BPA
Platform

Technical Overview

Web Service Connector Tool v1.2

codeless
PLATFORMS

Copyright

The copyright in this document is owned by Orbis Software Ltd T/A Codeless Platforms 2021. All rights reserved.

This publication may not, in whole or part, be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language in any form or by any means without the prior written consent of Orbis Software Ltd T/A Codeless Platforms.

Head Office:

Codeless Platforms

Suite 2 Bourne Gate

25 Bourne Valley Road

Poole

BH12 1DY

United Kingdom

Tel: +44 (0) 330 99 88 700

Email: *enquiries@codelessplatforms.com*

Trademarks

Orbis Software Ltd T/A Codeless Platforms owns the registered trademark "TaskCentre®".

All other Trademarks used are acknowledged as the property of their respective owners.

The information provided in this publication may contain errors, omissions, or typographical errors or may be out of date. Orbis Software Ltd T/A Codeless Platforms may change, delete, or update any published information at any time and without prior notice. The information published in this document is provided for informational purposes only and is not binding on Orbis Software Ltd T/A Codeless Platforms in any way except to the extent that it is specifically indicated to be so.

Contents

Introduction ... ***1***

System Requirements ... *1*

Technical Summary ... ***2***

Consuming from Other Tools ... *2*

Objects Consumed ... *3*

Exposing XML to Other Tools ... *3*

Objects Exposed ... *4*

REST Services — Output Data ... *5*

REST Services — Error Data ... *5*

Where Can the XML Output be Used? ... *6*

Error Handling ... *7*

TLS 1.2 Support ... *7*

Connecting to a Web Service ... ***8***

Creating a New Web Service Configuration ... *9*

Using WSDL Web Services ... *10*

Defining REST Web Services ... *11*

Using Extended Logging ... *14*

Additional Web Service Settings ... *16*

Step Configuration ... ***17***

About the General Tab ... *17*

About the Web Service Tab ... *18*

Refreshing the Operations' Schemas ... *18*

Changing Web Services when Mappings Exist ... *19*

About the Mapping Tab ... *19*

Creating Mappings ... *20*

Using Functions ... *20*

Additional Functionality ... *21*

About the Options Tab ... *22*

Introduction

The **Web Service Connector** tool allows BPA Platform to integrate with a wide number of commercial web services, making it possible to automate the passing of data requests and retrieval to and from the website that would normally be carried out manually. It enables communication with a wide range of web services — both Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) protocols. It is capable of working with XML and JavaScript Object Notation (JSON) formats, allowing you to post to and retrieve data from your business application.

If required, you can use a Web Services Description Language (WSDL) file to describe the web service used. Supported operations include `GET`, `POST`, `PUT`, and `DELETE`. You can control any header or authentication requirements, can even join together multiple operations to form a single web call. The response received back from a web call can be directly passed to other steps in a task, which can be useful for data integration type scenarios.

The **Web Service Connector** can consume a recordset or an XML document. You can also map exposed web service parameters to the supplied data source.

At task run-time, you can use the BPA Platform Event Log to view the contents of the request and response payloads received with each web call.

System Requirements

The **Web Service Connector** tool requires:

- BPA Platform 2020 Update 1 (Build 6466) or higher
- Microsoft .NET framework version 4.7.2 or higher

Technical Summary

The other tools that a **Web Service Connector** step can interact directly with are listed below:

Consuming from Other Tools

The **Web Service Connector** tool can consume objects from the following BPA Platform tools:

Icon	Tool Name	Tool Category
	Database Query (ODBC)	Input and Data Connectors
	Database Query (OLEDB)	Input and Data Connectors
	Database Query (HANA)	Input and Data Connectors
	Import Flat File	Input
	Import XML Document	Input
	Convert Recordset to XML	Format
	Convert XML to Recordset	Format
	Create Workflow Job	Format
	Format as Flat File	Format
	Format as HTML	Format
	Format as HTML Pro	Format
	Format as Text	Format
	Run Crystal Report	Format

Icon	Tool Name	Tool Category
	Run Microsoft Reporting Services	Format
	Run Microsoft Word (Merge)	Format
	Transform Data	Format
	Call Task	Execute
	Filter Data	General
	Applications Platform Connector	Data Connectors
	Web Service Connector	Data Connectors

In addition, the **Web Service Connector** can consume the output from other Data Connector tools that provide connectivity to an external application, such as an eCommerce, ERP or CRM system.

Objects Consumed

The **Web Service Connector** consumes the following objects exposed by other steps:

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

Exposing XML to Other Tools

The **Web Service Connector** tool exposes objects, properties, and XML documents that can be consumed by the following BPA Platform tools:

Icon	Tool Name	Tool Category
	Convert XML to Recordset	Format
	Run Microsoft Reporting Services	Format
	Transform Data	Format

Icon	Tool Name	Tool Category
	Save File	Output
	Web Service Connector	Data Connectors

Objects Exposed

The **Web Service Connector** tool is able to output an XML document for each `GET`, `POST`, `PUT`, and `DELETE` operation, depending whether the execution of the web call is successful at runtime. This XML can be used by subsequent steps in the task that can directly consume XML.

For REST web services, if an operation is retrieving XML or JSON, you must specify that the content is to be made available as an output to other steps in a task. For SOAP services, this is usually set within the WSDL.

> **NOTE:** The content of a web services' call response depends on the individual service, and some may not output data for consumption from operation types. In some cases, only a `200 OK` response is received. For more information, refer to the respective web service's API documentation.

The **Web Service Connector** tool outputs the following objects which can be consumed by other tools:

- **InputData** — This document contains the input XML received by the **Web Service Connector** tool. It is only available if a task step has been selected as the **Data Source** (see *About the General Tab*).
- **OutputData** — The **OutputData** object contains two sub-objects:
  - **XmlString** — This is the XML document produced by the tool, containing data returned from the external application's operation. Also included are the key fields for the mapped elements affected by the used operation — for example, if using a `POST` operation, the key fields for the top-level object instances that are created are returned — and a `SupplementaryReference` field for task auditing purposes.

    The mapped fields in the **Mapping** tab (see *About the Mapping Tab*) define the structure of this XML document.
  - **XmlSchema** — This contains the output schema in XSD format.
- **ErrorData** — The **ErrorData** object also contains two sub-objects:
  - **XmlString** — This contains any error data reported by the external application

    `<Error>` — All errors are created as an `<Error>` node, with the following sub-nodes:

    `<Object />` — The name of the requested object

    `<CODE />` — The error code returned by the external application, if applicable

    `<MESSAGE />` — The corresponding error message

    `<EXTENDEDINFO />` — A string containing additional information about the error

`<INPUTDATA />` — The header input data (excluding child nodes) mapped for the object, plus all data contained in `SupplementaryReference`

`</Error>`

For more information about the errors received, see *Error Handling*.

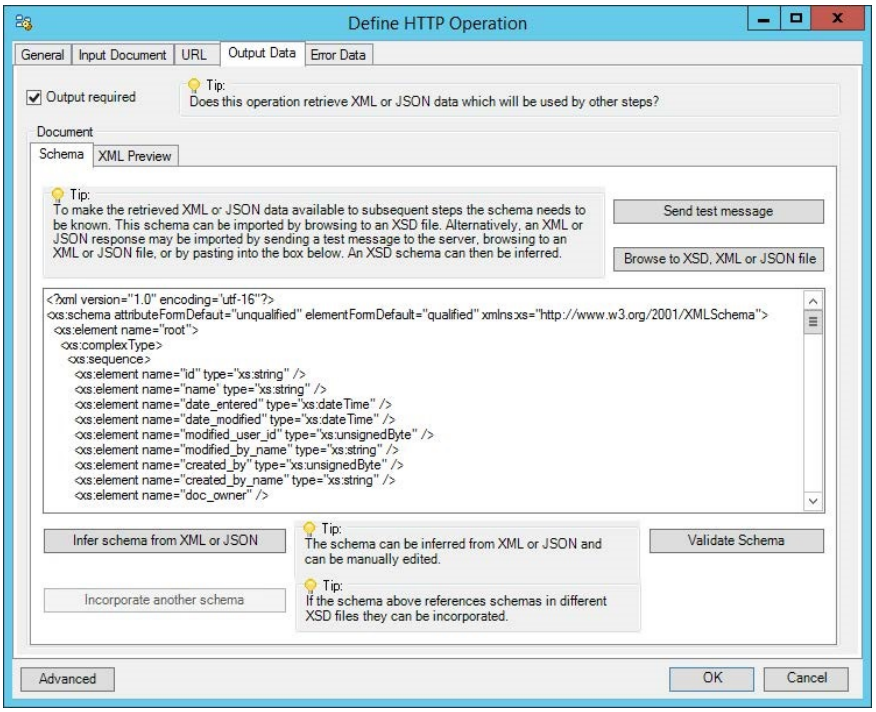
- **XmlSchema** — This contains the output schema in XSD format.

- **Step Properties** — Standard step properties are also available allowing you to use statistical data of the **Web Service Connector** step.
- **HTTPResponseHeaders** — The web service response headers if the option is selected in the **Response Headers - Output** tab of the **Advanced HTTP Message Settings** dialog.
- **Connection Constants** — Any constants created for the web service configuration, with fixed or dynamic values.

REST Services — Output Data

By default, the **Web Service Connector**, when in REST-mode, doesn't output data. To generate output, ensure you enable **Output Required** in the ***HTTP Web Service Configuration interface > Operations tab > Add > New HTTP Operation interface > any operation > Define HTTP Operation interface > Output Data tab***.

TIP: You can supply example XML or JSON here for the **Web Service Connector** to infer the schema from.



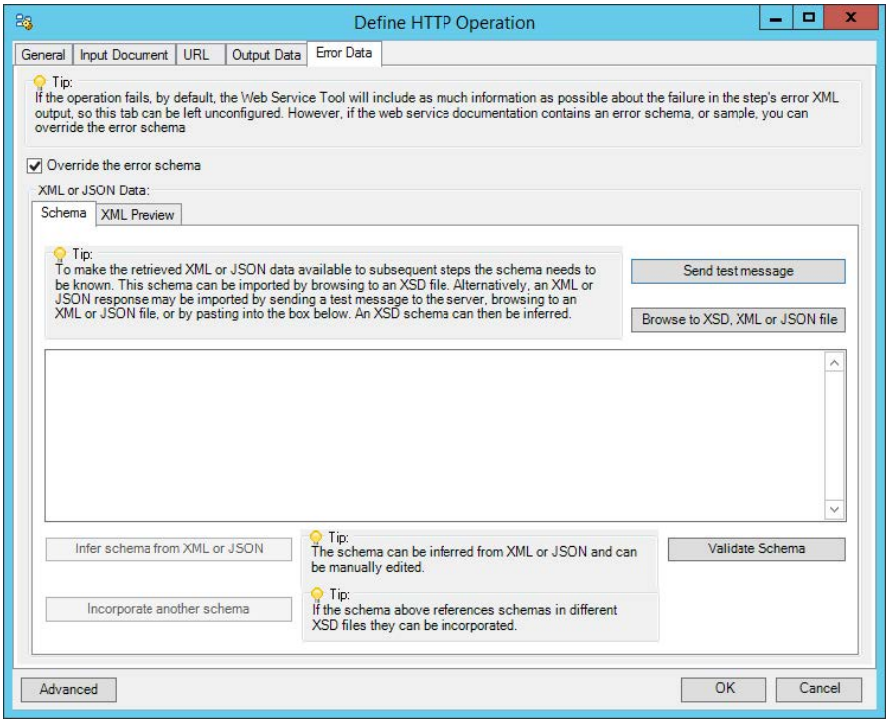
REST Services — Error Data

By default, if an operation fails, the **Web Service Connector** automatically tries to provide as much information as possible within its own error XML. This includes the input data along with any error details returned by the web

service.

However, if the web service provides an alternative error schema, and one that can provide further details for a failed web call, then enable **Override the error schema** in the ***HTTP Web Service Configuration interface > Operations tab > Add > New HTTP Operation interface > any operation > Define HTTP Operation interface > Error Data tab***, and supply the new error schema.

> **TIP:** You can supply example XML or JSON here for the **Web Service Connector** to infer the schema from.



Where Can the XML Output be Used?

The incoming XML is translated into the XML format for the object and operation selected in the configuration. The data for the linked fields is brought across into the output XML — only those fields that are linked are brought across. The XML is passed to the **Connector**, which then:

- Processes the data
- Performs the operation requested
- Receives an XML document containing the response

Both the `OutputData` and `ErrorData` documents can be directly used by succeeding task steps that can consume XML data, as part of an application integration or synchronisation process. To use the documents in a non-XML consuming tool, use a **Convert XML to Recordset** step first to create a recordset copy of the XML data.

The XML documents are also available as consumable objects from the Task Browser (`XmlString`). When used in a task step, such as **Format as Text** or **Save File**, this exposes the actual XML string.

Error Handling

Errors are written to the BPA Platform Event Log (***Tasks*** toolbar **> *Event Log*** ). You define how errors are handled in the **Options** tab of the tool (see *About the Options Tab*).

Reasons for the errors could include:

- Web service connection errors
- User privilege errors
- Errors, messages, and warnings from the API
- Any reported task runtime errors, such as, loss of connection

TLS 1.2 Support

The **Web Service Connector** tool supports TLS 1.2 where available. The **Web Service Connector** is able to negotiate to find a common protocol to use — it is recommended you leave this enabled if your web service supports negotiation. However, if your web service is unable to support TLS 1.2 and negotiation, you should turn this parameter off — the **Web Service Connector** defaults to whichever protocol (TLS 1.1 or lower) is advised by the web service during the initial handshake.

Connecting to a Web Service

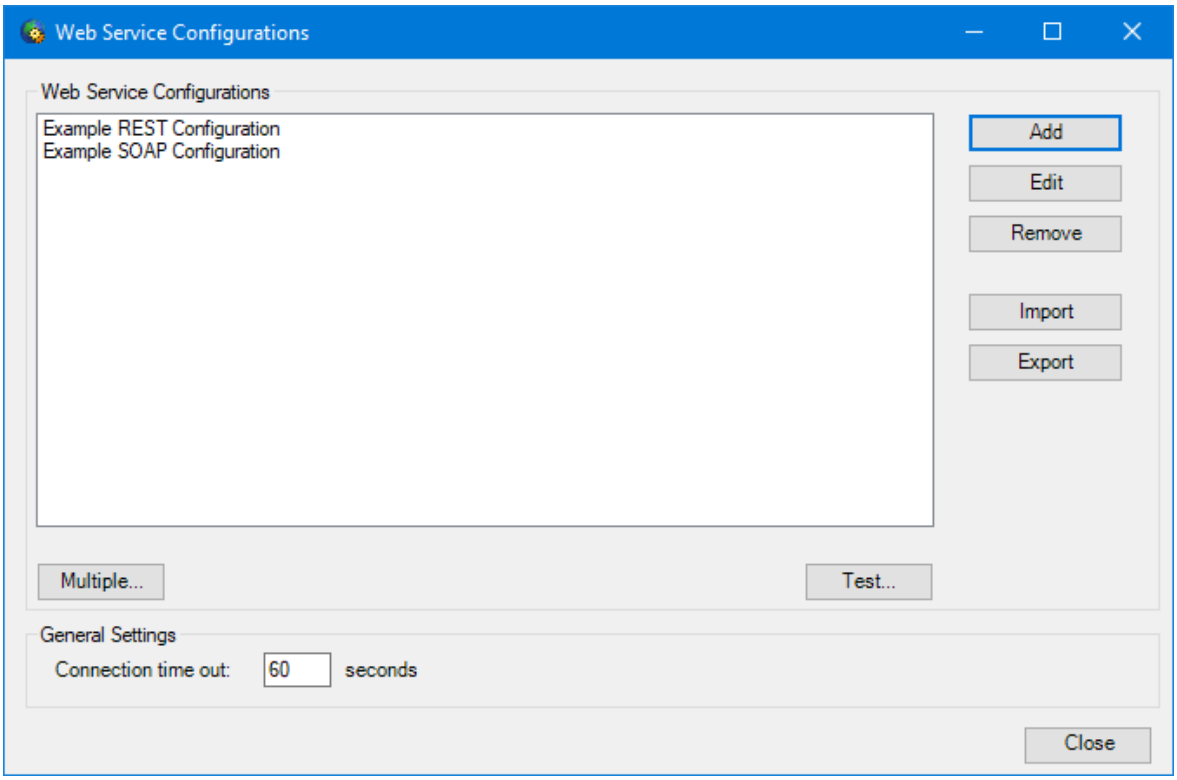
The global configuration for this tool allows you to setup a connection to a SOAP or REST web service.

For services with a WSDL or schema file that describes the service, this file can be used to automatically determine the available operations and their schemas. Alternatively, where a web service is HTTP- or REST-based, it can be configured manually.

Each connection is known as a **Web Service Configuration**.

> **NOTE:** You may be restricted as to the number of web service connections you can create. For more information, refer to the BPA Platform product help or consult your Codeless Platforms account manager or partner.

You open this interface from the resources tree — expand ***System*** > ***Tools*** > ***Data Connectors*** and double-click ***Web Service Connector*** in the items list.



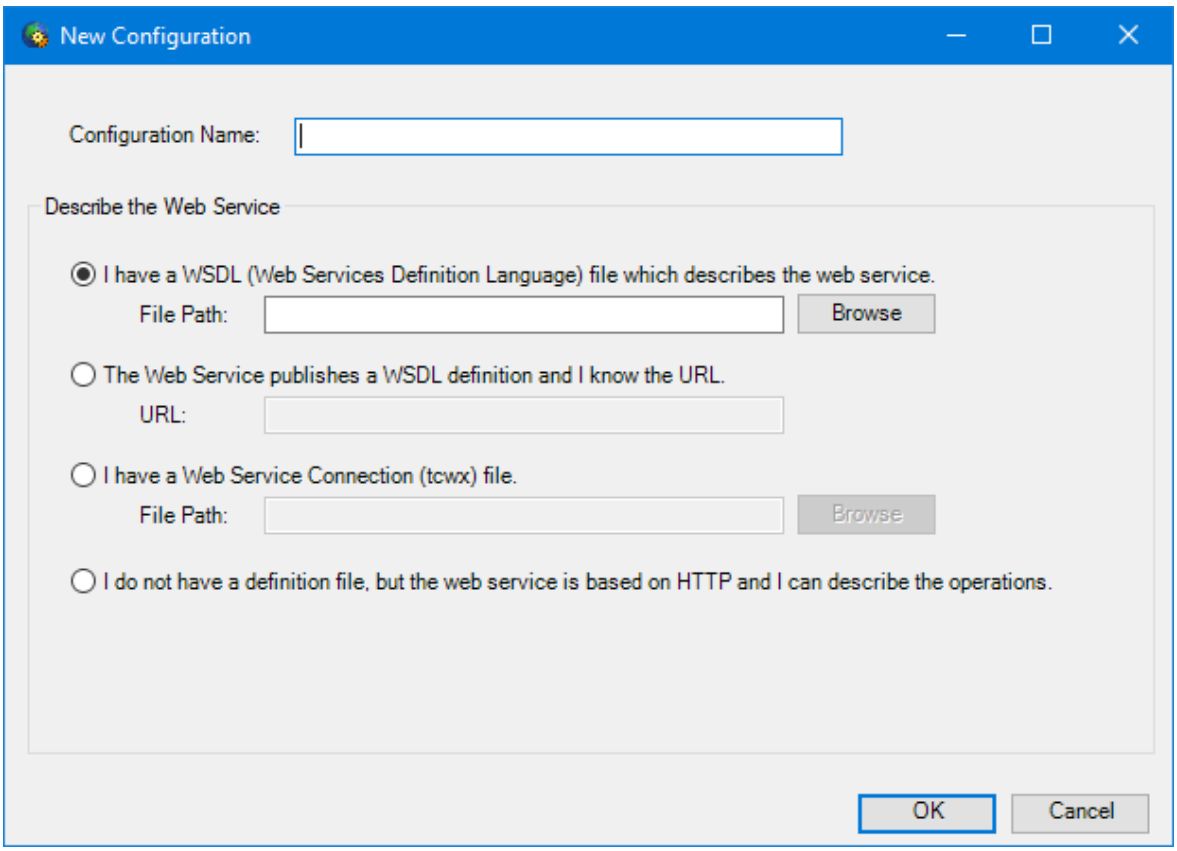
If you have exported your web service configuration from another **Web Service Connector** installation (**Export** button), use **Import** to import the configuration (`.TCWX`).

Otherwise, use **Add** to create a new web service configuration.

When the configuration or import is complete, you are returned to this dialog. Use **Test** to ensure the **Web Service Connector** can communicate successfully with the selected web service.

Creating a New Web Service Configuration

You must describe the web service the **Web Service Connector** uses to communicate with the external application.



Provide a unique **Configuration Name** for this web service. This is the name used when selecting the web service for a task — see *About the Web Service Tab*.

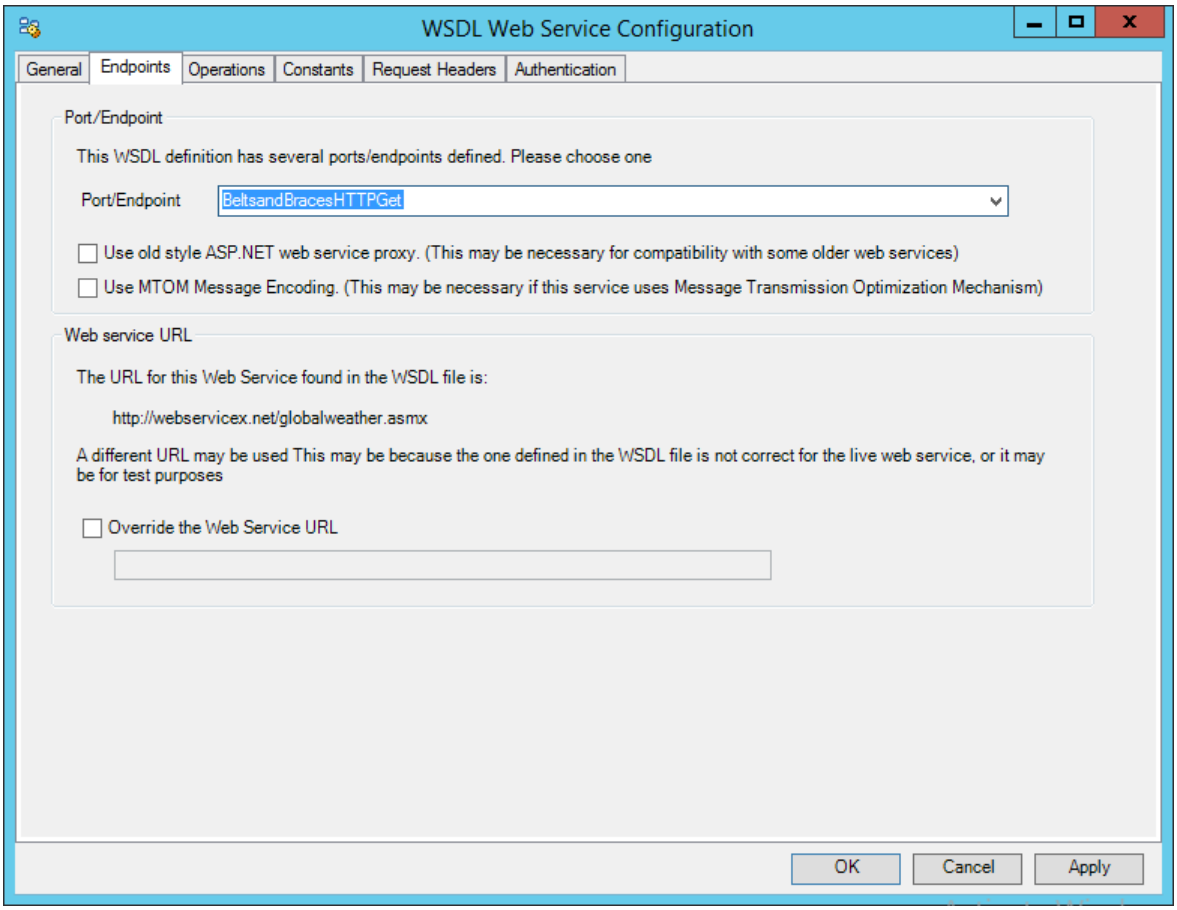
Four options are available as follows:

- **I have a WSDL (Web Services Definition Language) file which describes the web service** — Used for SOAP operations, the WSDL determines the operations available for this web service. You must upload the WSDL to BPA Platform first so it can interrogate it; details returned can be amended as required, such as adding authentication credentials. The WSDL is checked for errors before proceeding.
- **The Web Service publishes a WSDL definition and I know the URL** — Used for SOAP operations, the WSDL determines the operations available for this web service. For this scenario, the WSDL is hosted on a web server that is reachable by BPA Platform. Again, the WSDL is interrogated, and details returned can be amended as required, such as adding authentication credentials. The URL and WSDL are checked for errors before proceeding.
- **I have a Web Service Configuration (tcwx) file** — If you have exported your web service configuration from another **Web Service Connector** installation, use this option to import the configuration (`.TCWX`).
- **I do not have a definition file, but the web service is based on HTTP and I can describe the operations** — Used for REST operations, you must enter the web service operations manually — refer to the web service's API documentation.

Using WSDL Web Services

For many services that provide a WSDL, no further activity, beyond pointing at the file, is necessary. The WSDL file is interrogated by BPA Platform to determine as many details about the service as possible, such as the operations available and their parameters.

The details returned from the WSDL file interrogation can be amended and augmented as required, such as adding authentication details.

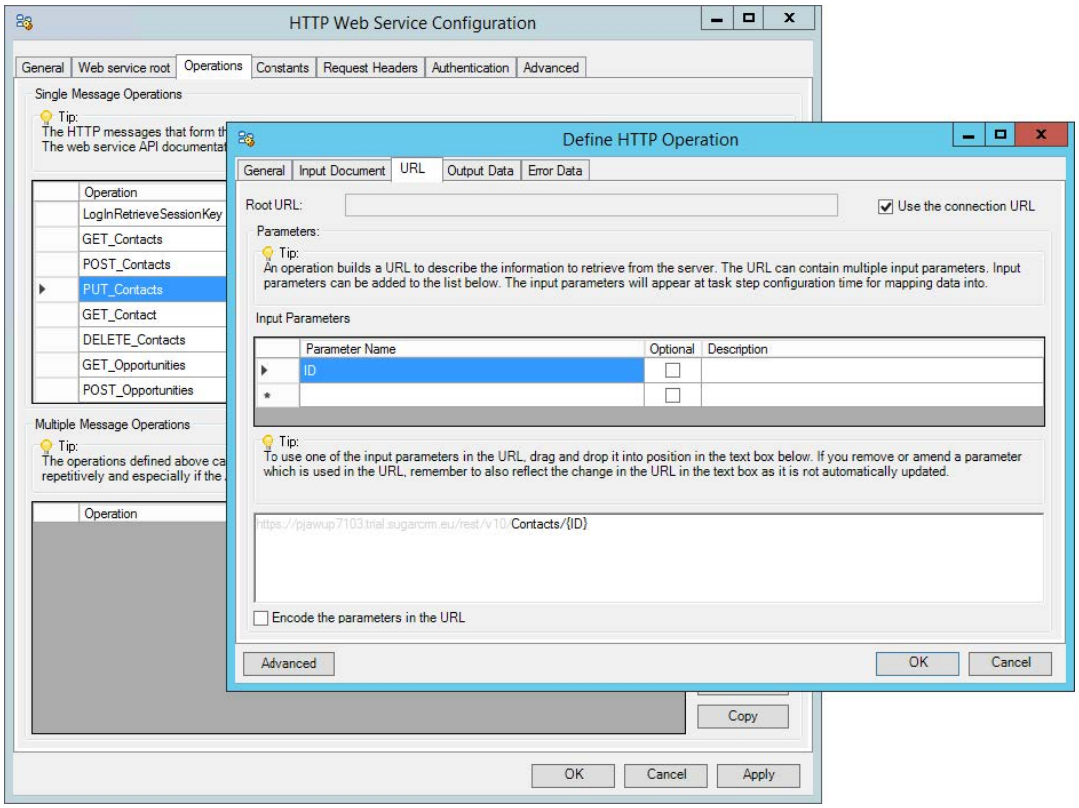


Defining REST Web Services

An HTTP web service (REST) typically communicates using common HTTP methods, such as, `GET`, `PUT`, `POST`, and `DELETE`, to read from and write to the web service.

When creating a REST-based service, you must describe the operations first, that is, supply the web service location and the criteria required by each operation before it can successfully be performed:

- The root URL of the web service
- The resource URL of the web service, that is, the area of the API that is interacted with, for example, "Orders", "Inventory" or "Customers".
- The supported HTTP methods, such as, `GET`, `PUT`, `POST`, or `DELETE`
- The supported parameters
- The supported request headers
- Any authentication requirements to identify yourself to the web service



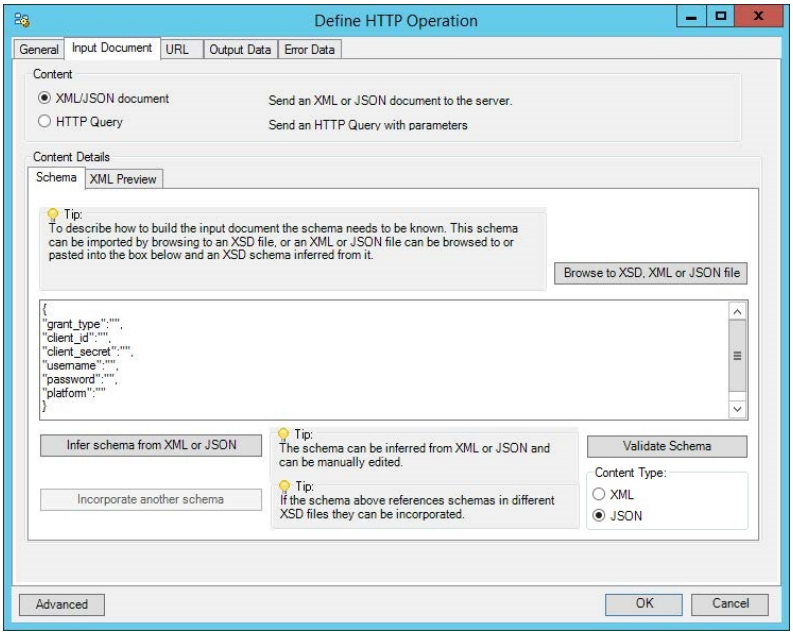
If required and allowed by the web service, you can add parameters to the URL. For operations such as `CREATE`, the input structure for the web call can be defined. These parameters appear when configuring the task step as input parameters — the advantage is that the task designer does need to see or be aware of the URL structure.

XML and JSON Support

Some web services support XML, JSON, or both.

When creating an operation, you can infer the schemas required for communicating with the web service from the supplied XML or JSON. At task step level, an XML-type structure is presented for mapping the request. At runtime,

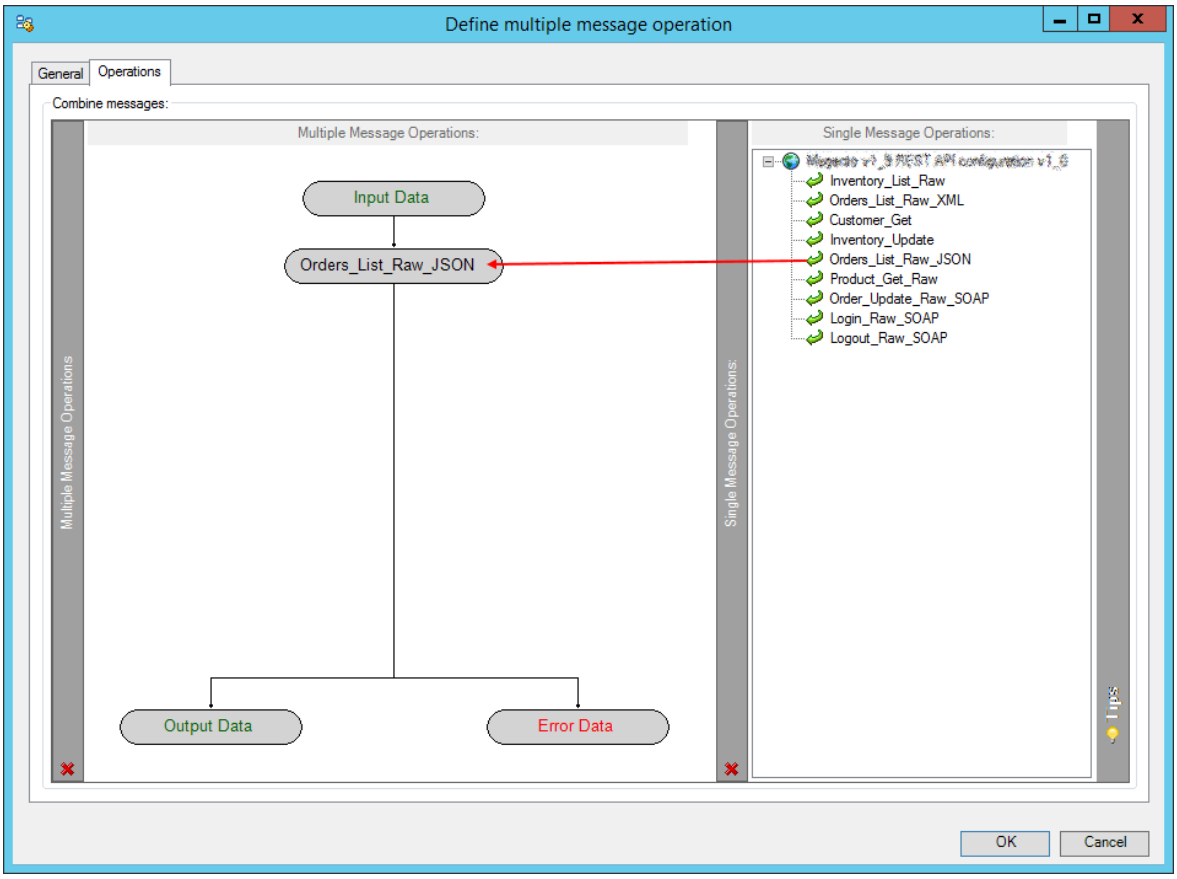
the appropriate XML or JSON is submitted. XML or JSON responses from the web service are made available to subsequent tool steps which can consume XML as standard — JSON responses are converted to XML first.



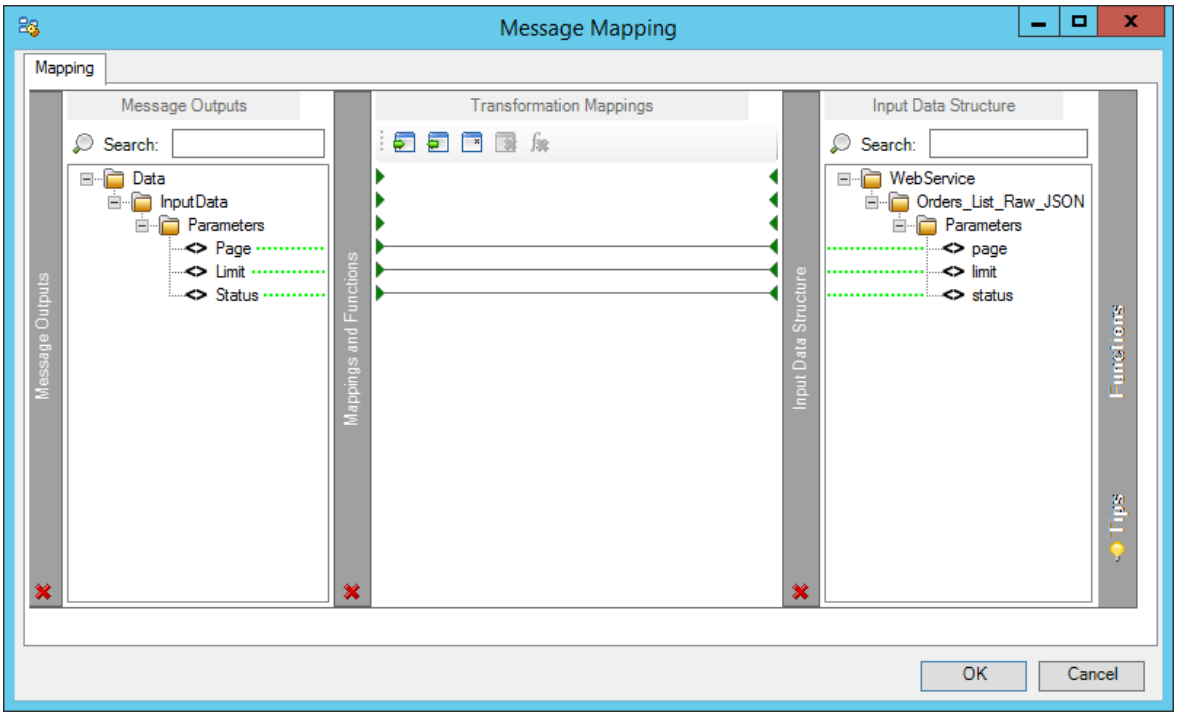
Combining Operations

If required, you can combine multiple operations into a single, user-defined web call. For example, a web service may require you to authenticate before any other operation can be performed, then log out once complete. You would create a multiple message operation that consists of three single operations linked together one after the other — one to perform the login, one to perform the data retrieval and one to logout. Alternatively, if for the web service a `POST` is always performed after a `GET`, you can combine the two operations allowing the task designer to only see a single operation and therefore map just one set of data.

Typically, the schema presented to the task designer contains all the necessary parameters for the web service. If required, you can restructure the schema to contain only the relevant parameters, with appropriate naming and ordering.



Double-clicking on a multiple message operation (left-hand panel), opens the **Message Mapping** window allowing you to map the user-defined schema and any results from previous operations into the input schema of the operation.



Using Extended Logging

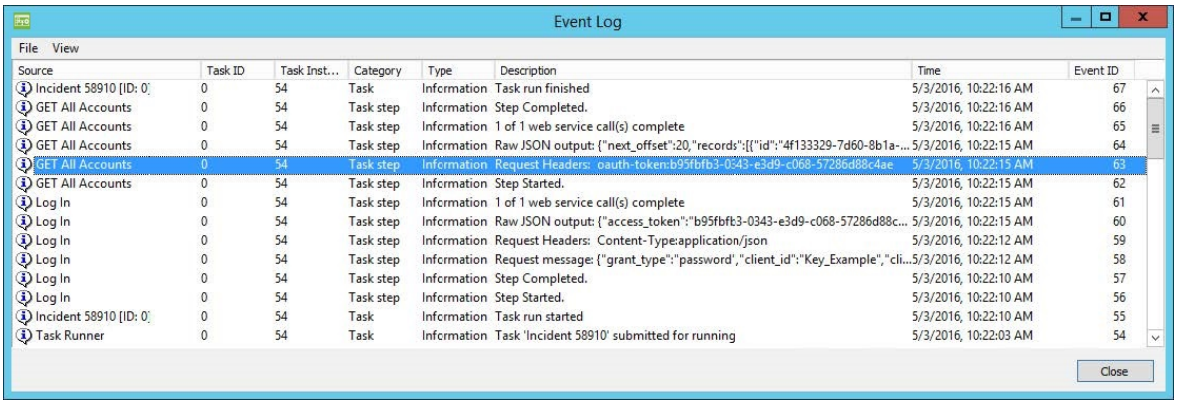
Selecting this option exposes the full XML parsed between the **Web Service Connector** and the external web application.

Without extended logging, the Event Log only contains start and end of transaction messages, plus any error messages encountered at runtime.

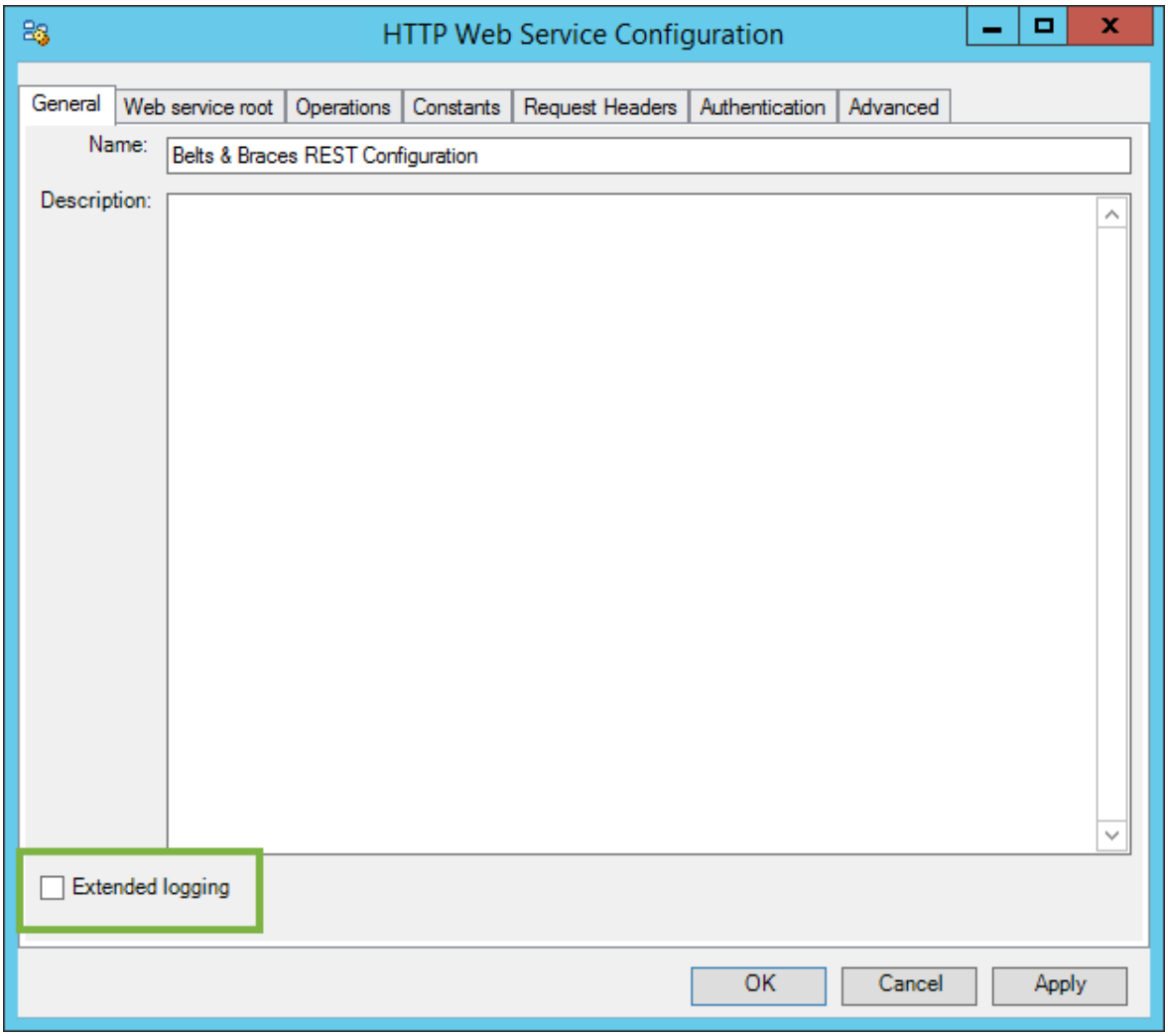
You can view the extended log in the BPA Platform Event Log (***Tasks*** toolbar **> *Event Log*** ).

> **TIP:** The Event Log database table may grow substantially large as extended logging adds additional rows for each call made to and from the **Connector**. To manage this, adjust the maintenance routine for the Event Log — refer to the BPA Platform product help.

> **NOTE:** By default, each Event Log entry is a maximum of 16,000 characters. As most web service requests and responses exceed this number of characters, an additional log file is created in the `IWTEMP` folder containing the full request and response. The number of log files can also grow substantially large — ensure you use extended logging for a limited amount of time only or implement a maintenance routine to manage this folder.

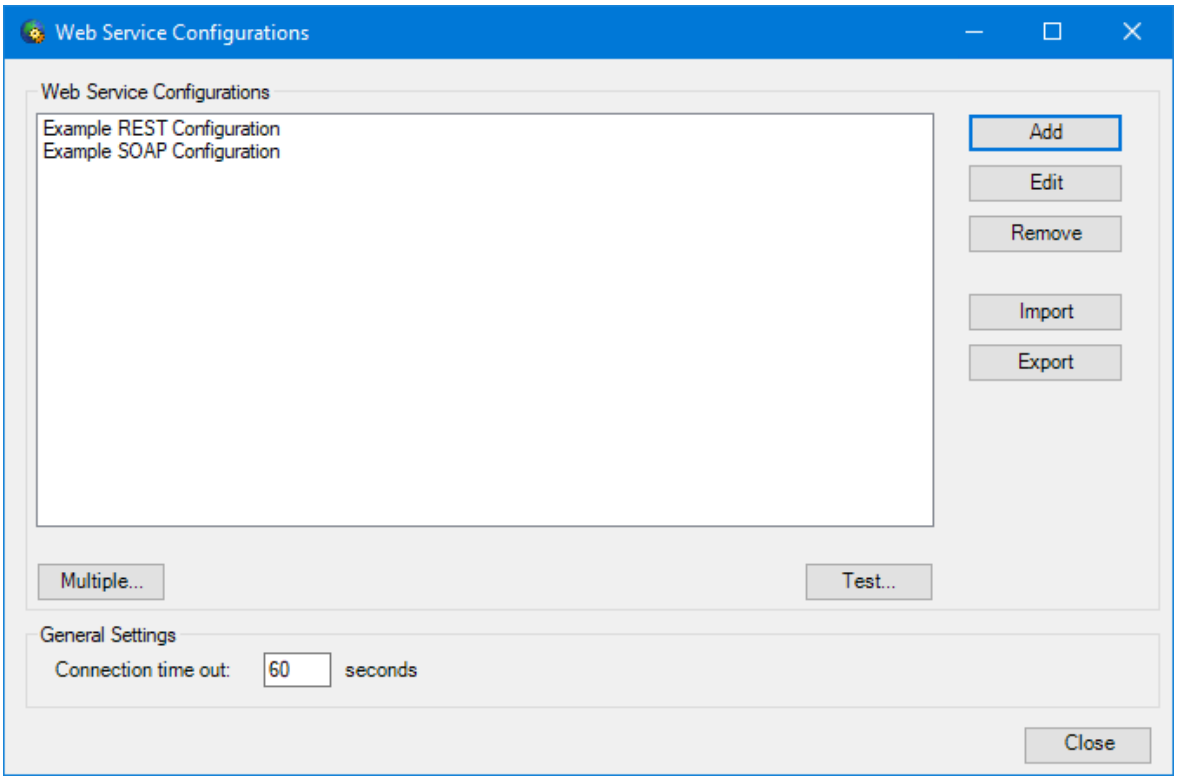


You turn on **Extended Logging** in the **General** tab of the **HTTP** or **WSDL Web Service Configuration** windows:



Additional Web Service Settings

The following settings are found in the global configuration window; use these settings to further control the web service behaviour at task runtime:



Web Service Connection Time Out

By default, the **Web Service Connector** assumes a connection has failed if no communication has been received after 60 seconds — an error is written to the BPA Platform event log. To change this, enter a time period between `1` and `9999` seconds, for the **Connection time out** parameter.

Adding Multiple Web Services to a Single Configuration Entry

If required, you can combine multiple web services into a single web service configuration entry. This is typically used where a web service provider publishes individual services for differing entities in a single application, or where different endpoints are used for specific activities, for example, logging in uses a separate endpoint to all other operations.

For this scenario, you would add or import a single web service configuration first, then use **Multiple** to add the others.

Step Configuration

When creating new tasks, the **Web Service Connector** tool is located under **Data Connectors** of the Task Browser.

> **NOTE:** It is not possible to add the **Connector** to the task until at least one connection to a web service has been defined in the global configuration.

To add a new **Web Service Connector** step to an existing task, do the following:

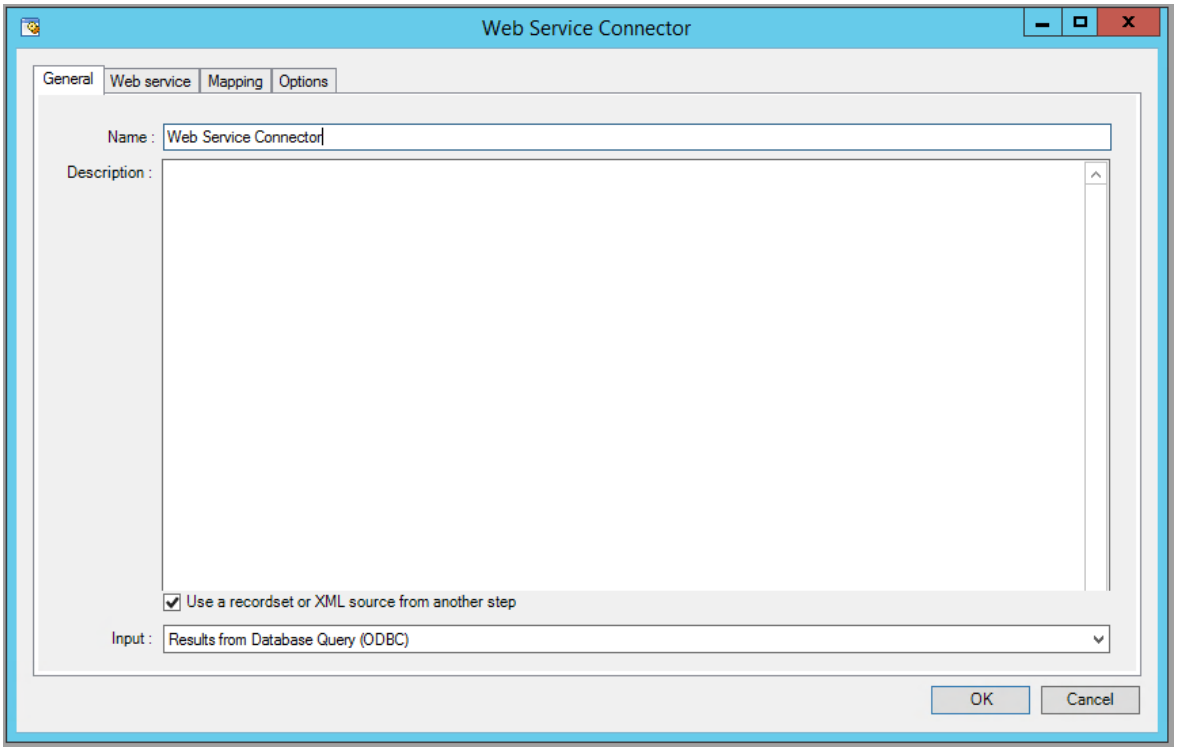
From the relevant task, either:

- Click and drag the **Web Service Connector** icon from the **Task Browser** to the task **Design** area.
- From the task's **Design** tab, right-click on empty space and select ***New > Data Connectors > Web Service Connector***.

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

About the General Tab

The **General** tab allows you to name the step and, if required, select where to get the source recordset or XML document for the web service from.



Provide a meaningful **Name** . This is the name used in the task's **Design** tab.

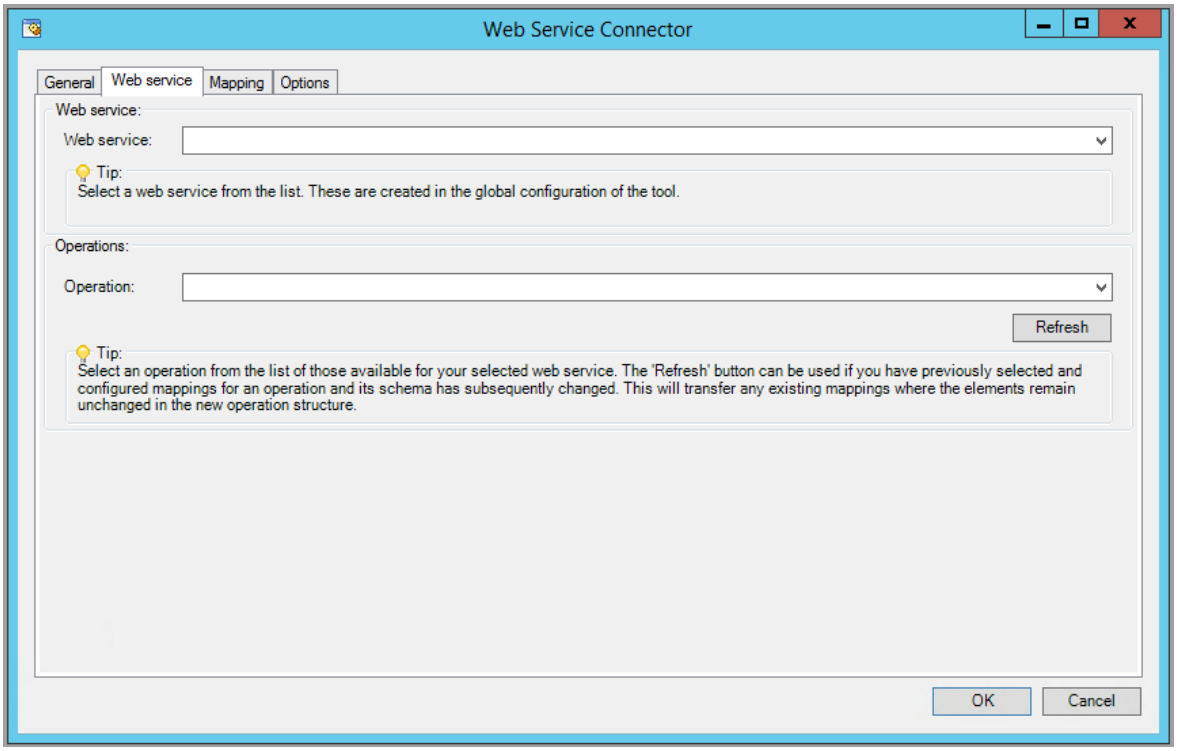
If required, enter a **Description** for this step.

If required, you can choose a task step whose output provides the input data source for this step. Elements from the connecting web service can then be mapped to elements in the input data — enable **Use a recordset or XML source from another step**.

From the **Input** drop-down, you select the step whose output is consumed by this step. If the **Web Service Connector** detects a complex schema in the data source, you are prompted to define any elements or data types.

About the Web Service Tab

You specify the **Web service** and **Operation** this step must use.



All web services created in global configuration (see *Creating a New Web Service Configuration*) are presented here.

The **Web Service Connector** queries the web service for a list of all available operations. These are then presented in the **Operation** drop-down. If the operation contains complex elements or data types that need defining, you are prompted to define these.

Once chosen, the operation's schema is presented in the **Mapping** tab.

Refreshing the Operations' Schemas

Use **Refresh** to ensure the latest operation schemas are downloaded. Any existing mappings (see *About the Mapping Tab*) will remain so long as the mapped elements are present in the same area of the schema structure.

NOTE: For HTTP and REST web services, you should use **Refresh** whenever a change is made to any part of an operations' schema.

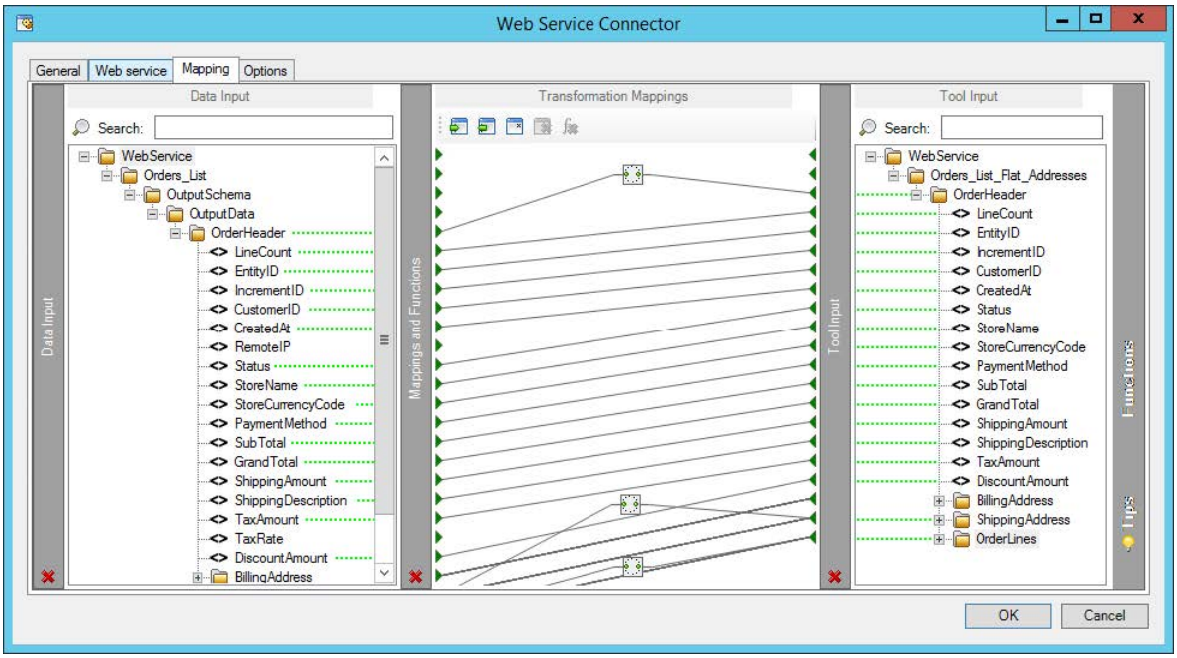
For SOAP web services, use **Refresh** only when changes have been made to the advanced features of an operation, such as, when adding or removing headers.

Changing Web Services when Mappings Exist

If required, you can switch between web services and/or operations without affecting any mapped elements (see *About the Mapping Tab*). This is particularly useful when switching from a test to a live environment, or when switching to a web service that contains similar operation schemas.

About the Mapping Tab

The **Mapping** tab of the **Web Service Connector** allows you to define the mappings between the input data source (see *About the General Tab*) and that required by the connecting web service. This defines how, at runtime, the incoming data is to be translated into the XML required for the relevant object.



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

The left-hand **Input Data** pane shows those data source fields available for mapping . The right-hand **Output Data** pane displays those fields for the selected **Object** and **Operation** combination.

TIP: If **No data source** was selected, you can create the input source in the **Transformation Mappings** pane by assigning variables to a **Fixed / Dynamic** function.

Creating Mappings

Create links by dragging and dropping a **Input Data** field onto its corresponding **Output Data**. Those marked with  are mandatory and must be included (mapped) in the output XML schema. Only linked fields are used in the output XML.

Each operation has an additional field, `SupplementaryReference`, which allows for traceability when transferring data from one place to another. When mapped, the data resides locally at runtime. It is added to the output, and creates a record for reference purposes only — you can choose to map any field to `SupplementaryReference` to assist with checking where the data originated from or at what time the data transfer occurred, for example.

Using Functions

The **Mapping** tab makes use of the Data Transformation Layer (DTL) feature of BPA Platform, where you can use available functions to manipulate the data. Use the **Functions** pane to add transformation functions:

Function Type	Function	Description
Aggregation	—	Aggregation functions define operations for specific nodes.
	 Node Count	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. For more information, refer to the product help.
	 Sum	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. For more information, refer to the product help.
Data	—	Data functions perform operations on input data to generate new output data.
	 Fixed / Dynamic	This function passes a static or dynamic value to the output XML schema. Use a variable or recordset column to generate the dynamic data. For more information, refer to the product help.
	 Run VBScript	Use this function to perform VBScript operations to process input data, generate output data, or both. For more information, refer to the product help.

Function Type	Function	Description
Lookup	—	Lookup functions are used to find values in a nominated source, by a key.
	 External Lookup	Use this function to lookup values from an external database. This uses existing **Database Query (ODBC)** or **Database Query (OLEDB)** global connections. For more information, refer to the product help.
	 Internal Lookup	Use this function to find an alternative value for input data from a predefined lookup table. For more information, refer to the product help.
Looping	—	Looping functions loop through the input recordset or XML document to perform functions on all iterations of a node.
	 Interleaved Merge	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. For more information, refer to the product help.
	 Simple Loop	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. For more information, refer to the product help.

Additional Functionality

In the **Transformation Mappings** pane, click:

 to import and use an existing mappings file (`.DTLX`) from a previous task

 to export the current mappings for use in another task or as a backup (`.DTLX`)

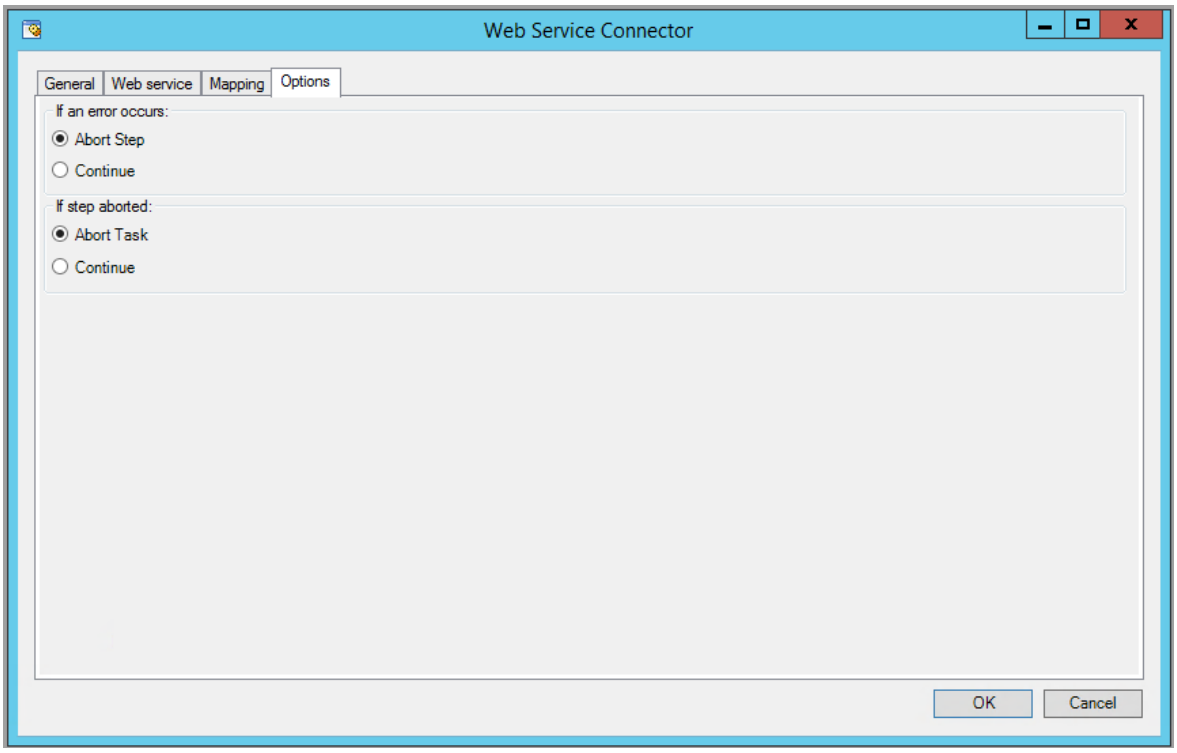
 to reset all current mappings

 to delete the highlighted mapping

 to delete the highlighted function

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.

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