ChainBuilder ESB

Visual Enterprise Integration™

Version 1.0

Map Editor Guide

2800 Corporate Exchange Drive
Suite 260
Columbus, OH 43231

Acknowledgements

Trademarks

The following trademarks and acknowledgments apply to the information presented in this manual:

- ChainBuilder is a registered trademark of Bostech Corporation.
- Adobe and Acrobat Reader are registered trademarks of Adobe, Inc.
- Java is a registered trademark of Sun Microsystems, Inc.
- Windows (NT, 2000, XP, and Server 2003), .NET Framework, Internet Information Services (IIS) are registered trademarks of Microsoft Corporation.

Credits

The following third-party products are used within the ChainBuilder product, and acknowledgments apply to the information presented in this manual:

- Acrobat Reader is created and licensed by Adobe, Inc.
- This product includes software developed by the Apache Software Foundation (http://www.apache.org/)
- This product includes software developed by Eclipse (http://www.eclipse.org/)

Table of Contents

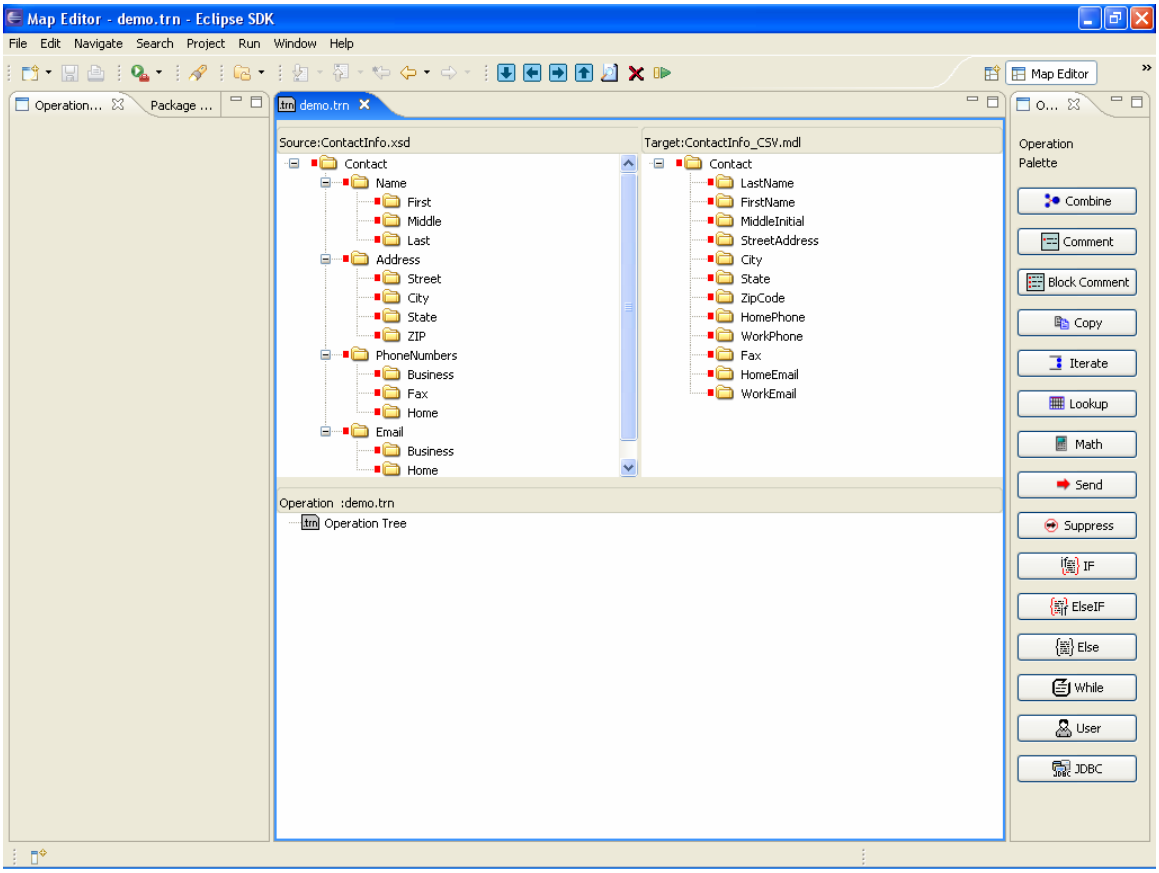
1. Introduction

1.1. Message Mapping

Message mapping is used to transform a data message from one format to another format in an automated, highly configurable manner.

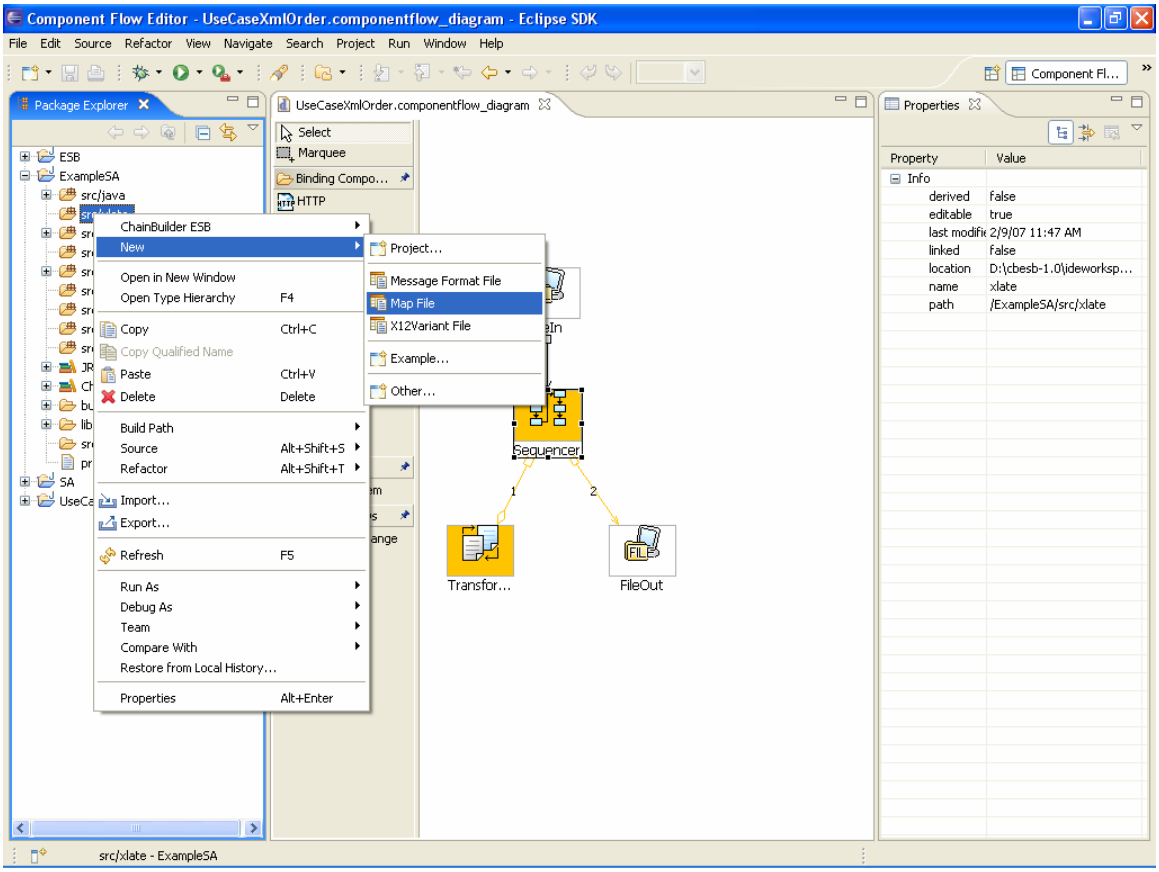
1.2. Map Editor

The Map Editor allows the user to configure mapping assignments and logic at the individual field level. The main parts of the editor are the OperationProperties tab on the left, the Source and Target trees in the top middle, Operation Tree in the lower middle, and Operation Palette to the right. The OperationProperties tab contains configurable property settings relating to an individual operation. The Source and Target trees show the source (input) and target (output) data formats, including repetition information. The Operation Tree lists all operations that have been configured in this map, and shows the order and logic flow that will be executed. The Operation Palette includes buttons which create the various mapping operations.

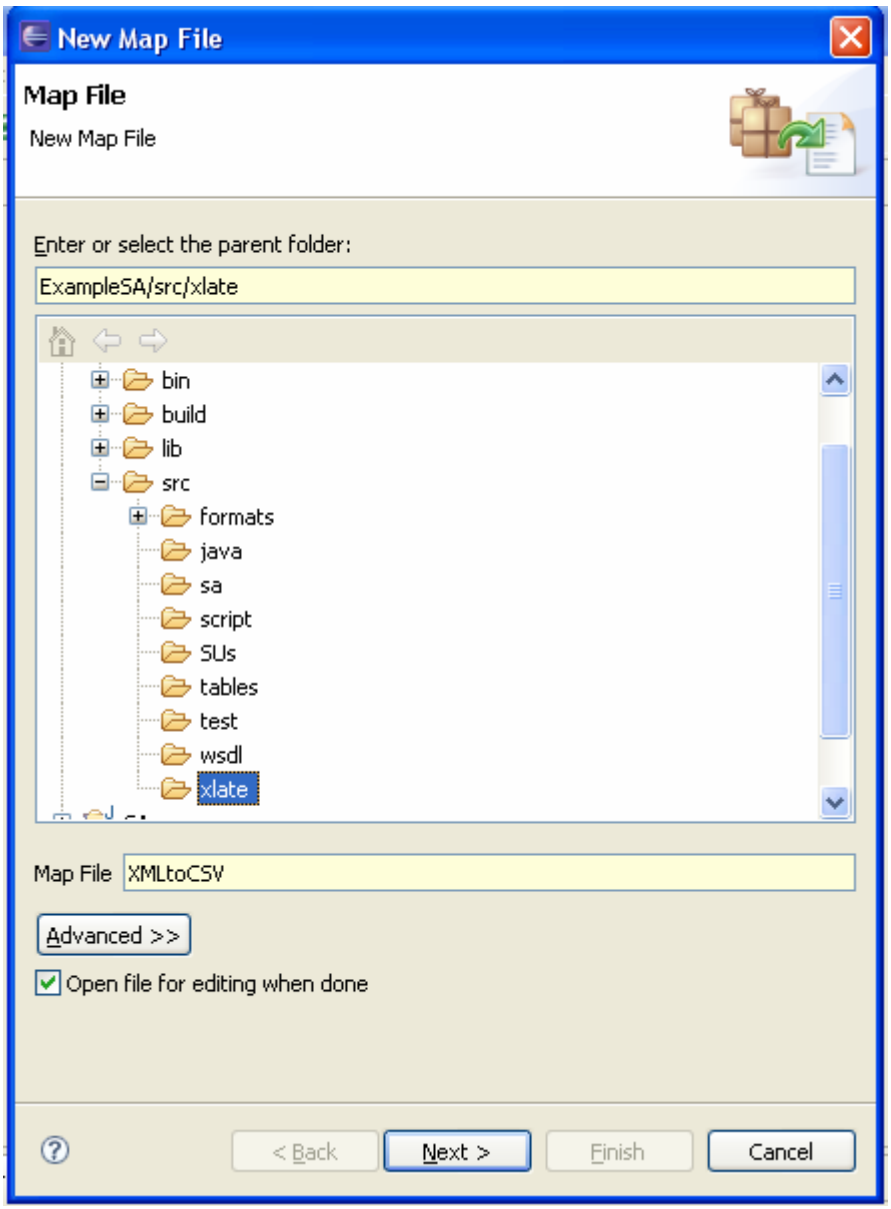


2. Creating a New Message Map

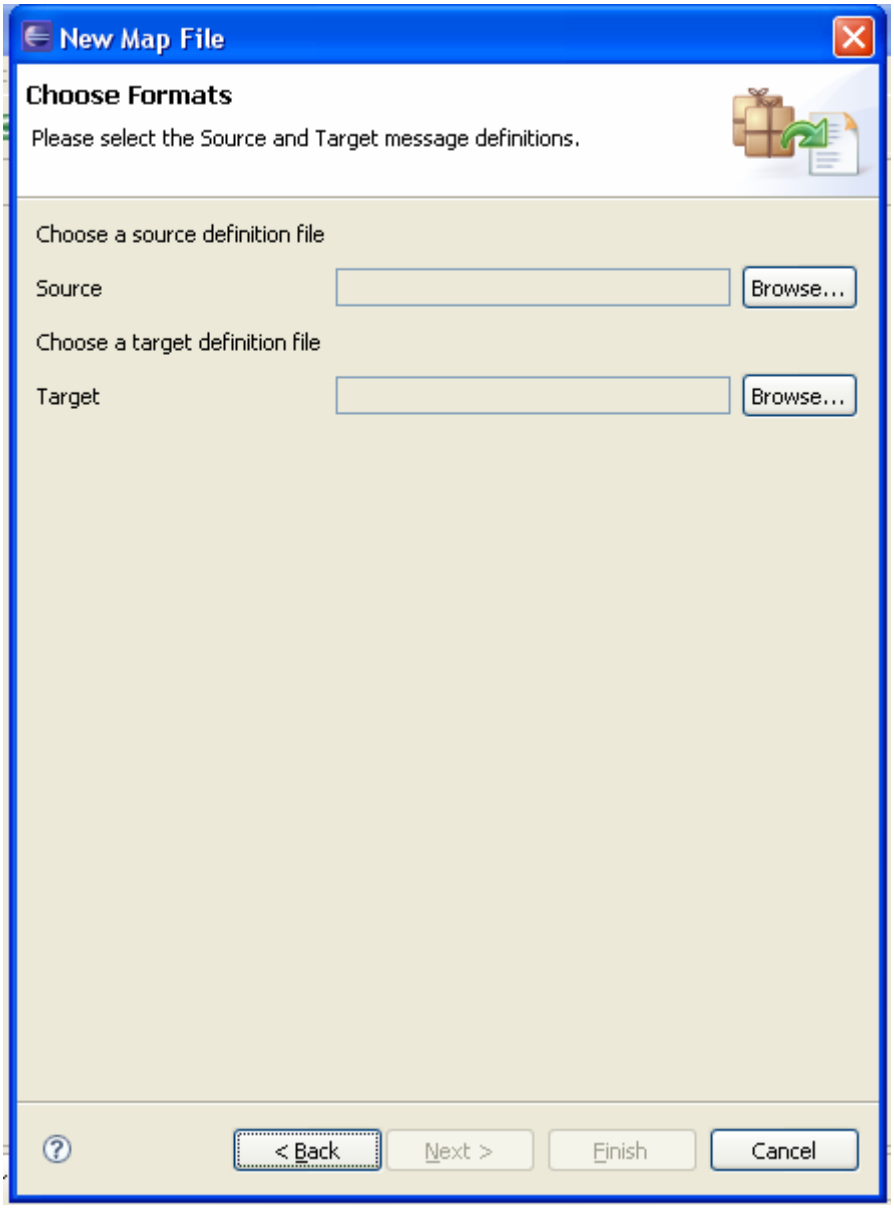
To create a new message map, right-click on the src/xlate directory under the appropriate Service Assembly Project, then select New -> Map File.



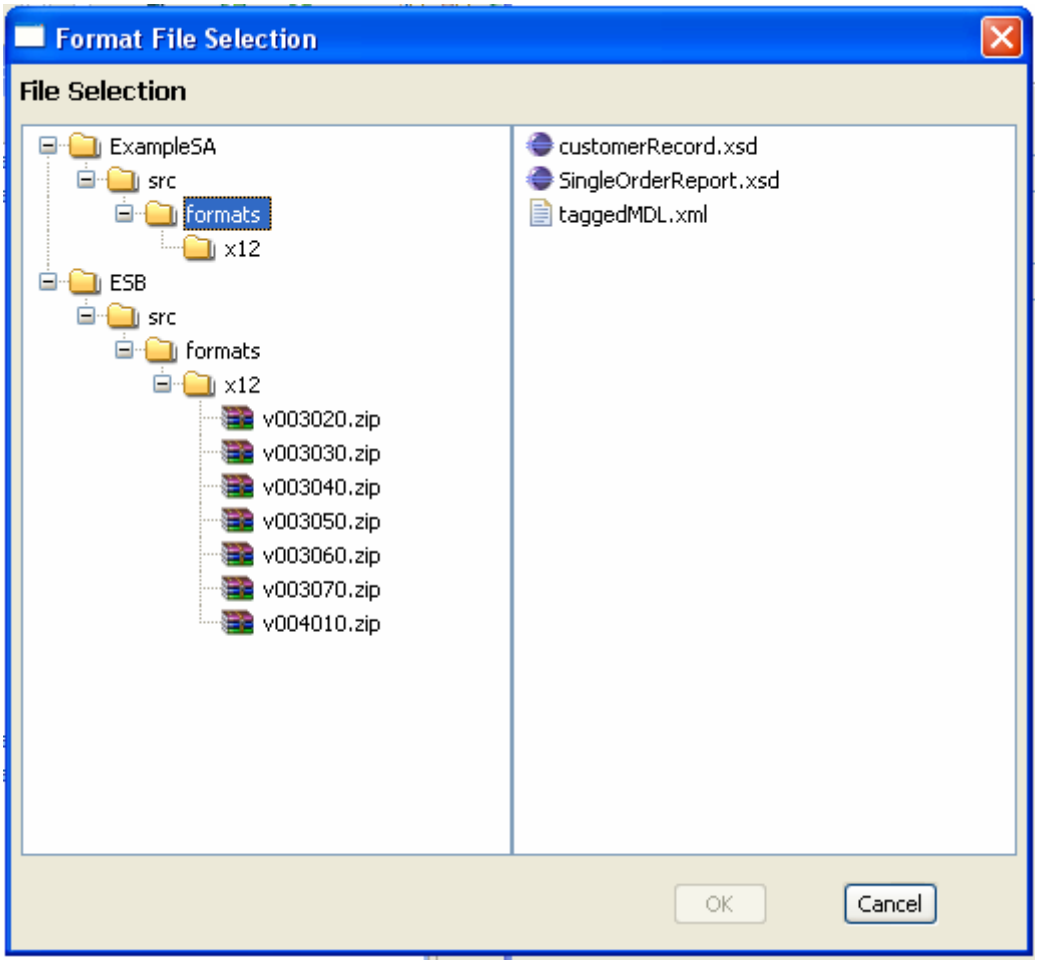
Follow the wizard, first by providing a name for the map file. Maintain the provided .trn file extension.



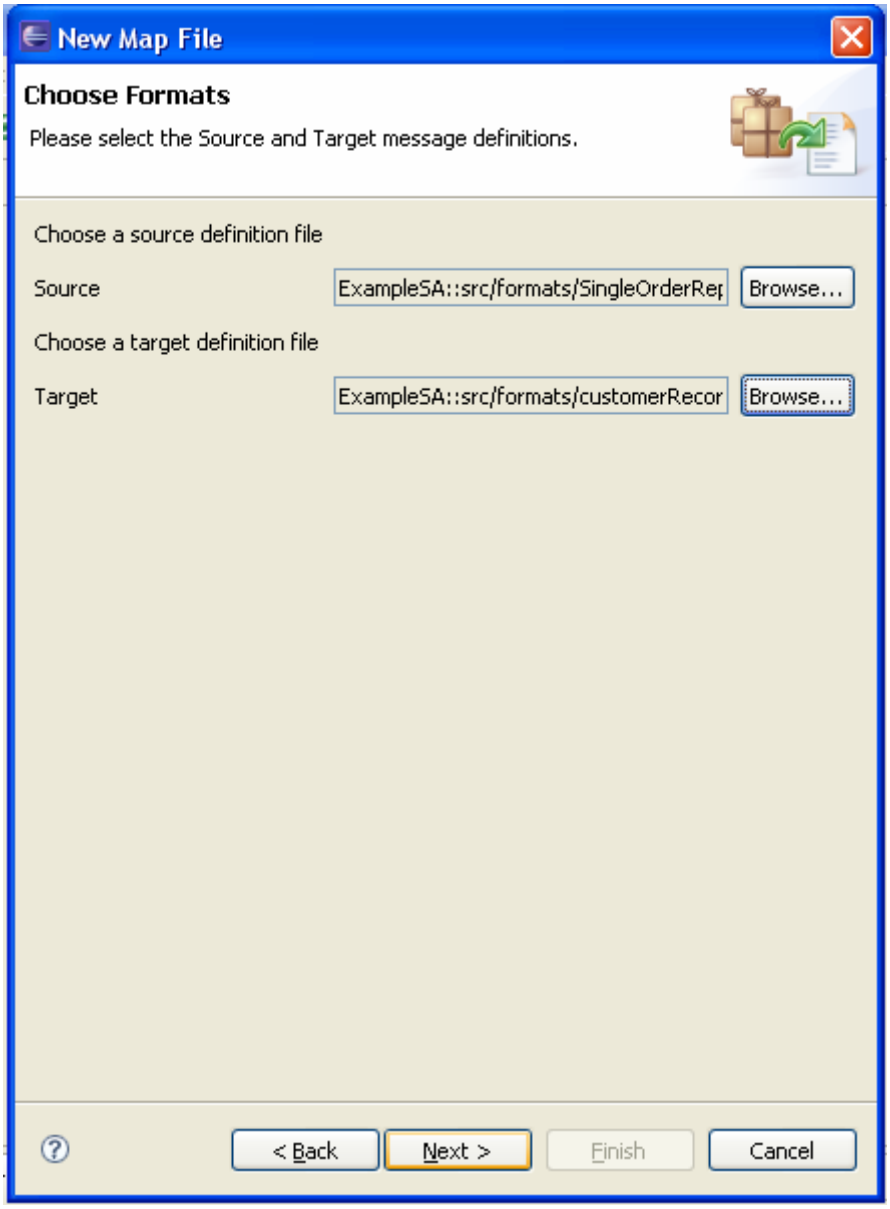
Next, choose source and target formats. Both formats may be the same, or two different formats may be used. Use the Browse button to open the Resource Selection screen, which allows for the selection of the appropriate message format.



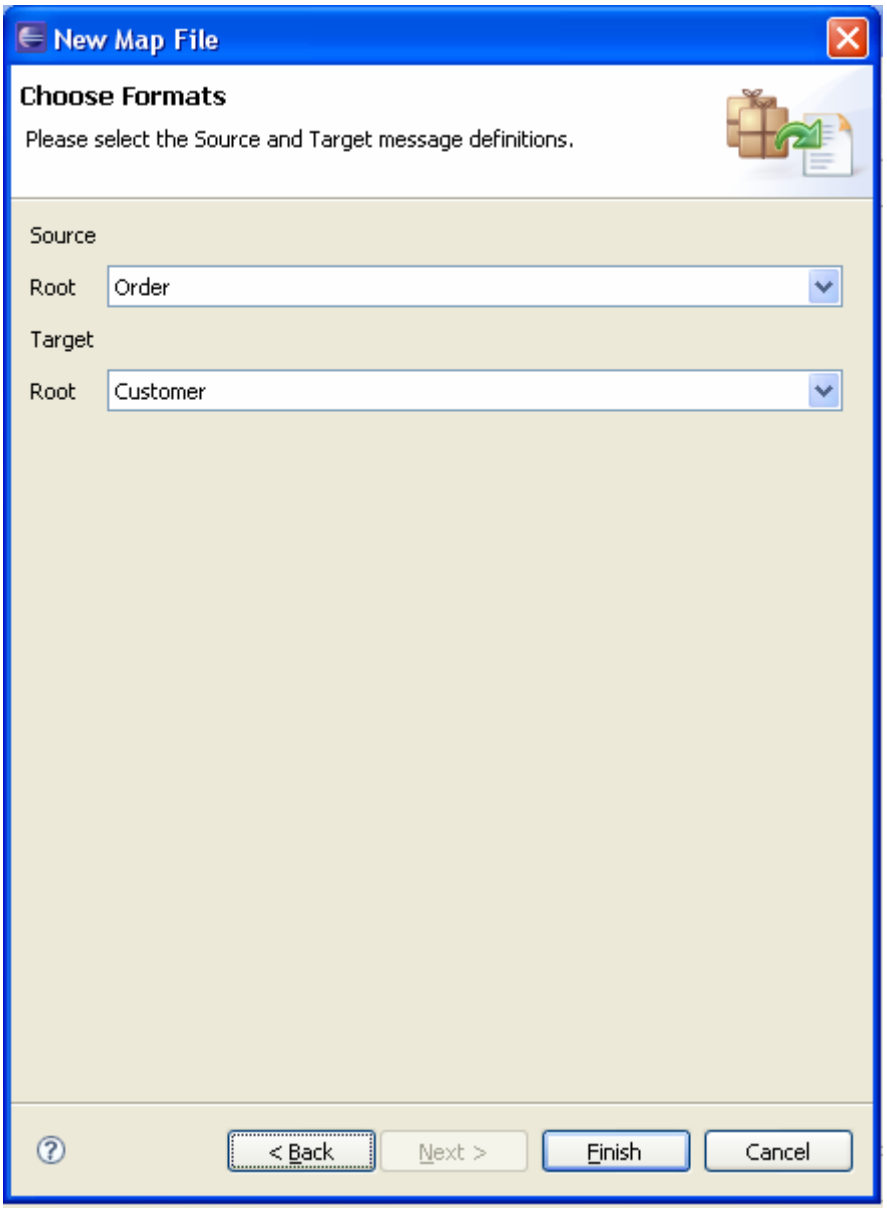
Click on the "Browse" button to bring up the format selection dialog. Choose the appropriate format definition file.



After selecting the input format, select the output format. Continue to the next screen after both formats have been selected and appear properly in the Location field.

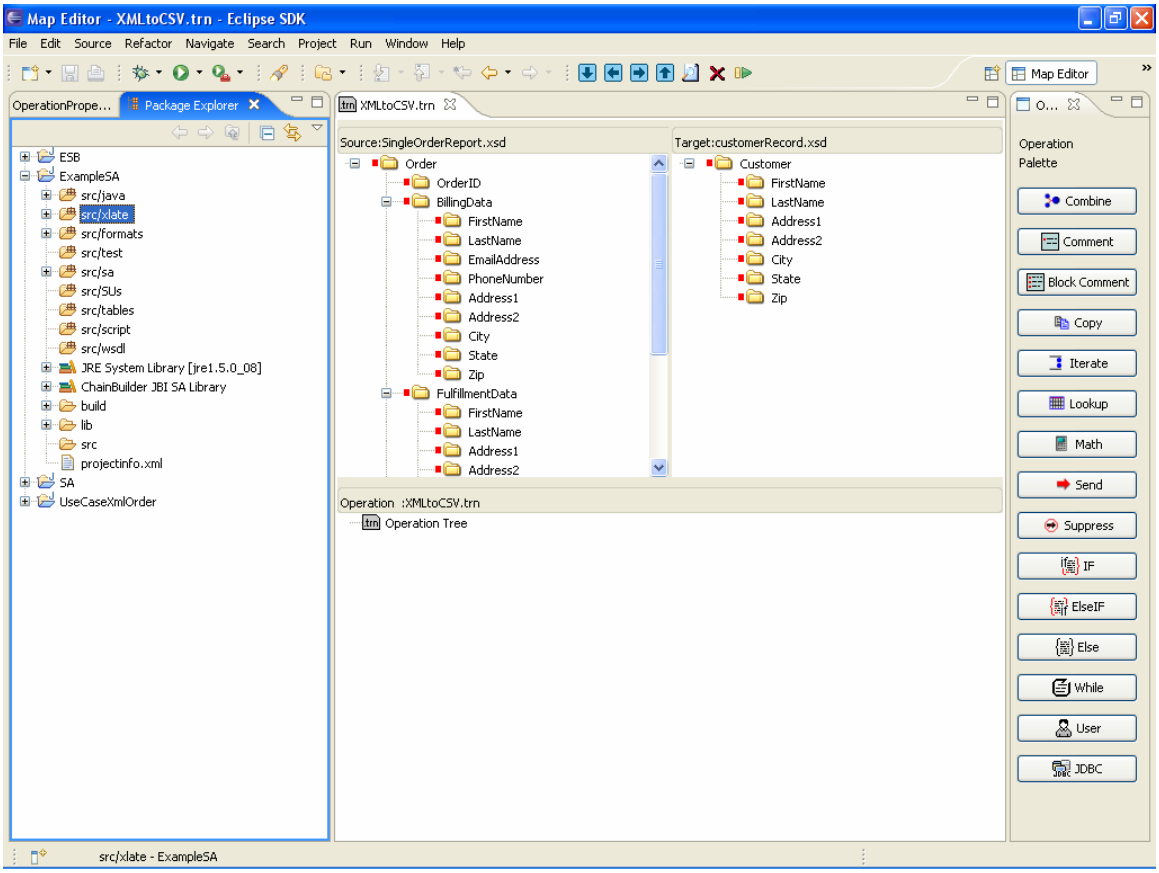


From the drop-down lists, select a root node for each format.



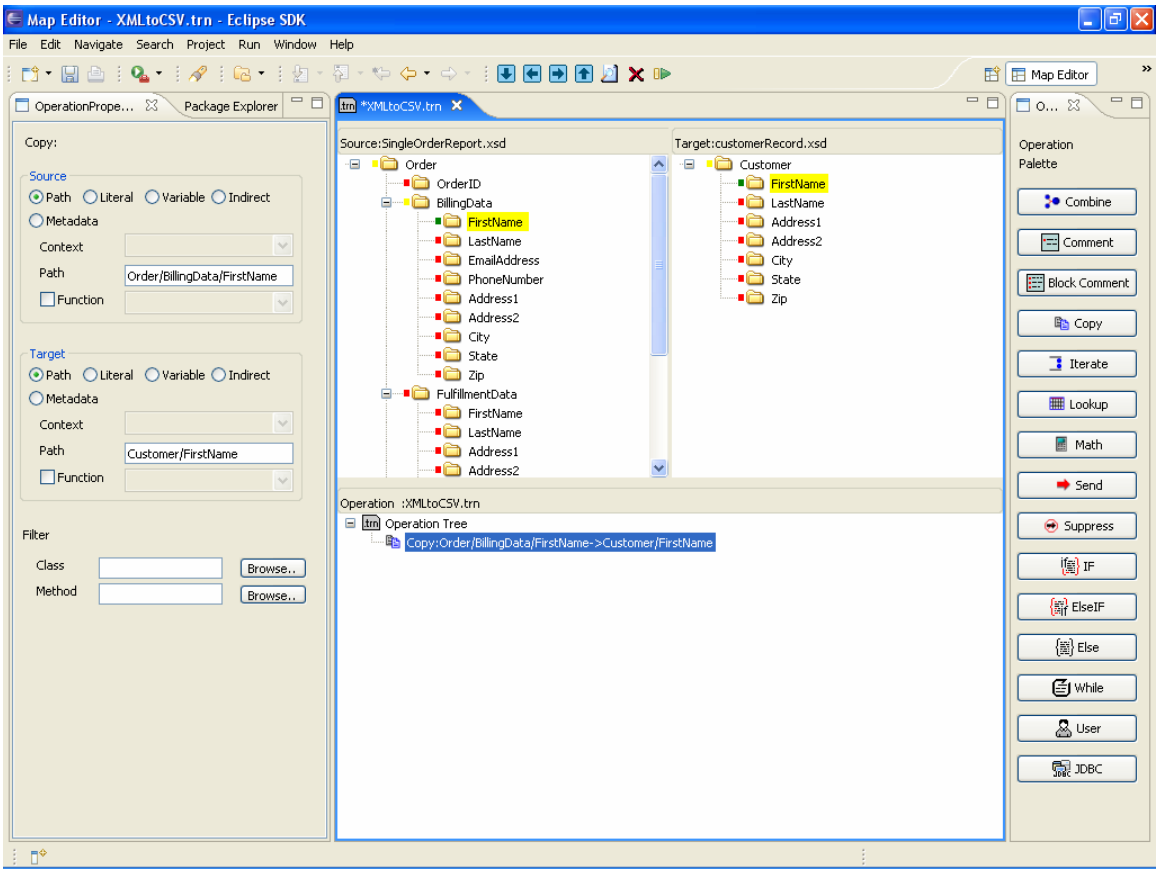
After the root nodes have been selected, click Finish to exit the wizard.

This will open the Map Editor Perspective. Note the OperationProperties tab on the left. You may need to manually switch to this tab. It is empty currently but will contain operation configuration information once an operation has been created and selected.



3. Adding Operations to the Map

Adding operations to the map is as easy as dragging a field from the Source window to the proper destination field in the Target window. This will add a Copy operation. For example, dragging the BuyerName field from the Source window to the Target window BuyerName field creates a copy operation between those two fields. The red boxes next to each involved field change to green, showing that those fields are involved in a mapping operation.



In addition to creating mappings via the field dragging method (which creates Copy operations) a variety of operations are available in the Operation Palette (right side of the above image). Clicking on an operation in the Operation Palette will create an empty version of that operation below the currently selected operation in the Operation pane. This action then needs to be properly configured.

3.1. Specifying Target and Source

The configuration of the OperationProperties tab for a newly created Operation is dependent on the type and context of the operation; certain fields are important only for certain operation types or for nested operations.

Source and Target Operations have 4 different I/O modes: Path, Literal, Variable, and Indirect. Path allows for a reference to a field in an inbound or outbound data format; these values are filled in the Path box by dragging the appropriate field (from the center message format window) into the Source or Target box. Use of the Literal option enables the Value field, into which the user can enter a string directly. This would generally be used on the Source to hard-code an input value. The Variable mode allows the user to store information into and retrieve information from local variables. These variables can be named with a string identifier. Once the variable has been initialized (by using the variable as a Target) it will become available for use in the Source within the drop down box. Indirect requires the user to select a variable; this variable must contain the value of a target field descriptor. The field which is named in the variable will be used; the effect will be just as if that field was entered directly into the Path.

The Message Mapper operates in a top-down fashion, executing the operations within the Operation Tree in the order in which they are listed. Therefore, an operation which places data into a certain Target field will overwrite any data that had previously been entered into that field.

The Iterate structure allows for the definition of a Context, or named iterative loop. This context is defined at the bottom of the OperationProperties. While there is a default value, it can be modified to create descriptive context names. This Context can then be used by Copy and other statements that are nested within the Iterate, in order to allow for selection of the proper repetition within the given looping structure.

Looping operations allow for operations to be nested within them. The right and left arrows in the icon bar near the top of the window allow the user to move operations into and out of a nested operation.

3.2. Specifying a Filter

Filters provide an easy method to perform string manipulation and other light data handling. The Filter function is not an operation on the palette but instead is usable from the Operation Properties tab on a variety of operations, including Copy. Filters are individual methods which can all be placed within the same class if desired. The example below shows a basic Filters class with a single method. The method should receive a string input and return a string.

```
public class Filters {

	public String trimQuotes (String in) {
		// removes all double-quote characters (") from input data
		in = in.replaceAll("\"", "");
		return in;
	}
}
```

This compiled class should be stored in the src/java directory. In order to use the Filter in a map operation, within the Operation Properties, first browse to select the class, then browse to select the method within the selected class.

4. Detailed Operation Descriptions

4.1. Combine Operation

The Combine operation acts to concatenate together multiple Sources into a single Target. As a default, a single Source is listed in the OperationProperties tab. Clicking the Add Source button adds an additional Source to the OperationProperties. As many Sources as needed may be added in this manner.

4.2. Comment Operation

Comment operations allow for the storage of helpful information about the configuration of the Map. Each Comment includes a large box into which free text may be entered by the user.

4.3. Block Comment Operation

The Block Comment operation allows the user to comment-out existing operations, so that they will not be utilized. Unlike other operations, the Block Comment does not add an additional operation to the Operation Tree. Instead, it is used by first clicking on the operation (or operations, by shift-clicking) to be modified, then clicking the Block Comment button. The now commented operations will be grayed out and will not function unless the process is reversed by again selecting the operation(s) and clicking Block Comment.

4.4. Copy Operation

Copy is the basic command used to map the entire contents of one field (or variable) into the Target.

4.5. Iterate Operation

Iterate is used to loop through a repeating value in the source data message or to populate repeating values in the target message. The Iterate operation associates a context name with some repeating structure in the message. A context name will be defaulted, but may be modified to make the name more specific to its usage. Within the Iterate operation, other operations may be nested. These nested operations will be executed (in order) once for each repetition of the repeating field that occurs in the source data message. A target iteration creates a new repetition in the output message each time the iterate statement is entered. Operations nested within an iterate should reference the appropriate context in order to utilize data elements from the proper loop iteration. By nesting Iterate operations and choosing the proper Context within the Source of the nested operation, it is possible to nest Iterate statements in order to properly parse input data formats with nested repeating elements.

4.6. Lookup Operation

The Lookup operation utilizes a predefined lookup table. The Source data will be used as the key to the table, and the table output value will be placed in the Target. The lookup table should be named with a .tbl extension and placed in the src/tables directory. The table is an XML file and should use the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
        <comment>LookUp Table</comment>
        <entry key="key1">value1</entry>
        <entry key="key2">value2</entry>
</properties>
```

If the Source field is not found as a key in the table, the default value (defined in operation properties) will be used. The Lookup operation is case sensitive.

4.7. Math Operation

The Math operation allows for binary mathematical operations. This can be accomplished with a single source and a second numeric value within the expression, or two sources with only the mathematical operand in the expression. Allowable mathematical operations are:
Addition: +
Subtraction: -
Multiplication: *
Division: /
Modulus: %
Exponent: ^

4.8. Send Operation

The Send operation will create an output message from the mappings that have been performed up to the point when the Send is encountered. Specifically, the results of operations listed below the Send will not be included in the resultant message as they have not occurred at the time the Send is executed. Typically, this is used to create an outbound message for every iteration, when the inbound data format contains repeating elements but the target format does not.

4.9. Suppress Operation

Suppress is used to prevent the normal end of transformation message from being generated. When the Transformation reaches its natural end, the completed message is automatically created, unless a Suppress operation has previously been executed. The Suppress operation requires no configuration, as it has no relevant properties. Suppress is commonly used after an Iterate which contains a Send operation; this prevents the lat message from being sent twice.

4.10. If Operation

The If operation is the basic conditional expression, allowing for a certain set of additional operations to be executed only in the case that the expression is evaluated to true. The comparison type can be set in the Arithmetic Type box in the OperationsProperties tab. An Arithmetic Type of String can be used to make comparisons between two string values, which Integer and Floating Point Types should be used to make comparisons between those data types. This operation is binary in nature, a requirement which can be fulfilled by either using two Source inputs (the second Source is available by clicking the Add Source button in the OperationProperties tab) or by referencing a single source with the right-hand side of the comparison in the expression.

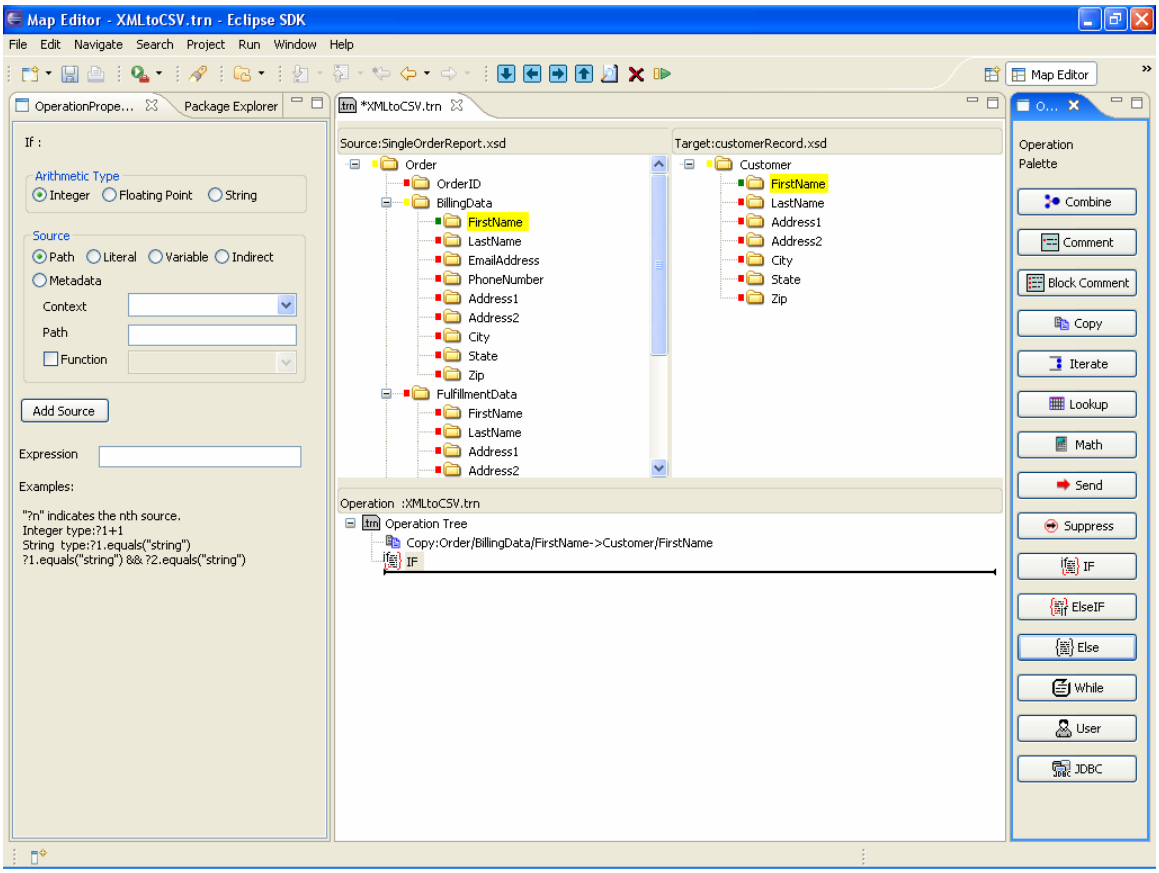
Expressions should be formatted as shown below, with ?# used to indicate which Source is being compared. Or conditions can be created with double pipes (||) while and conditions are created with double ampersands (&&).

Allowable string expressions include (shown as examples):
?1.equals("parent")
?1.equals("parent") || ?1.equals("orphan")
?1.equals(?2)==false && ?1.equals(?3)==false

Allowable numeric expressions include:
Equals to: ==
Greater Than: >
Greater Than or Equals to: >=
Less Than: <
Less Than or Equals To: <=
Not Equal To !=

4.11. Else Operation

The Else operation works only in conjunction with the IF and must be placed directly after an If or and ElseIf. This command must be dragged over from the Operation Palette to be placed directly under the If (or ElseIf) when a black line appears across the screen. Releasing the left mouse button when the black line appears (as shown below) will properly locate the Else operation.



All operations within the Else operation will be executed only if and only if the condition in the corresponding If and all conditions in any corresponding ElseIf operations have been evaluated as false.

4.12. ElseIf Operation

The ElseIf operation works in conjunction with an IF operation, providing an additional comparison which will only occur if the preceding IF (and any preceding ElseIf operations within the same IF) resolves to false. If the above comparisons resolve to false and the ElseIf comparison resolves to true, then the operations within the ElseIf will be executed. Otherwise, control passes to the next ElseIf (if one exists) or an Else (if that exists) or past the end of the If. Configuration of the ElseIf operation occurs in the same way as configuration of the If operation.

Similarly to the Else, care must be taken in placing this operation by dragging the ElseIf operation button to directly below the proper If or ElseIf operation until the black line appears on across the screen.

4.13. While Operation

The While operation is a looping construct that allows a set of code (any operations included within the While operation) to be executed as long as a certain condition evaluates to true. The first time the loop is encountered, and every time the last operation within the loop is completed, the provided Expression will be evaluated. If the Expression evaluates to true, the loop will be executed; if the evaluation results in a return of false, the Transformation will resume processing at the end of the While operation. Configuration of the While operation occurs in the same way as configuration of the If operation.

4.14. User Operation

The User operation allows user-written Java classes to be called from within the Transformer. In all other ways, it works similarly to the standard Copy operation, with the exception being that the value placed in the designated Target is the result of the Java class. The Java class is configured by entering the proper name into the ClassName field, or selected by clicking the Browse button next to the ClassName field.

4.15. JDBC Operation

The JDBC operation enables SQL database queries during the execution of the map. This includes any valid SQL, such as looking up a value with a select statement or inserting a record into a table. The operation can use one of three different modes: Setup, Exec, or Fetch. A typical utilization of JDBC within a map would include a single Setup operation, one or more Exec operations, and possibly one or more Fetch operations.

Setup mode must be executed first; it is responsible for initializing the connection to the database. For each database, a single Setup mode operation will be required. Variable name, Driver name, URL, username, and password are among the configurable operations within Setup mode.

Exec mode is responsible for executing SQL statements against the selected database. After a JDBC setup mode operation has been created (with a variable name), future Exec operations will be able to select the proper database connection from a drop-down list (this populates with Variable names from setup operations). Any valid SQL statement can be used, and multiple source fields are permitted. Question marks in the SQL statement will automatically be replaced with source attributes in the order in which they appear. For example, given the statement:
Select phone from contact where last = ? and first = ? and zip = ?
The three question marks will be replaced with the contents of the first three sources, in order.

Fetch mode is used to access particular data elements within a result set that was returned from an Exec mode operation. The source may contain a string "row x column y" where x and y indicate the row and column to be returned from the result set. For example, the string "row 3 column 4" would return the 4th column from the 3rd row of the result set.

Additionally, the string "count" may be used in the source to return the total number of rows in the result set. This can be used in conjunction with a while loop to work through the entire result set.

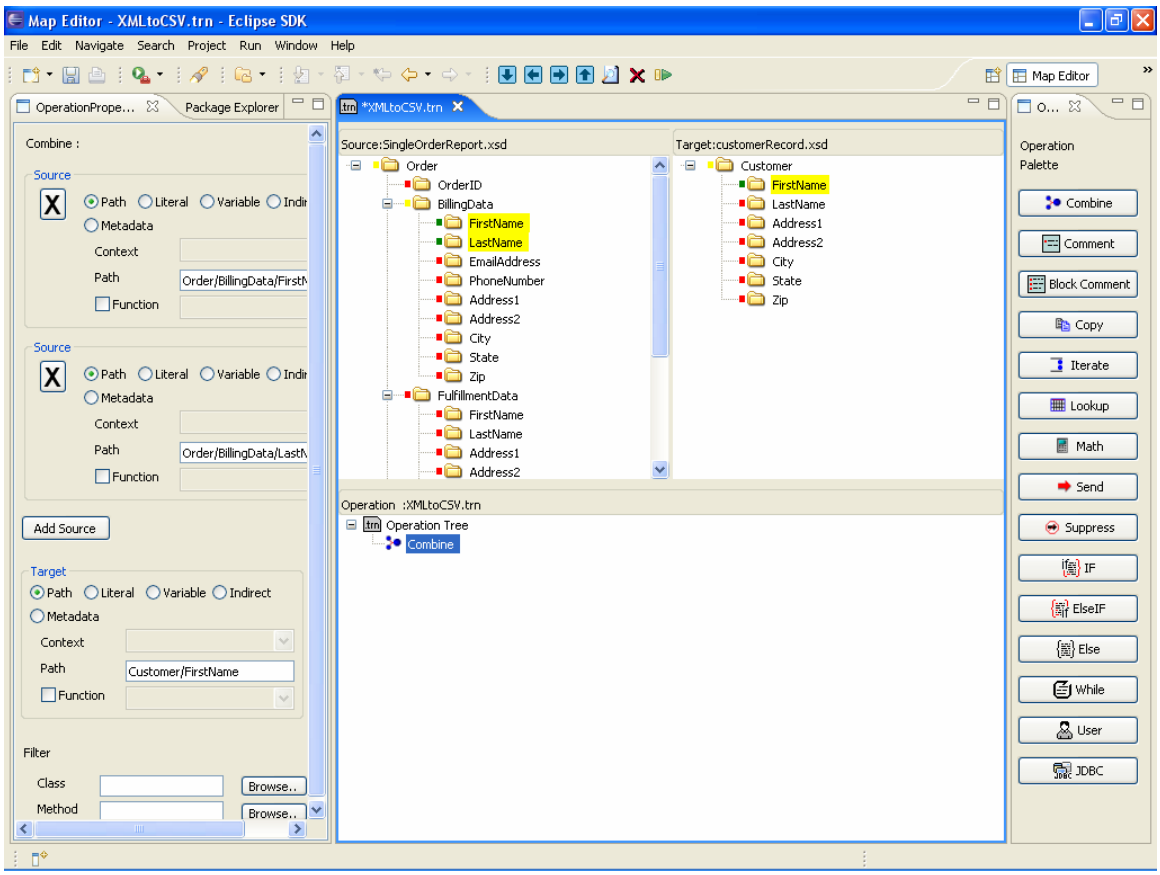
5. Dragging Behaviors

.

5.1. Dragging a Format Node into a Source or Target

A source or target address in the Operation Properties sheet can be populated simply by dragging a node from the source or target tree display. This will always result in an Absolute or Context address. If the node can be reached by a currently active context then the context should be used since it has a specific repetition associated with it.

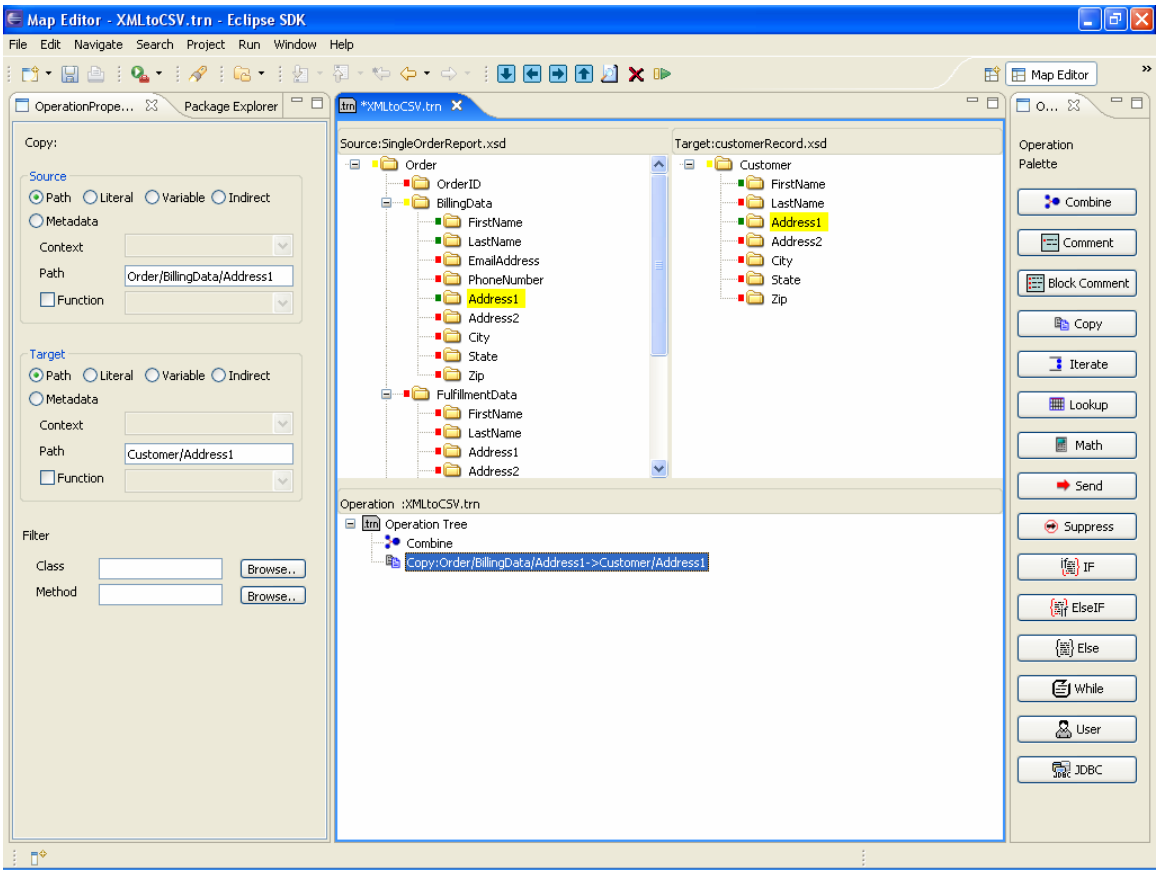
The next screen shot shows the result of a Combine operation by dragging the "FirstName" and "MiddleInitial" nodes from the Source tree into the Source addresses and dragging the "First" node in the Target tree into the Target address in the Operation Properties sheet.



5.2. Dragging a Source Node into a Target Node

Dragging a source node directly into a node in the target tree will automatically add an appropriate copy operation to the end of the current operation list. Copy operations are by far the most common so this will be a big time saver. The addressing modes in the Operation Properties sheet should be absolute or context.

The next screen shot shows the result of dragging the "LastName" node from the Source tree into the "Last" node in the Target tree.

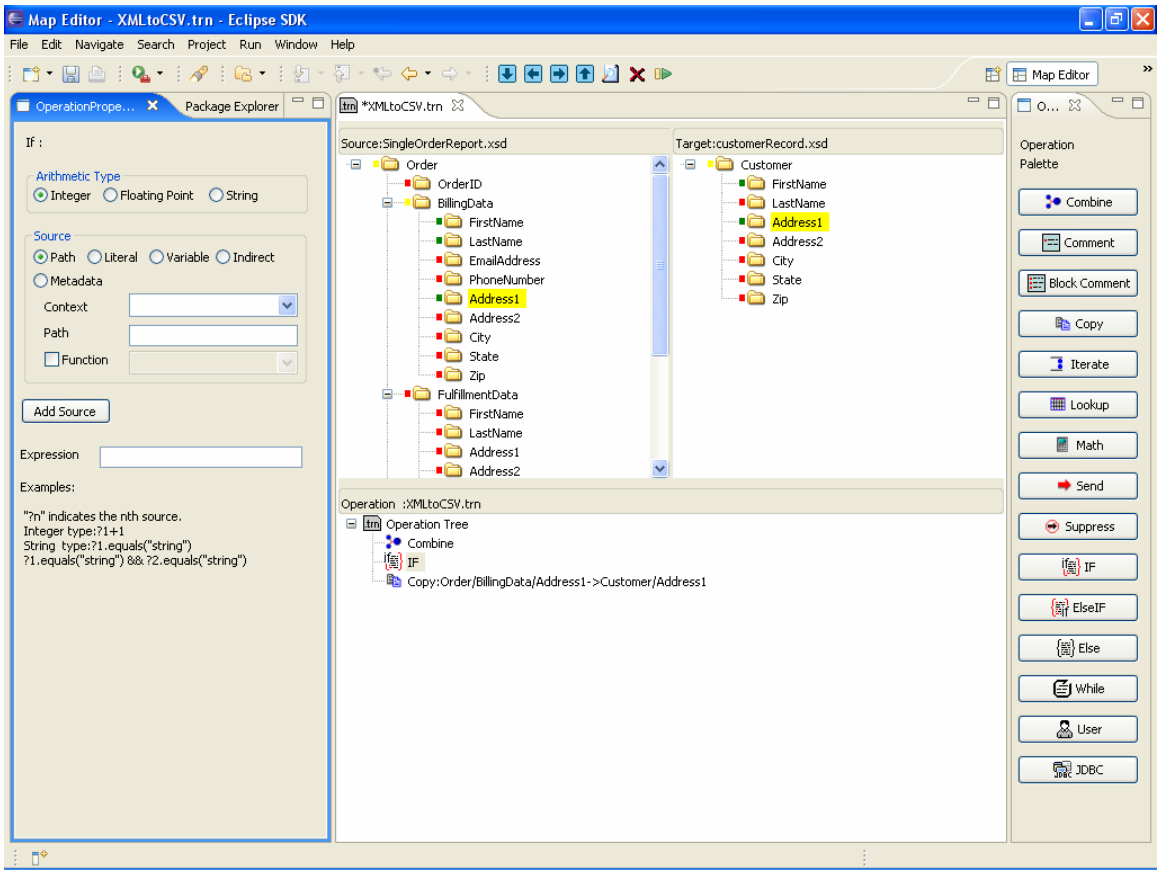


As you can see from the screen, whenever a node is used in a map operation whether it is a source or target node, the small red indicator next to it will change to green. This will give users a great visual overview of which nodes are currently being mapped.

5.3. Dragging an Operation from the Palette

An operation can be dragged from the palette into the desired location in the Operation Tree.

The next screen shot shows the result of dragging an "If" operation from the palette into the Operation Tree between the "Combine" and the "Copy" operations.

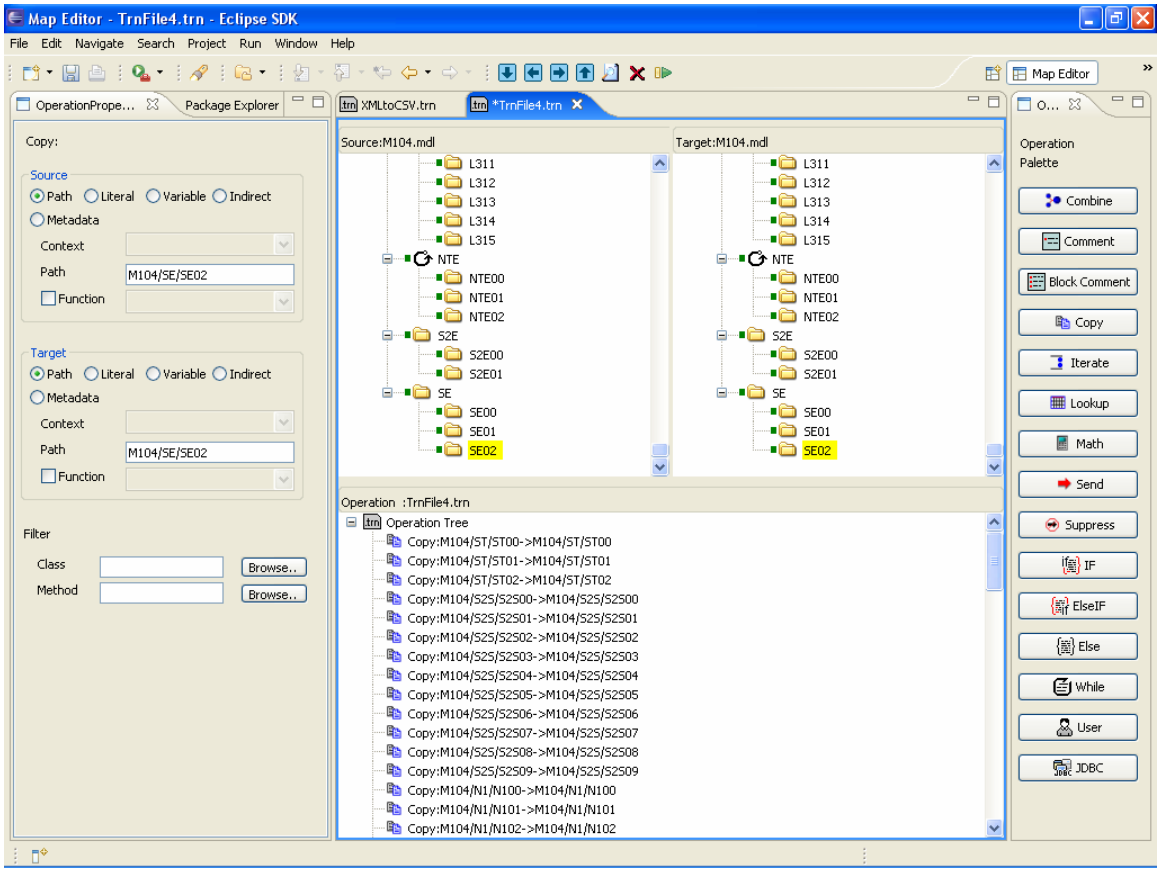


6. Auto Mapping

Auto Mapping is one of the most powerful features in the Map Editor. When you drag a non-leaf node from the Source tree into a non-leaf node in the Target tree, the editor will systematically perform matching between their children. If matching child nodes are found, then Copy operations are automatically added to the Operation Tree. If a child node is repeating, the Copy operations will be placed in an Iterate operation with the correct context defined in both source and target.

This feature is especially useful if the Source and Target definitions are very similar with minor difference. The Auto Mapping feature can be a huge time-saver.

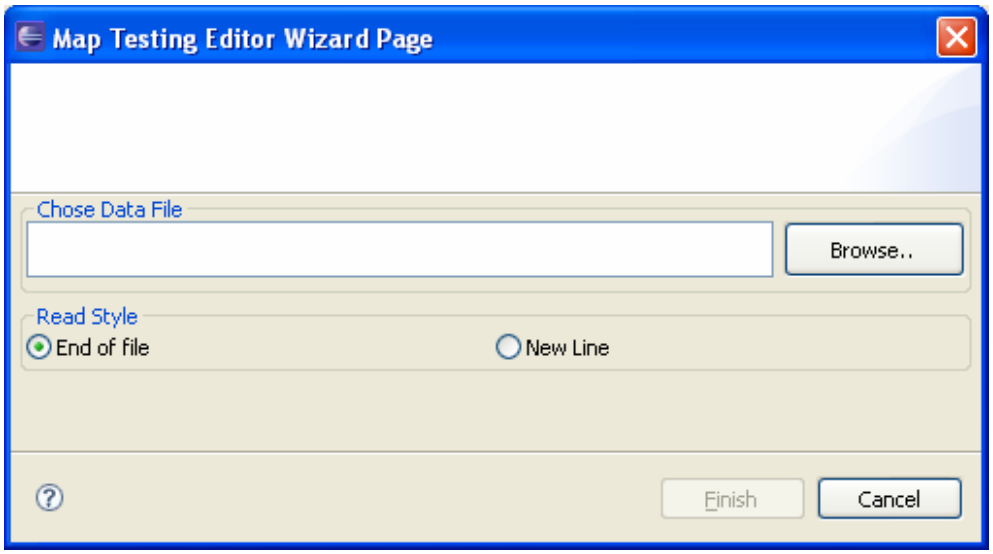
The following screen shot shows the result of Auto Mapping from the message 104 of X12 v00350 into the message 104 of the X12 v003020.



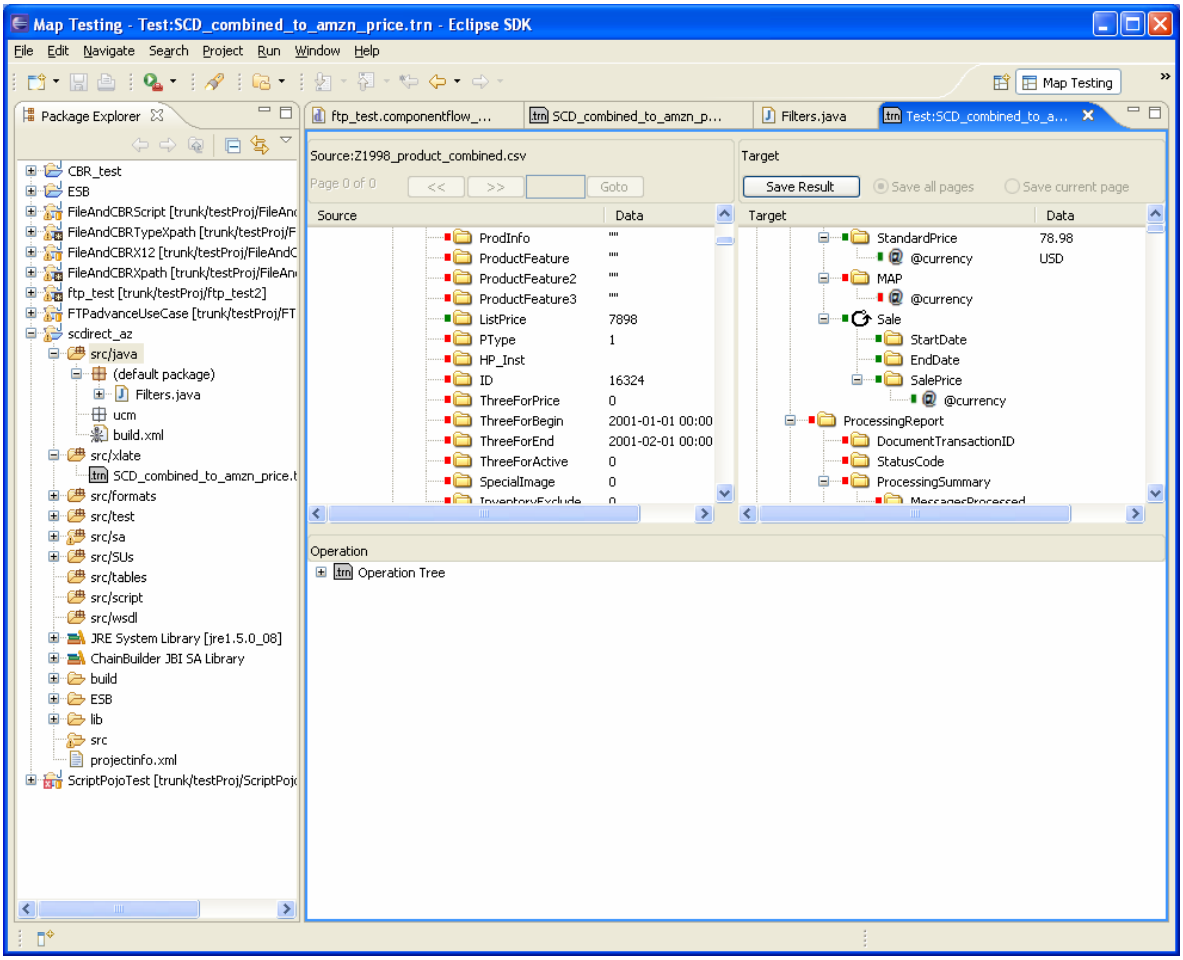
7. Map Tester

After you define a map, you can test it to verify that the result is correct. The test reads a sample source record from a file and displays the parsed result in the source tree based on the source message definition. The result of the map is displayed in the target tree. You can then visually examine if the result is what you expect to ensure that the map is correct before it is used in a Transformation Service Engine component.

You can start the Map Tester by clicking the green "Test" button on the toolbar. You may be prompted to save the map before continuing, click OK to continue. You will be prompted to select a source data file as shown below:



Select a file that contains test data that matches the Source forma, select the appropriate Read Style and click "Finish". The data from the file will be parsed and populated on the Source tree as shown below:



If you would like to save the result of the map, you can simply click the "Save Result" button in the target window which will bring up a dialog box to prompt you to enter the file name and location to save the resulting message.

8. ChainBuilder ESB Community

ChainForge.net is the internet's premier destination to share ChainBuilder and JBI knowledge with your peers.

Join the ChainBuilder ESB Community:
http://www.chainforge.net/community

As a member you can view content and contribute to a Forum:
http://www.chainforge.net/community/forums.html

Read ChainBuilder ESB related Blogs:
http://www.chainforge.net/blogs