BPA
Platform

Technical Overview

Transform Data Tool

codeless
PLATFORMS

Copyright

Head Office:

Codeless Platforms

Suite 1 & 2 Bourne Gate

25 Bourne Valley Road

Poole

BH12 1DY

United Kingdom

Tel: +44 (0) 330 99 88 700

Email: enquiries@codelessplatforms.com

Trademarks

Contents

Introduction

Transform Data is a **Format** tool that enables a new structure to be created from an existing XML or recordset data source. The output generated by this tool is ready for use in a destination system which consumes or requires XML.

You can use this tool where a requirement exists to manipulate data into a customised structure such as, flattening or merging areas of content.

Features

▶ Consume XML or recordset data from existing steps as the input

▶ Create custom XML schemas for output to other tools

▶ Replicate the schema of the input data source

▶ Output to other tools in XML format

Working With Other Tools

The **Transform Data** tool can directly interact with the following tools:

Consuming from Other Tools

The **Transform Data** tool can directly consume output from the following tools:

Icon	Tool Name	Tool Category
	Call Stored Procedure (OLEDB)	Input, Data Connectors, Output, and Execute
	Database Query (ODBC)	Input and Data Connectors
	Database Query (OLEDB)	Input and Data Connectors
	Import Flat File	Input
	Import XML Document	Input
	Retrieve Text Message	Input
	Convert Recordset to XML	Format
	Convert XML to Recordset	Format
	Transform Data	Format
	Call Task	Execute

Icon	Tool Name	Tool Category
	Filter Data	General
	Applications Platform Connector	Data Connectors

Objects Consumed

The following objects, outputted by the above tools, can be directly consumed by the Transform Data tool:

▶ **XML** — XML data from any BPA Platform tool capable of exposing such data (see above)

▶ **Recordset** — Tabular data from any BPA Platform tool capable of exposing such data (see above)

Exposing to Other Tools

The **Transform Data** tool output can be directly consumed by the following tools:

Icon	Tool Name	Tool Category
	Retrieve Text Message	Input
	Format as Flat File	Format
	Run Microsoft Reporting Services	Format
	Save File	Output
	Call Task	Execute
	Applications Platform Connector	Data Connectors

The **Transform Data** tool outputs the following objects which can be directly consumed by the above tools:

▶ **OutputXML** — This object exposes two sub-objects:

  ☐ **XmlString** — This is the XML produced by the **Transform Data** tool at task run-time.

  ☐ **XmlSchema** — This the XSD schema for the XML output.

▶ **ErrorXML** — This object exposes two sub-objects:

  ☐ **XmlString** — This XML document contains any error data reported back from the consuming task step. All errors are created as an `<ERROR>` node with the following child nodes:

    `<CODE />` — The returned error code.

    `<MESSAGE />` — The corresponding error message.

    `<INPUTDATA />` — The input data that was sent which caused the error.

  ☐ **XmlSchema** — This the XSD schema for the XML output.

▶ **CustomXMLOutput** — This is the XML produced if you have requested custom output XML (see Output Data Structure Settings).

▶ **Step Properties** — Standard step properties are available allowing you to use statistical data of the tool

Global Configuration

The **Transform Data** tool does not require any global configuration, and can be used in a task immediately.

Step Configuration

To add a new **Transform Data** step to an existing task, you either:

▶ Click and drag the **Transform Data** icon from the **Task Browser** to the task **Design** area.

▶ From the task's **Design** tab, right-click on empty space and select *New > Format > Transform Data.*

For a detailed description of how to create new tasks, refer to the product help.

About the General Tab

The **General** tab is used to enter the following details for the step:

▶ **Name** — Enter a meaningful name for the step

> **TIP:** If this task instance makes use of two or more **Transform Data** steps, ensure the **Name** used is unique for each individual step.

▶ **Description** — If required, enter a description of this step

Data source can either be:

▶ **No Data Source** — This allows you dynamically create the required XML document in the step itself — you do this in the Main tab

▶ **Use a recordset or XML output from another step** — Enable this parameter if recordset data or an XML document from a previous task step is to be transformed

☐ **Source** — Contains all available inputs from steps previously created in the task

About the Main Tab

The **Main** tab is used to design your required XML structure for output.
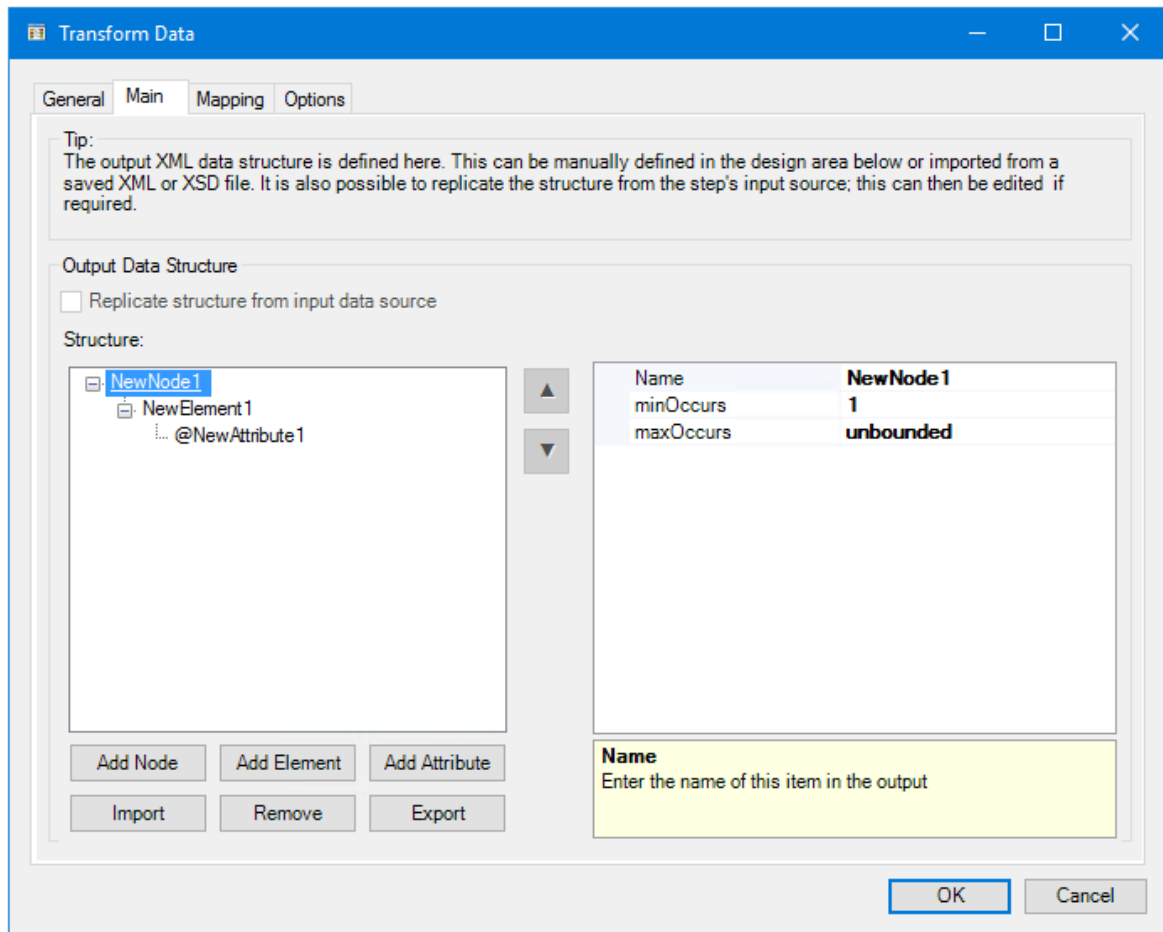
Defining the XML Structure

The structure you create here appears in the **Mapping** tab and is ultimately made available to subsequent steps in the task. You can create this by doing one of the following:

▶ **Replicate the structure from the input data source** — If using an input source from a previous task step, you can replicate the already provided XML structure or use the recordset column names as the XML elements

▶ Import an existing XML structure — If you have an existing XML or XSD file containing the XML structure you would like this task step to use, click **Import** to replicate the structure

▶ Manually create the structure from scratch — Use **Add Node**, **Add Element**, and **Add Attribute** to manually create the required XML structure

If required, click **Export** to use the final XML structure defined here in another **Transform Data** task step, or as a backup.

Defining XML Properties

Irrespective of the method used, additional configuration is required for the node, elements, and attributes:

▶ **Name** — The name of the node, element, or attribute.

▶ **minOccurs**

☐ If a node, this is the minimum number of times this node can appear in the final output.

☐ If an element, this is the minimum number of times this element can appear in its parent node.

☐ If an attribute, this is the minimum number of times this attribute can appear it its parent element.

▶ **maxOccurs**

☐ If a node, this is the maximum number of times this node can appear in the final output.

☐ If an element, this is the maximum number of times this element can appear in its parent node.

☐ If an attribute, this is the maximum number of times this attribute can appear it its parent element.
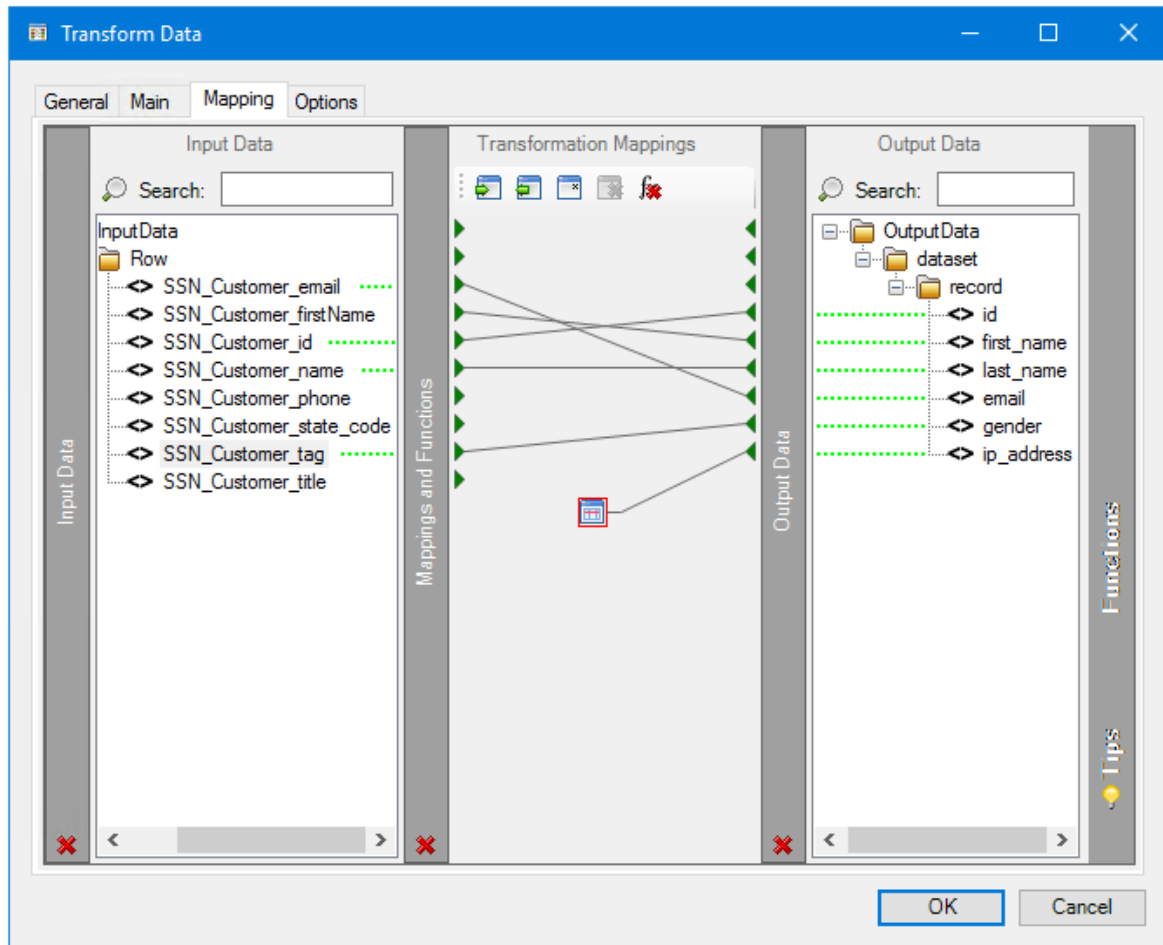
Use **Unbounded** to indicate an unlimited number of appearances.

▶ **Type** — Only available for attributes. This is the data type of the attribute; standard data types are available.

About the Mapping tab

The **Mapping** tab uses the Data Transformation Layer (DTL) to define links between the input XML or recordset data and the new XML output structure generated at runtime.

The left-hand **Input Data** pane reflects the current structure of the data source — if you see "No schema has been loaded" you can create the input source in the **Transformation Mappings** pane by assigning variables to a **Fixed / Dynamic** function (refer to the product help). The **Output Data** pane on the right reflects the designed XML structure from the [Main tab](#). To create a mapping, drag an element from the left pane on to an element in the right pane.

From here you can:

▶ Automatically map where input and output parameter names match

▶ Create mappings from a set of transform functions to change the data between input and output

▶ Use nested looping to support hierarchical data structures

▶ Import and export mappings so that they can be reused in other steps

For a detailed description of how to map data in this tab, refer to the product help.

About the Function Types

A number of functions are available with the Data Transformation Layer (DTL):
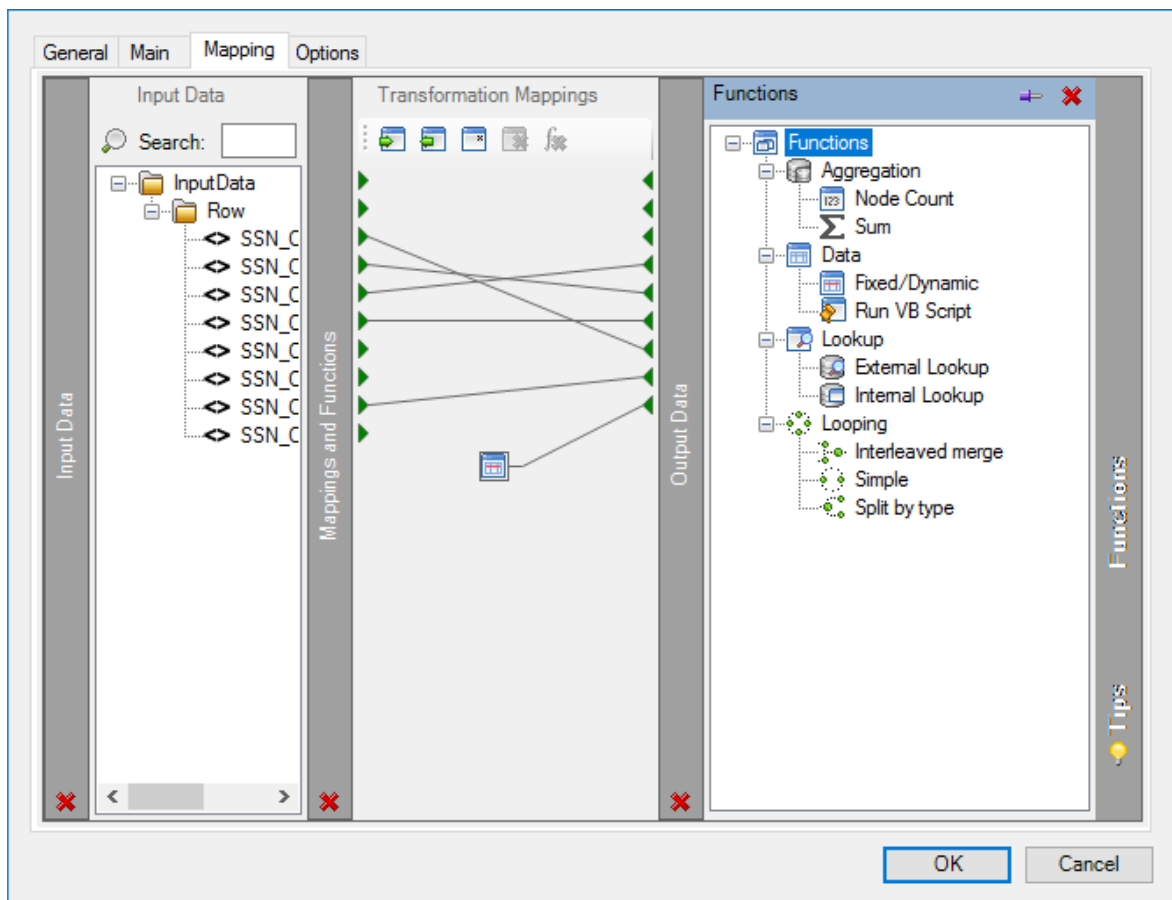
Aggregation Functions

Aggregation functions define operations for specific nodes.

The Node Count Function

This function counts the number of occurrences of a node in the input recordset or XML document. The result is then passed to the mapped output node.

The Sum Function

This function calculates the total of the values from all iterations of an element in the input recordset or XML document. The sum of the values is used as the new value to an output element.

Data Functions

Data functions perform operations on input data to generate new output data.

The Fixed / Dynamic Function

This function passes a static or dynamic value to the output XML schema. Use a variable or recordset column to generate the dynamic data.

The Run VBScript Function

Use this function to perform VBScript operations to process input data, generate output data, or both.

Lookup Functions

Lookup functions are used to find values in a nominated source, by a key.

The External Lookup Function

Use this function to lookup values from an external database. This uses existing **Database Query (ODBC)** or **Database Query (OLEDB)** global connections.

The Internal Lookup Function

Use this function to find an alternative value for input data from a predefined lookup table.

Looping Functions

Looping functions loop through the input recordset or XML document to perform functions on all iterations of a node.

The Interleaved Merge Function

This function loops through the input recordset or XML document and merges data from specified elements into a single occurrence for the mapped output data.

The Simple Loop Function

This function loops through the input recordset or XML document and creates an output data node for every iteration of an input node it finds.

The Split by Type Function

This function is the opposite to the Interleaved Merge function in that it takes data from a single input node and splits it into two or more output data nodes.

About the Options Tab

The **Options** tab allows you to define how errors in this step are handled at task runtime.
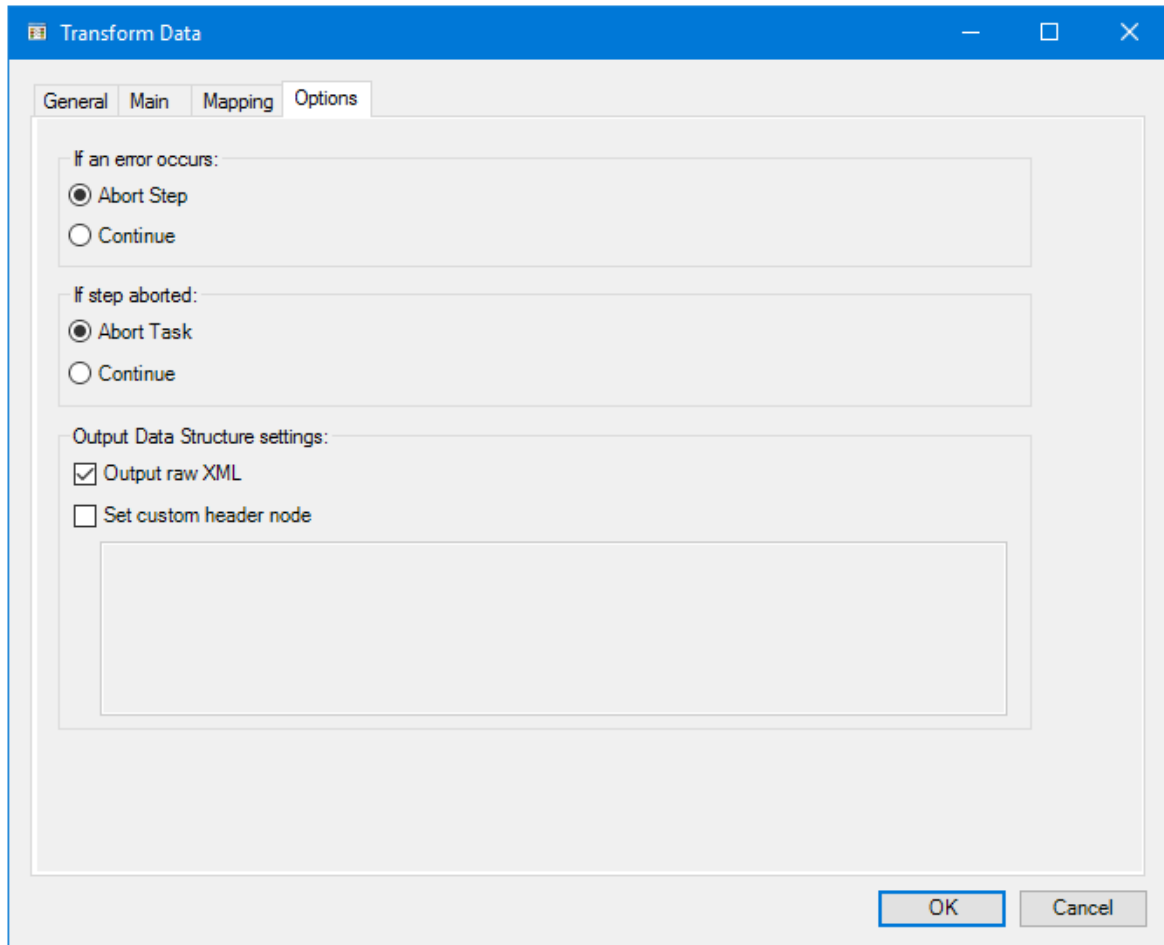
If an error occurs, you can decide whether the step should **Continue** processing, or terminate the step immediately (**Abort Step**).

If the step is aborted, you can choose to **Continue** processing onto the next step in the task, or terminate the whole task immediately (**Abort Task**). By allowing the task to continue, you can use the error XML received back in a **Save File** step for investigation purposes, for example.

By default, the XML document is outputted with `<OutputData>` as the header node with a standard namespace declared. If required, you can choose to output the XML with no namespace declaration or with a custom header node. This is particularly useful if you want to directly use the XML in an external system.

To change the **Output Data Structure settings**, enable **Output raw XML** — this setting alone removes the namespace declaration. If a different header node is required, enable **Set custom header node** and specify it below (to use no header node, leave this blank). In either circumstance, use the **CustomXMLOutput** property rather than the **OutputXML** property when saving the output to file — see CustomXMLOutput Property.

Want to learn more?

Discover how Codeless Platforms can help your business by improving performance, boosting efficiency and cutting costs.

+44 (0) 330 99 88 700

enquiries@codelessplatforms.com

www.codelessplatforms.com

codeless PLATFORMS