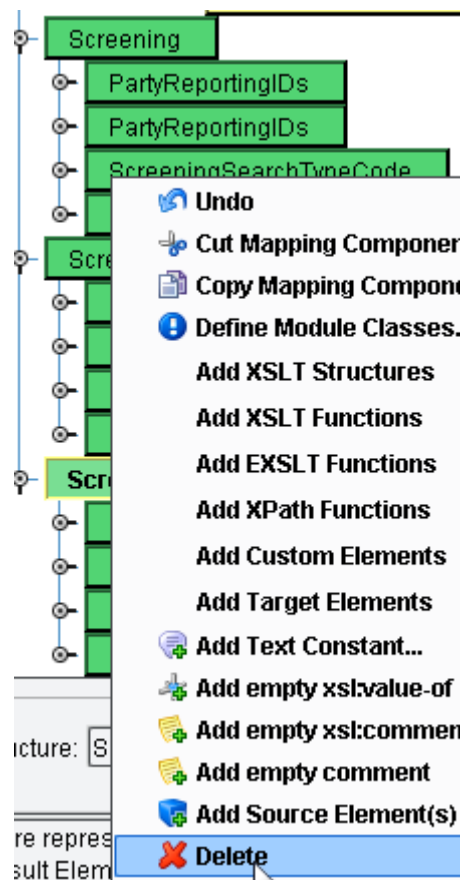
However, thinking back to your sample, the CSV had data in MM/dd/yy format. Here, you'll want the format as yyyy-MM-dd. To handle such formatting you can use the Date/Time Formatter. Under the Custom tab in the pallet and the Date sub-tab, find the **Date/Time Formatter** tool and drag it onto the DOB node.



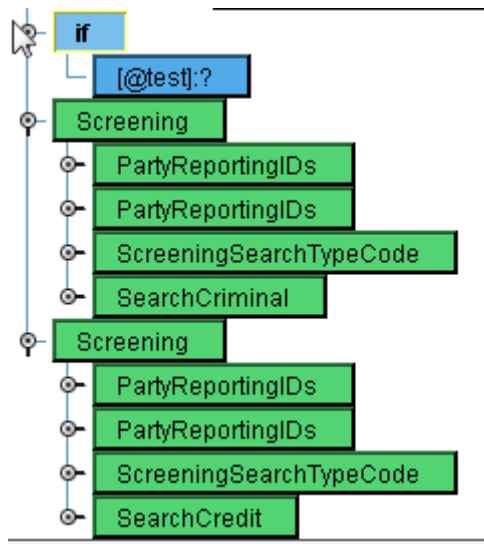
When the mouse is released the Add Date/Time Formatter dialogue will appear. Enter your desired Input pattern (MM/dd/yy) and replace that with your desired output pattern (yyyy-MM-dd). Samples of how this should appear are displayed to the right using the current day. Click **OK**.



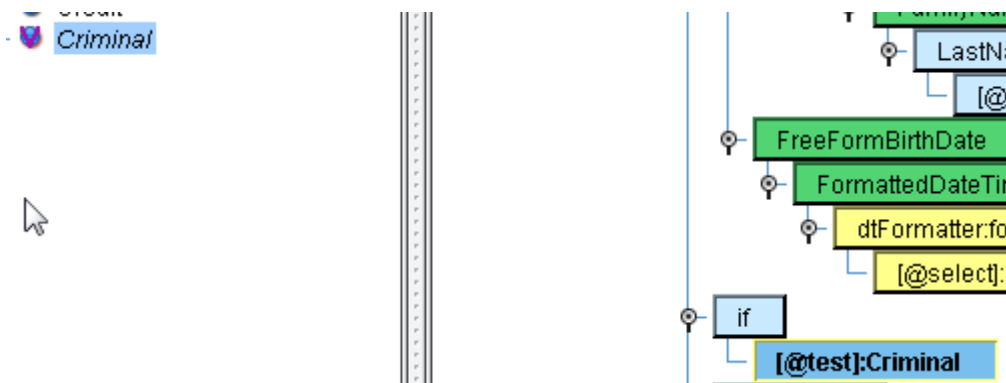
The blue node will now become yellow with all of the logic having been input into the mapping.



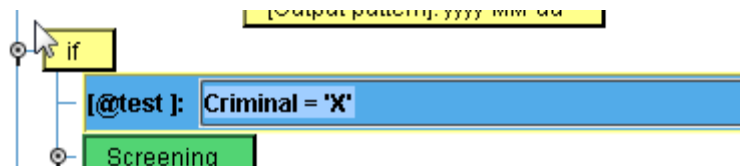
You now have the Credit and Criminal fields to work with. Remembering back to your sample file you'll see that the Credit and Criminal fields were either blank or contained an X. An X indicates this is a check that you'd like to perform. This is an example of where you'd want to create a conditional mapping based on the content of the fields. Examining the first two screening nodes you'll see that one represents a criminal check, the second represents a credit check. The third one you can delete. Do that now.



If the credit field has an X you'll want to create a structure very similar to the Screening representing the credit check. Similarly if criminal has an X you'll want to create the other screening structure. To implement this conditional logic click the XSLT Structures tab and choose the Flow Control sub-tab. Select the "if" tool... And drag it directly above the Screening.



The first check that you'll do is for the criminal X. Select the Criminal node and drag it onto the test.

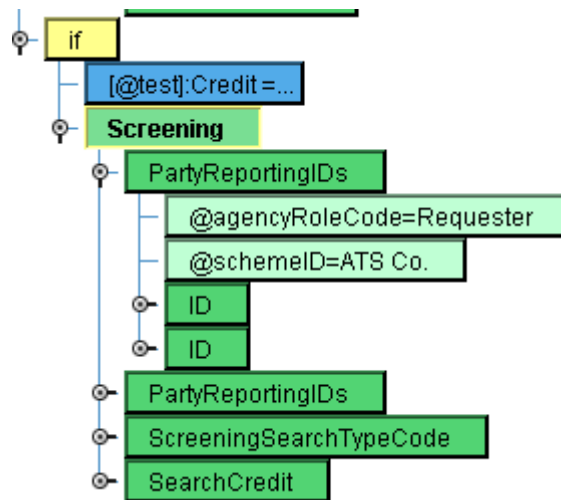


Then, double click to input the condition. You want to say if the Criminal equals X you'll wish to create this screening code Criminal = 'X'. Enter that condition into the text box and click **Return**.





Now, as you did with the for each, you'll want to make this Screening a child of that condition. Select the Screening node and drop it on the if.



Let's repeat this process for the credit check. Use Credit = 'X'. While in a real life implementation you'd likely want to modify some additional details about this screening, this is sufficient for the tutorial.

```

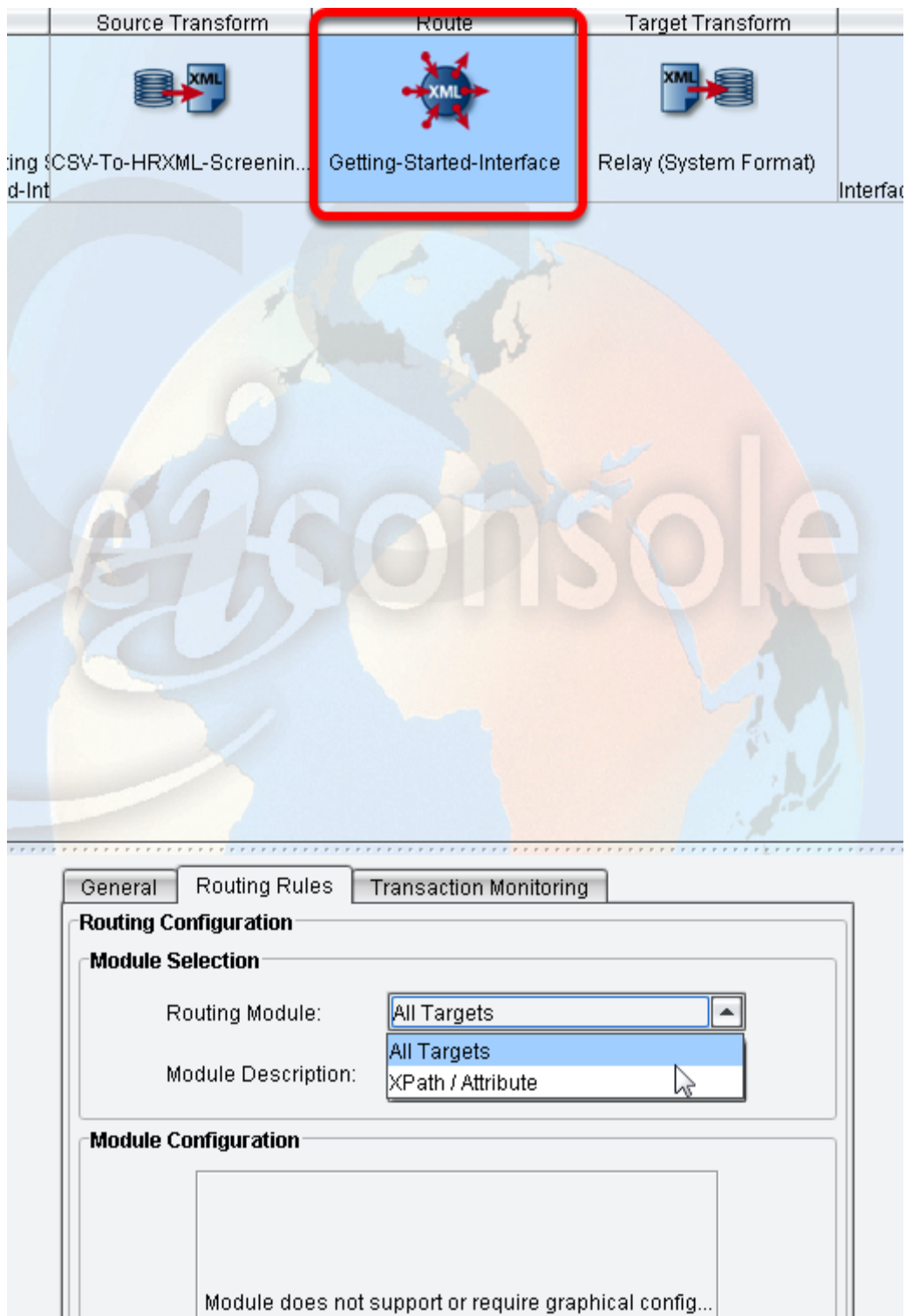
97 >> >> >> >> >> >> >> <od:CountrySubdivisionCode listID="Country">AC</od:CountrySubdivisionCode>
98 >> >> >> >> >> >> >> </hr:SearchCriminal>
99 >> >> >> >> >> >> >> </hr:Screening>
100 >> >> >> >> >> >> >> </xsl:if>
101 >> >> >> >> >> >> >> <hr:Screening>
102 >> >> >> >> >> >> >> <hr:PartyReportingIDs agencyRoleCode="Requester" schemeID="ATS-Co.">
103 >> >> >> >> >> >> >> <hr:ID schemeID="CandidateID">1111478111</hr:ID>
104 >> >> >> >> >> >> >> <hr:ID schemeID="RequisitionID">998Eng901</hr:ID>
105 >> >> >> >> >> >> >> </hr:PartyReportingIDs>
106 >> >> >> >> >> >> >> <hr:PartyReportingIDs agencyRoleCode="Customer" schemeID="ABC-Billing-ID">
107 >> >> >> >> >> >> >> <hr:ID schemeID="ProjectID">Eng-2007-1401</hr:ID>
108 >> >> >> >> >> >> >> <hr:ID schemeID="AccountID">1478111</hr:ID>
109 >> >> >> >> >> >> >> <hr:ID schemeID="DepartmentID">Eng901</hr:ID>
110 >> >> >> >> >> >> >> </hr:PartyReportingIDs>
111 >> >> >> >> >> >> >> <hr:ScreeningSearchTypeCode>Credit</hr:ScreeningSearchTypeCode>
112 >> >> >> >> >> >> >> <hr:SearchCredit>
113 >> >> >> >> >> >> >> <hr:CreditBureauCode>Experian</hr:CreditBureauCode>
114 >> >> >> >> >> >> >> <hr:SearchCreditTypeCode>Summary</hr:SearchCreditTypeCode>
115 >> >> >> >> >> >> >> <hr:EndUserName>Acme-Co.-Inc.</hr:EndUserName>
116 >> >> >> >> >> >> >> <hr:ScreeningPermissiblePurposeCode>Employment</hr:ScreeningPermissiblePurposeCode>
117 >> >> >> >> >> >> >> </hr:SearchCredit>
118 >> >> >> >> >> >> >> </hr:Screening>
119 >> >> >> >> >> >> >> </xsl:if>
120 >> >> >> >> >> >> >> </hr:ScreeningPackage>
121 >> >> >> >> >> >> >> </xsl:for-each>
122 >> >> >> >> >> >> >> </hr:ScreeningOrder>
123 </hr:DataArea>
124 </hr:ProcessScreeningOrder>
125 </template>
126 </sheet>
127

```

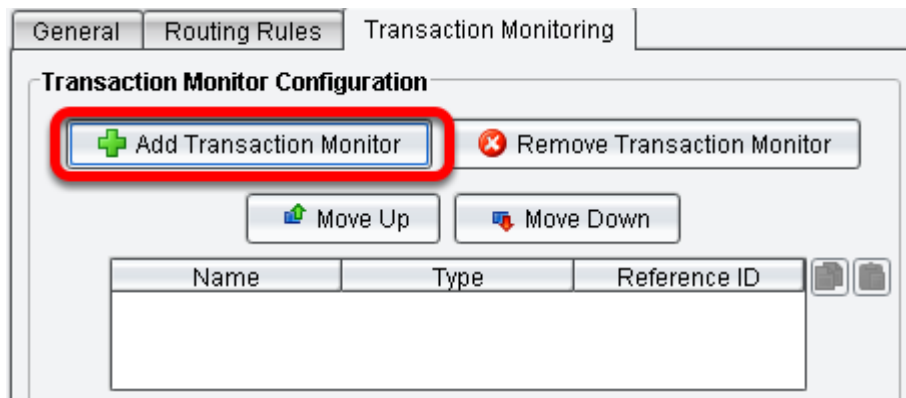
*Note: all of the work you've done that has been completed through drag & drop and wizards is represented under the covers as W3C standard XSLT. Clicking on the XSLT view tab underneath provides a text based view of the generated XSLT. This XSLT may be modified by hand at any time, and such changes will be directly reflected in the mapping.*

For now, let's move on.

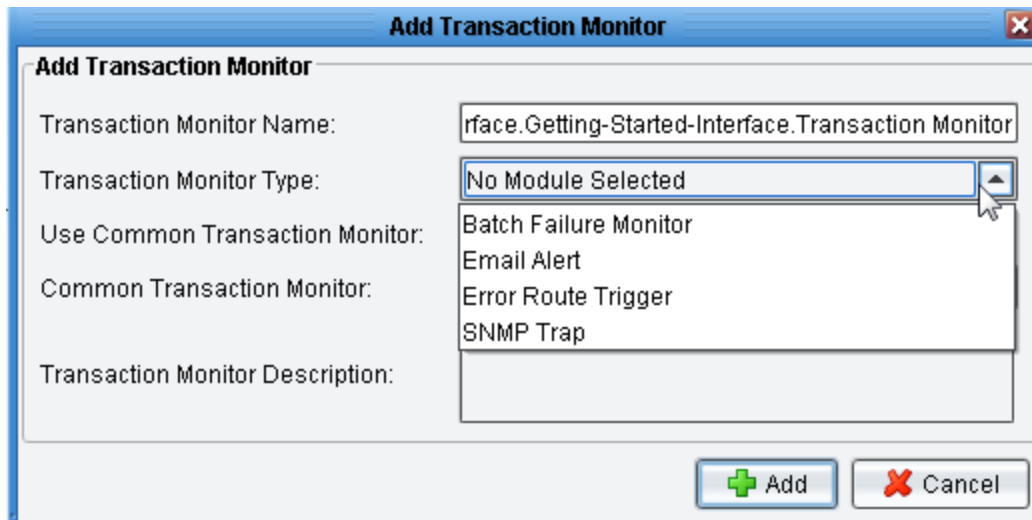
Click the **Save** icon in the top menu and then the **Return to Console** icon.



Next, click the **Route** icon in the main route grid. You won't need to configure any Routing Rules in this interface, but it's important to know that the Routing Rules tab can be used to set up XPath based routing rules to direct the data to one or multiple target systems based on its content.



Also configured at this point in the main route grid are Transaction Monitors. Transaction Monitors are the eiConsole's tool for exception handling.



Email, other interfaces, or SNMP traps may be set up so that someone can be proactively notified if something goes wrong in the execution of an interface.

Format Profile: Relay (System Format) Search Formats + Add Format X Delete Format Copy Format

**Format Info**

**System Name**

System Unnamed

**Format Metadata**

Tag Name	Tag Value
----------	-----------

+ Add X Remove

Now click on the **Target** Transform icon. When the format info panel opens, you'll note that the Target Transform is set to Relay. The XSLT Configuration has Use Direct Relay checked and No Transformation Module has been configured.

Processor Configuration **Transport Configuration** Post-Processors

Transport Configuration **Retry Configuration**

**Transport Configuration**

Transport Name: erface.Getting-Started-Interface.Directory Transport File Icon Folder Icon

Transport Type: **Directory** Up Arrow Down Arrow

Use Common Transport: **Directory**

Transport Description: Email (SMTP)  
FTP  
Generic Socket Transport  
HTTP Post  
JMS  
LDAP  
Listener Trigger

**Basic** **Advanced**

Target directory: out File Icon

Target file name: File Icon

Target file extension: File Icon

Specify full file path: Disabled File Icon

Path to file: File Icon

Append to File: false File Icon

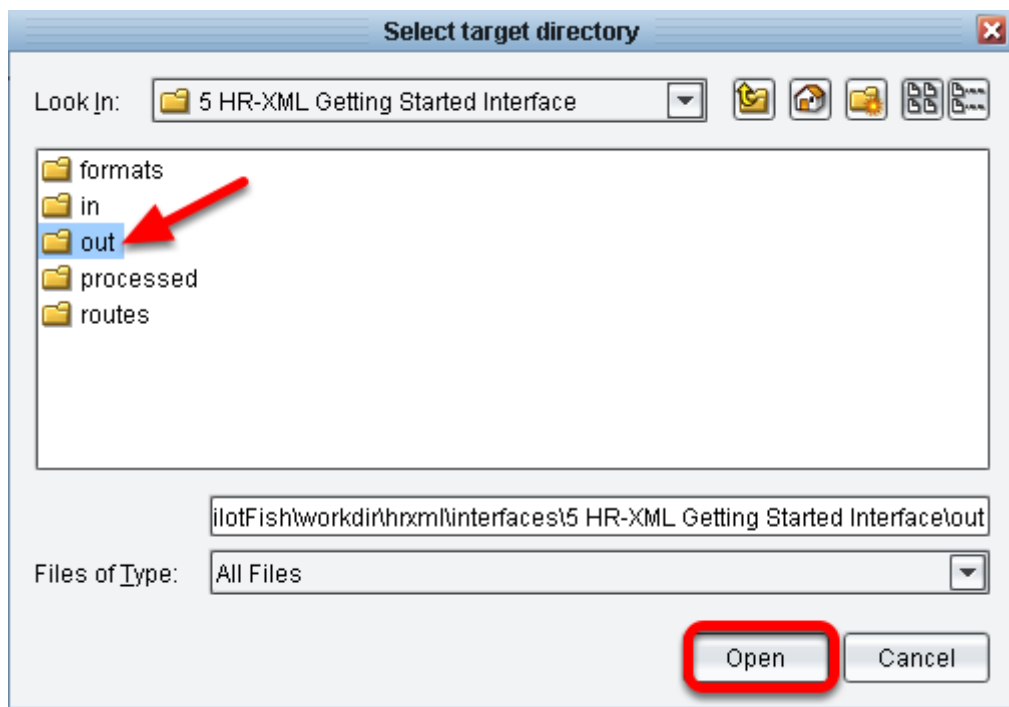
Maximum File Size: -1 File Icon

This is because you'll expect your Target system to directly consume the HR-XML 3.0 BOD that you've created. However, if you need to tweak this, or you need to convert it into a completely different file format, this Target Transformation could be used, again leveraging the Data Mapper, File Specification Editor, and

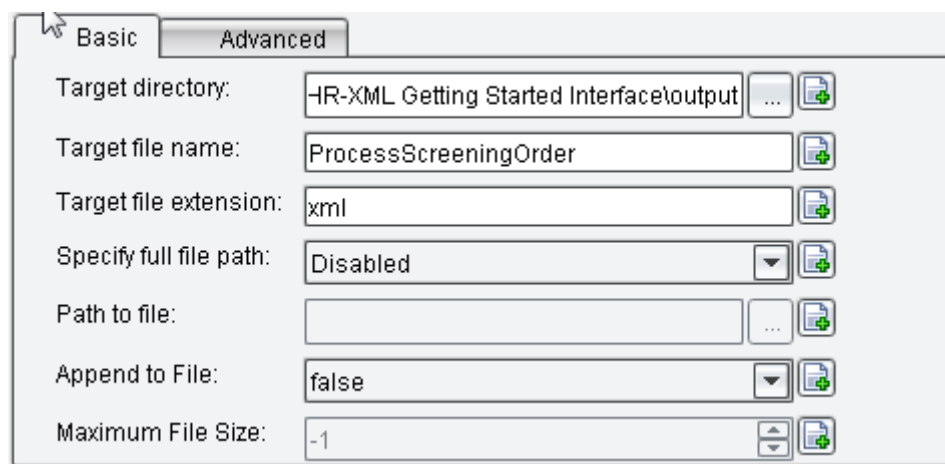
other tools to do further transformation of that data so that it appears exactly how the Target System can consume it.

Finally, click on the **Transport** stage. The Transport stage is the mirror image of the Listener. It allows you to configure how the data goes from the eiConsole to the Target system – it handles that connectivity. The various connectivity options can be exposed through expanding the Transport Type dropdown. Here you simply have a Directory configured and you'll stick with that configuration.

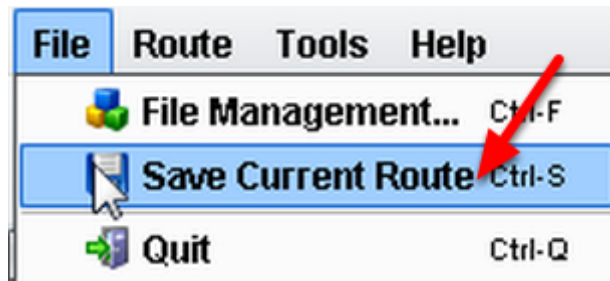
As you did on the Listener side, click the browse button. Click the **X** button next to the Target directory configuration item first, if there is no browse button.



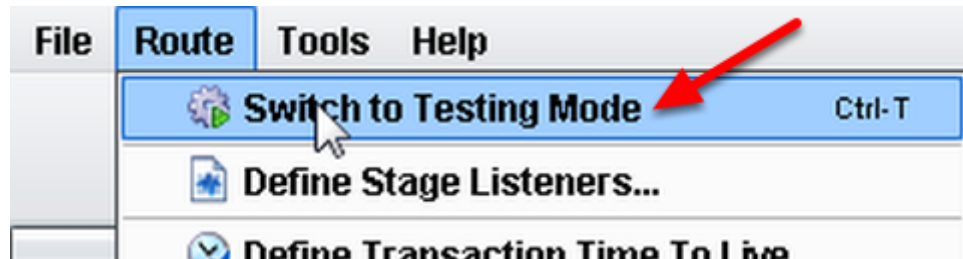
And create a new folder for the output. This time create a folder called **"out"** and click **Open**.



You'll note that you can also change the Target file name or the Target file extension.



Now that you've configured your topology you can save your interface by choosing File, **Save Current Route**.

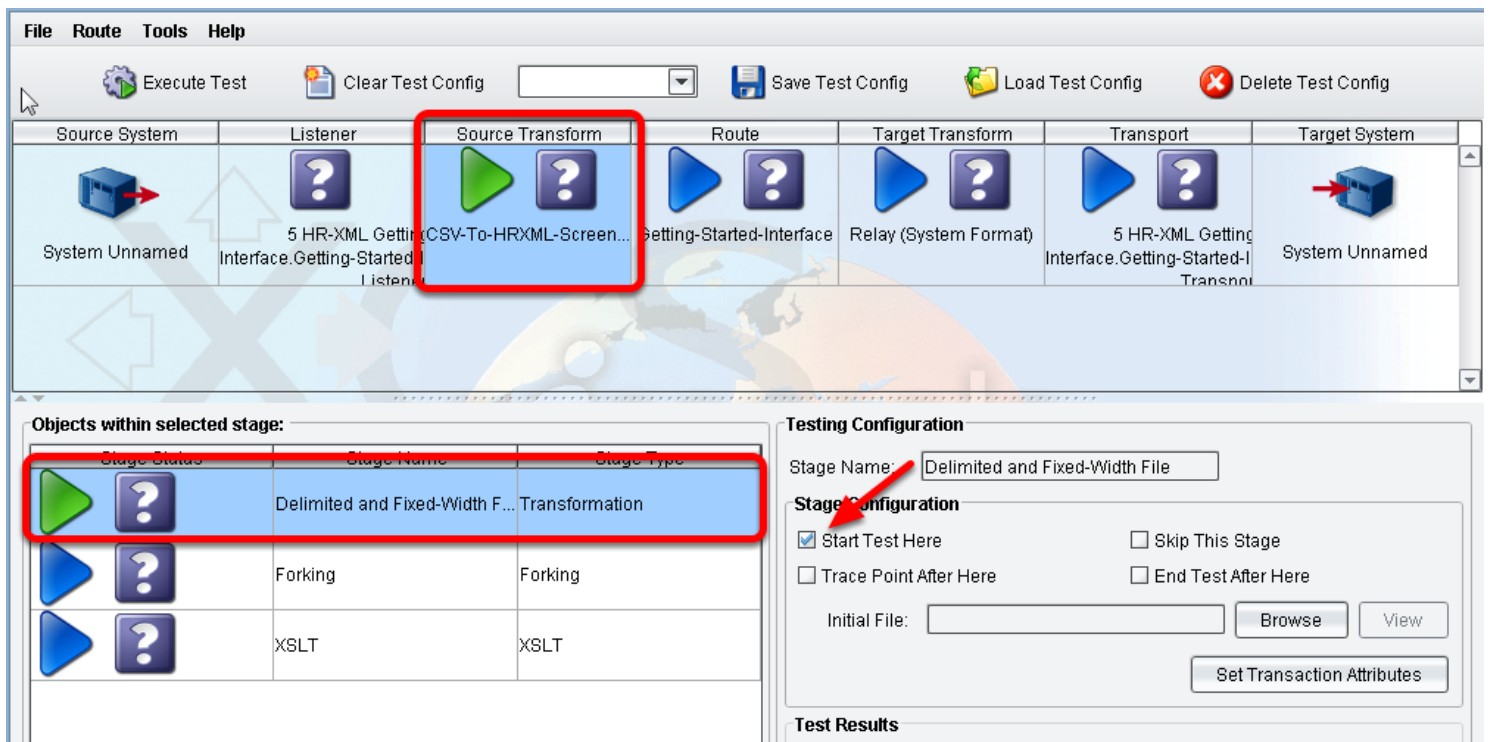


The final step is to test your work. Under the Route menu select **Switch to Testing Mode**.

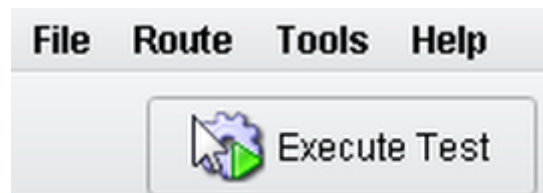
Source System	Listener	Source Transform	Route	Target Transform	Transport	Target System
System Unnamed	5 HR-XML Getting Started-Int Interface.Getting-Started-Int Listener	CSV-To-HRXML-Screenin...	Getting-Started-Interface	Relay (System Format)	5 HR-XML Getting Started-Int Interface.Getting-Started-Int Transport	System Unnamed

In testing mode you'll note that the icons between the Source System and the Target System now appear as question marks. These indicate stages of a test that you may choose to run or may choose not to run.

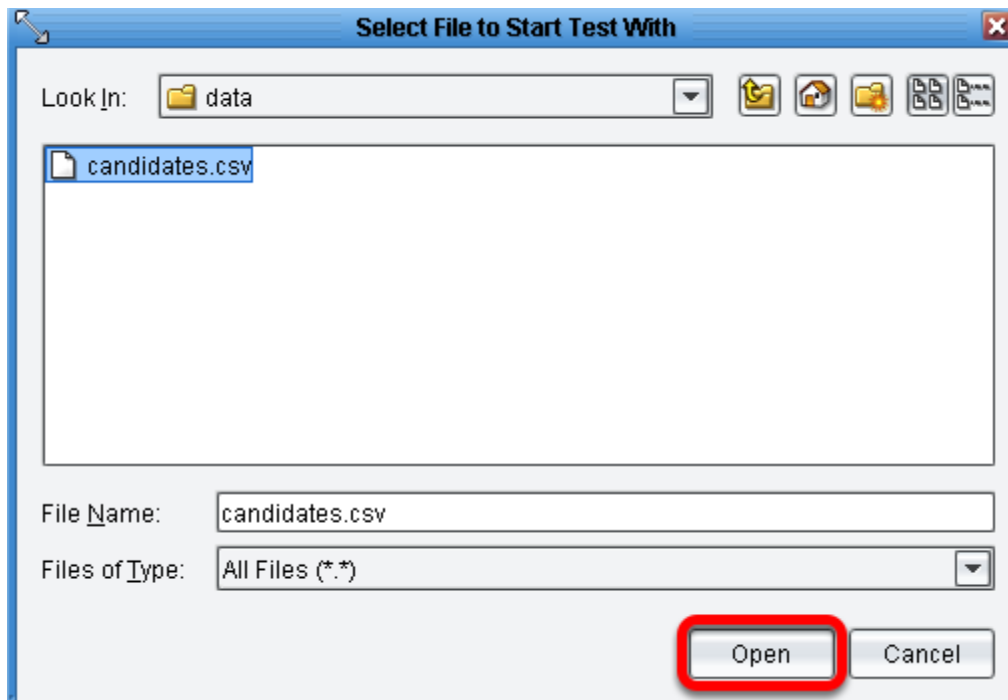




The default configuration will start the test with the Listener and allow it to run all the way through to the Transport. However, let's start with manually specifying the test file and beginning with the Source Transform. To do this, select the **Source Transform** icon in the grid and ensure that the Delimited and Fixed-Width file row is selected in the Objects within selected stage area. Then click **Start Test Here** in the Stage Configuration panel.



You'll note that the green and blue arrows above shift in order to indicate the path that this test will take. Click the **Execute Test** button.



A file dialogue will appear that will allow you to choose the file you want to start your test with. Choose the **candidates.csv** file that you're now familiar with. Again, this is located in the input folder of your tutorial subfolder. Click **Open**.

PilotFish eiConsole [5 HR-XML Getting Started Interface.Getting-Started-Interface]

File Route Tools Help

Execute Test Clear Test Config Save Test Config Load Test Config Delete Test Config

Source System	Listener	Source Transform	Route	Target Transform	Transport	Target System
System Unnamed	5 HR-XML Getting Started Interface.Getting-Started-Interface.Listener	CSV-To-HRXML-Screen... Getting-Started-Interface	Getting-Started-Interface	Relay (System Format)	5 HR-XML Getting Started Interface.Getting-Started-Interface.Transport	System Unnamed

Objects within selected stage:

Stage Status	Stage Name	Stage Type
	Delimited and Fixed-Width F...	Transformation
	Forking	Forking
	XSLT	XSLT

Testing Configuration

Stage Name: Delimited and Fixed-Width File

Stage Configuration

☒ Start Test Here ☐ Skip This Stage

☐ Trace Point After Here ☐ End Test After Here

Initial File: h:\workdir\hrxml\data\candidates.csv Browse View

Set Transaction Attributes

Test Results

TX ID	Time	Percent	Status
1	21:22:02.786	22.89	Success

View Stage Output

As each stage completes the question mark will either turn into a check, indicating success, or an X indicating failure. Here we see it all completes successfully, green arrows.

**Testing Configuration**

Stage Name:

**Stage Configuration**

☒ Start Test Here
 ☐ Skip This Stage  
☐ Trace Point After Here
 ☐ End Test After Here







Initial File:

**Test Results**

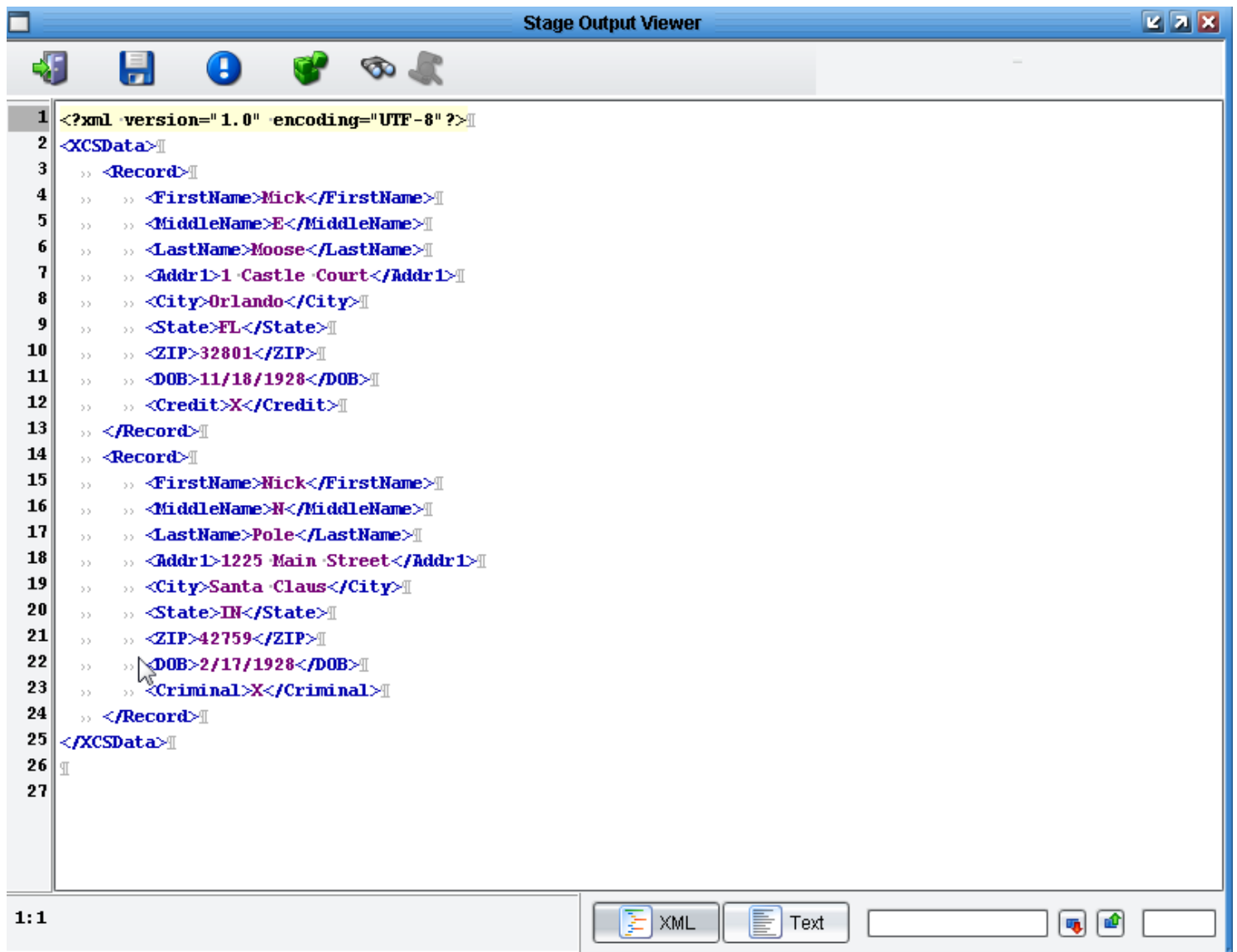
TX ID	Time	Percent	Status
1	21:22:02.786	22.89	Success

You can now take a look at how the data appeared at each point in the process. Clicking the **View** button next to the initial file will again show you your input.

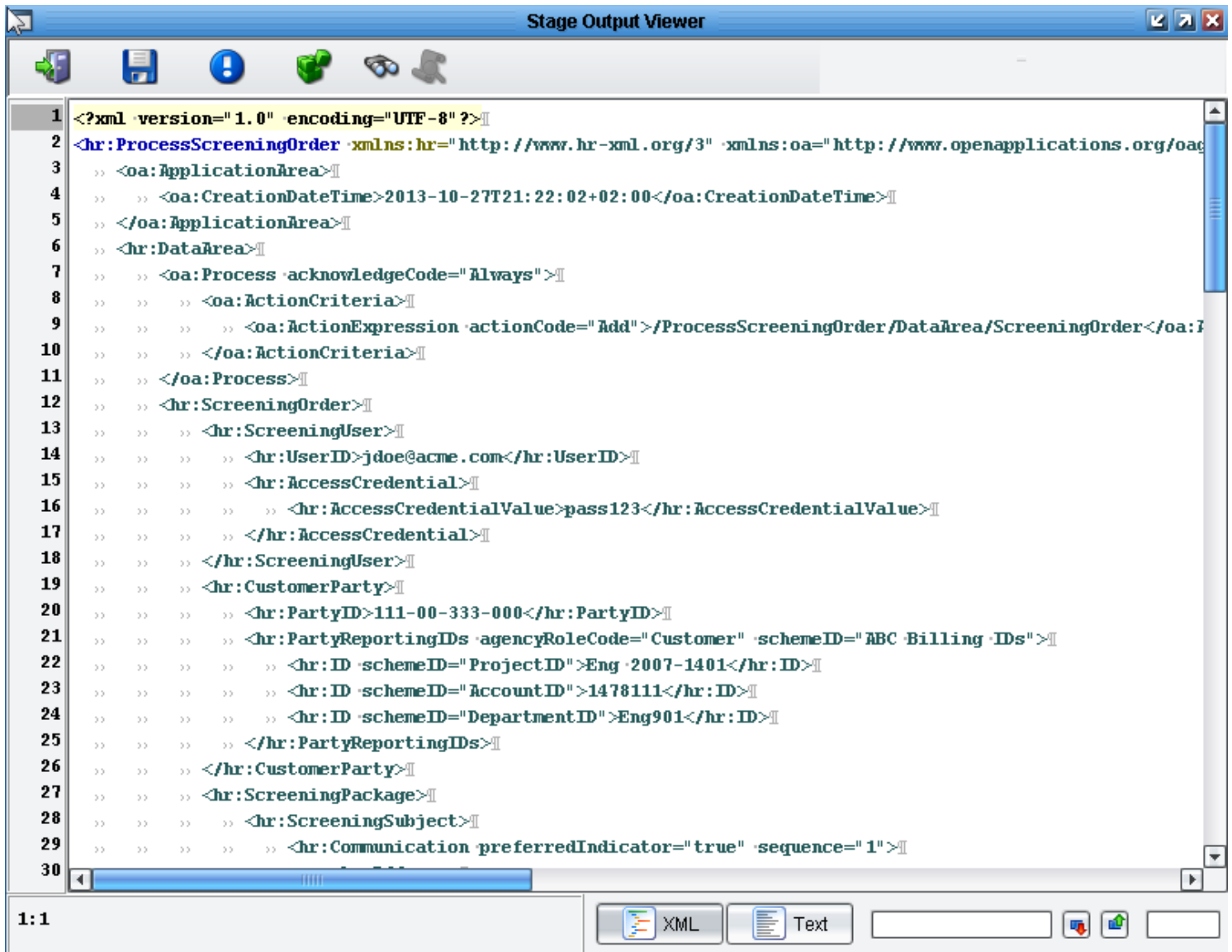
**Objects within selected stage:**

Stage Status	Stage Name	Stage Type
 	Delimited and Fixed-Width F...	Transformation
 	Forking	Forking
 	XSLT	XSLT

Double clicking on the Delimited and Fixed-Width file sub-stage in the Objects within selected stage area...



This will bring up the XML representation of that flat file.



Next you used XSLT generated in the Data Mapper to convert that data into the HR-XML 3.0 BOD. Double clicking the XSLT stage will show you the generated Process Screening Order.

```
34 >> >> >> >> >> >> <hr:CountryCode>US</hr:CountryCode>]]
35 >> >> >> >> >> >> <oa:CityName>Orlando</oa:CityName>]]
36 >> >> >> >> >> >> <oa:CountrySubDivisionCode>FL</oa:CountrySubDivisionCode>]]
37 >> >> >> >> >> >> <oa:PostalCode>32801</oa:PostalCode>]]
38 >> >> >> >> >> >> </hr:Address>]]
39 >> >> >> >> >> >> </hr:Communication>]]
40 >> >> >> >> >> >> <hr:ScreeningSubjectName>]]
41 >> >> >> >> >> >> <hr:ScreeningPersonName>]]
42 >> >> >> >> >> >> <hr:ID>01</hr:ID>]]
43 >> >> >> >> >> >> <hr:PersonName nameTypeCode="Current Name">]]
44 >> >> >> >> >> >> <oa:GivenName>Mick</oa:GivenName>]]
45 >> >> >> >> >> >> <hr:MiddleName>E</hr:MiddleName>]]
46 >> >> >> >> >> >> <hr:FamilyName>Moose</hr:FamilyName>]]
47 >> >> >> >> >> >> </hr:PersonName>]]
48 >> >> >> >> >> >> </hr:ScreeningPersonName>]]
49 >> >> >> >> >> >> <hr:FreeFormBirthDate>]]
50 >> >> >> >> >> >> <hr:FormattedDateTime>1928-11-18</hr:FormattedDateTime>]]
51 >> >> >> >> >> >> </hr:FreeFormBirthDate>]]
52 >> >> >> >> >> >> </hr:ScreeningSubject>]]
53 >> >> >> >> >> >> <hr:Screening>]]
54 >> >> >> >> >> >> <hr:PartyReportingIDs agencyRoleCode="Requester" schemeID="ATS Co.">]]
55 >> >> >> >> >> >> <hr:ID schemeID="CandidateID">1111478111</hr:ID>]]
56 >> >> >> >> >> >> <hr:ID schemeID="RequisitionID">998Eng901</hr:ID>]]
57 >> >> >> >> >> >> </hr:PartyReportingIDs>]]
58 >> >> >> >> >> >> <hr:PartyReportingIDs agencyRoleCode="Customer" schemeID="ABC Billing IDs">]]
59 >> >> >> >> >> >> <hr:ID schemeID="ProjectID">Eng 2007-1401</hr:ID>]]
60 >> >> >> >> >> >> <hr:ID schemeID="AccountID">1478111</hr:ID>]]
61 >> >> >> >> >> >> <hr:ID schemeID="DepartmentID">Eng901</hr:ID>]]
62 >> >> >> >> >> >> </hr:PartyReportingIDs>]]
63 >> >> >> >> >> >> </hr:ScreeningSubjectName>]]
```

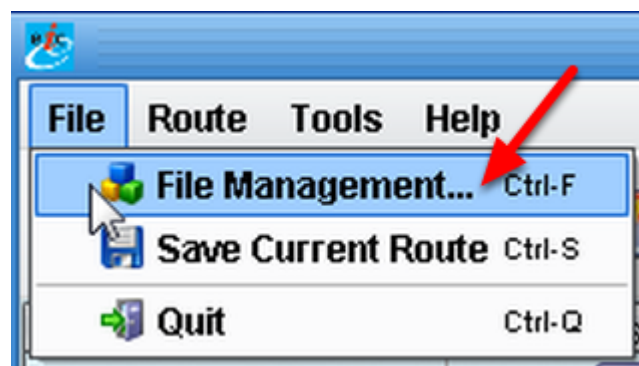
You'll see the values that you mapped dynamically populated into the template.

```

File Edit Options Encoding Help
<?xml version="1.0" encoding="UTF-8"?><hr:ProcessScreeningOrder
xmlns:hr="http://www.hr-xml.org/3"
xmlns:udt="http://www.openapplications.org/oagis/9/unqualifieddatatypes/1.1"
xmlns:oa="http://www.openapplications.org/oagis/9"
systemEnvironmentCode="Production" releaseID="3.0"
languageCode="en-US"><oa:ApplicationArea><oa:CreationDateTime>2013-10-27T21:39:01
+02:00</oa:CreationDateTime></oa:ApplicationArea><hr:DataArea><oa:Process
acknowledgeCode="Always"><oa:ActionCriteria><oa:ActionExpression
actionCode="Add"/></oa:ActionCriteria></oa:Process><hr:ScreeningOrder><hr:ScreeningUser><hr:Use
rID>jdoe@acme.com</hr:UserID><hr:AccessCredential><hr:AccessCredentialValue>pass1
23</hr:AccessCredentialValue></hr:AccessCredential></hr:ScreeningUser><hr:Custom
erParty><hr:PartyID>111-00-333-000</hr:PartyID><hr:PartyReportingIDs schemeID="ABC
Billing IDs" agencyRoleCode="Customer"><hr:ID schemeID="ProjectID">Eng
2007-1401</hr:ID><hr:ID schemeID="AccountID">1478111</hr:ID><hr:ID
schemeID="DepartmentID">Eng901</hr:ID></hr:PartyReportingIDs></hr:CustomerParty><
hr:ScreeningPackage><hr:ScreeningSubject><hr:Communication sequence="1"
preferredIndicator="true"><hr:Address><oa:LineOne>1 Castle
Court</oa:LineOne><oa:LineTwo/><hr:CountryCode>US</hr:CountryCode><oa:CityName>Or
lando</oa:CityName><oa:CountrySubDivisionCode>FL</oa:CountrySubDivisionCode><oa:P
ostalCode>32801</oa:PostalCode></hr:Address></hr:Communication><hr:ScreeningSubje
ctName><hr:ScreeningPersonName><hr:ID>01</hr:ID><hr:PersonName
nameTypeCode="Current
Name"><oa:GivenName>Mick</oa:GivenName><hr:MiddleName>E</hr:MiddleName><hr:Family
Name>Moose</hr:FamilyName></hr:PersonName></hr:ScreeningPersonName></hr:Screening
SubjectName><hr:FreeFormBirthDate><hr:FormattedDateTime>1928-11-18</hr:FormattedD
ateTime></hr:FreeFormBirthDate></hr:ScreeningSubject><hr:Screening><hr:PartyRepor
tingIDs schemeID="ATS Co." agencyRoleCode="Requester"><hr:ID
schemeID="CandidateID">1111478111</hr:ID><hr:ID

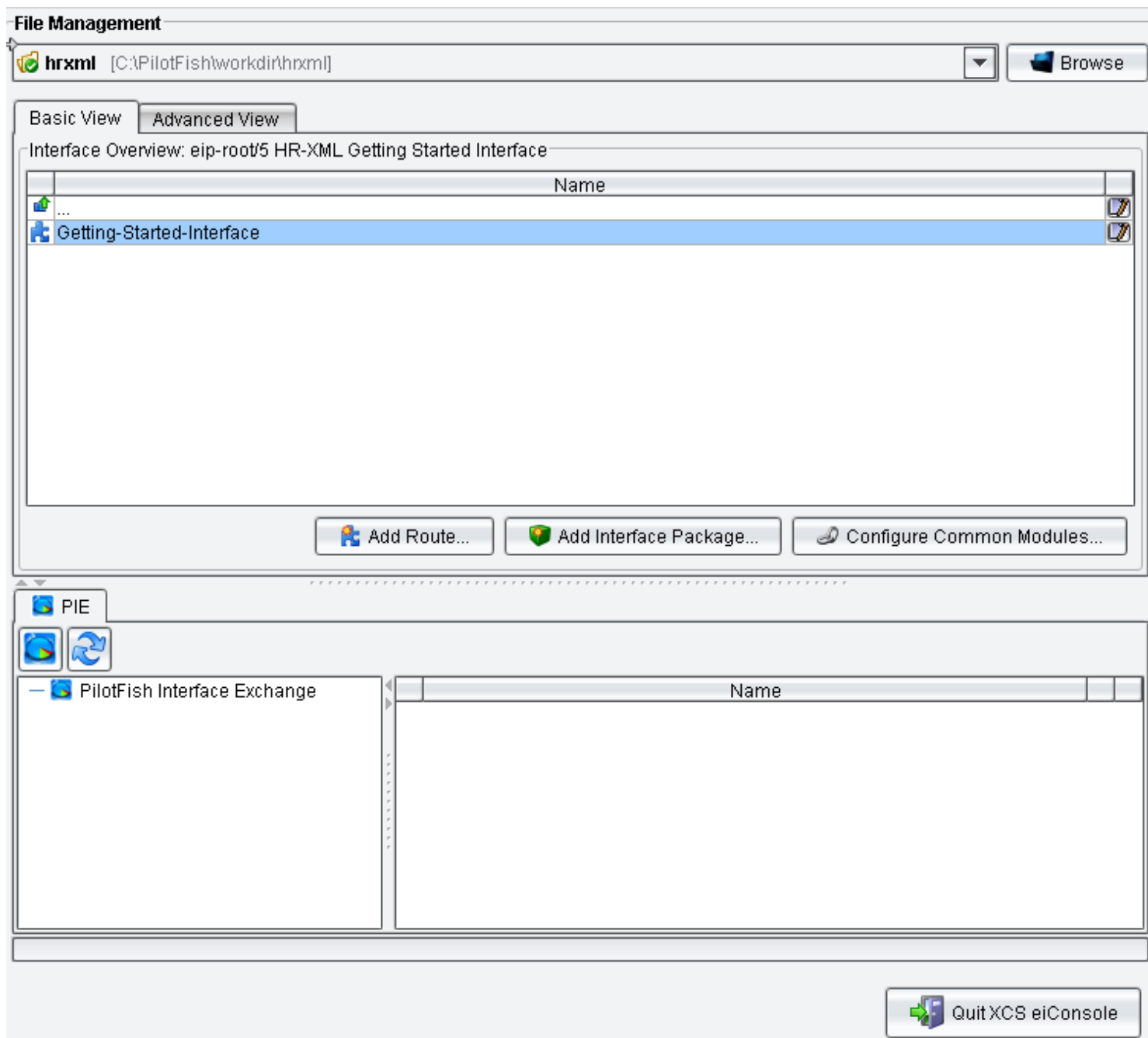
```

The Routing and Target Transformation stages didn't do much work here. And finally you dropped the file in a directory. Navigating to the output folder that you created you should see a file called ProcessScreeningOrder.xml containing the same view that you saw as the output of your process.



Congratulations! You've now built your first HR-XML 3.0 interface using the eiConsole for HR-XML. The next step, of course, is to deploy this interface somewhere. To do that, save your interface again. Then choose **File, File Management**.





Your Getting Started interface is now ready for deployment to an eiPlatform server.

Thanks again for downloading the eiConsole for HR-XML. We're absolutely sure you'll find this is the best, fastest, most cost effective way to build these interfaces. If you have any more questions, please visit our website ([www.pilotfishtechology.com](http://www.pilotfishtechology.com)) or give us a call (860-632-9900)... we are always happy to offer some free training.