BPA
Platform

Technical Overview

Call Task 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

The information provided in this publication may contain errors, omissions, or typographical errors or may be out of date.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

The **Call Task** tool is used to trigger another task to run from the current task. If required, you can use consumed XML from another step or variables to pass task data between the two during run-time.

Tasks initiate all steps irrespective of whether they are required. This can result in extended completion times. The best use of the **Call Task** tool is to break down large tasks into smaller, manageable tasks consisting of a few steps, with a central "control" task to trigger the other tasks. By doing this, you balance the load on the computer as the BPA Platform server only needs to initiate those tasks and steps required for that process.

Features

- Configure the **Call Task** API via an easy-to-use interface
- Trigger another task to run either in sync with the original task or after
- Call other tasks to run that are present on another BPA Platform server
- Pass variables and XML data into a **Call Task** step for use in the task to be run
- Retrieve data back from the called task when run in sync
- Create a record of a task call for reference purposes

Working with Other Tools

Consuming from Other Tools

The following lists those tools that a **Call Task** step can consume XML from. Data from these steps can be passed to variables in the task being called.

Icon	Tool Name	Tool Category
	Import Flat File	Input
xml	**Import XML Document**	Input
	Retrieve Text Message	Input
</>	**Convert Recordset to XML**	Format
	Transform Data	Format
	Call Task	Execute
	Applications Platform Connector	Data Connectors

Objects Consumed

The **Call Task** tool consumes the following objects exposed by other steps:

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

Exposing to Other Tools

The following lists those tools that can directly consume a **Call Task** step's output. This is useful when the tool retrieves output data back from the called task and saves to a variable. The variable data can then be used in the original task.

Icon	Tool Name	Tool Category
	Retrieve Text Message	Input
	Convert XML to Recordset	Format
	Format as Flat File	Format
	Run Microsoft Reporting Services	Format
	Transform Data	Format
	Save File	Output
	Call Task	Execute
	Applications Platform Connector	Data Connectors
WEB SERVICE	**Web Service Connector**	Data Connectors

Objects Exposed

The **Call Task** tool outputs the following objects which can be consumed by other tools:

- **InputData**

  This document contains the XML received by the **Call Task** 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 called task for all operations. Also included are the key fields for the mapped elements affected by the used operation and a `SupplementaryReference` field for task auditing purposes.

    The mappings made 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 called task

    `<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 called task

    `<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>`

  - □ **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 **Call Task** step.

Where Can the XML Output be Used?

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.

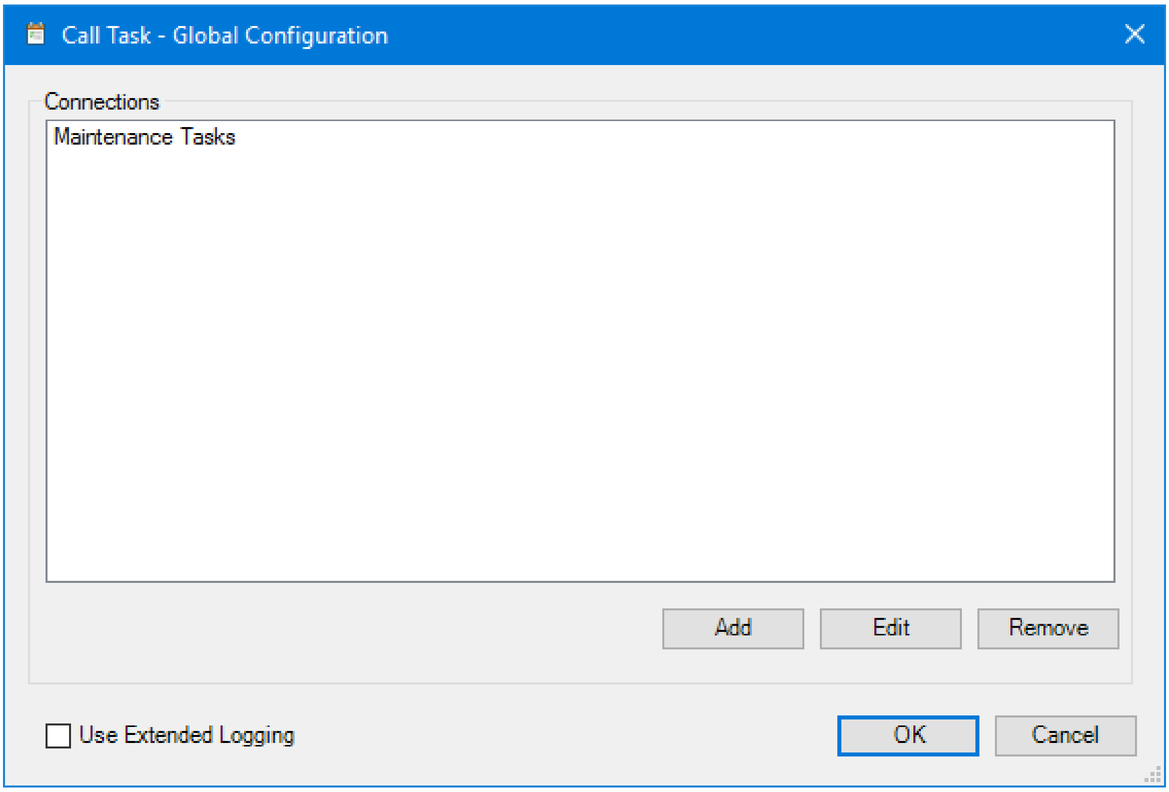
The XML documents are also available as consumable objects from the BPA Platform Browser (`XmlString`). When used in a task step, such as **Format as Text** or **Save File**, this exposes the actual XML string.

Global Configuration

Before using the **Call Task** tool in a task, you must select the folders which contain the tasks available to use elsewhere.

You open the **Call Task - Global Configuration** interface by either:

You open this interface from the resources tree — expand ***System* > *Tools* > *Execute*** and double-click ***Call Task*** in the items list.



Using Extended Logging

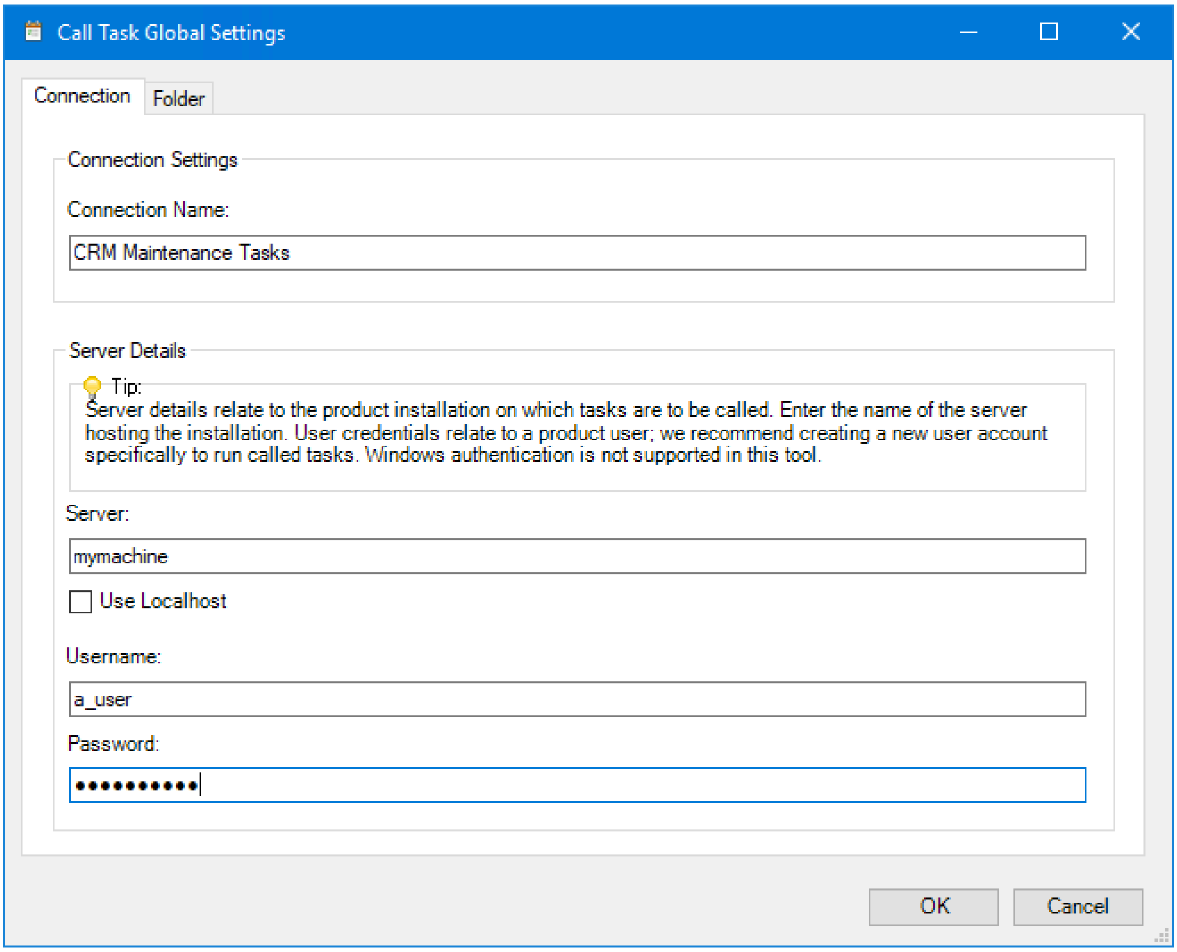
Enabling this option exposes the XML parsed. The task's folder is also exposed.

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 (***Manage* > *Event Log***).

About the Connection Tab

In the **Connection** tab, you specify the location of the task folder this connection refers to.



Connection Settings

Connection Name: CRM Maintenance Tasks

Server Details

Tip:
Server details relate to the product installation on which tasks are to be called. Enter the name of the server hosting the installation. User credentials relate to a product user; we recommend creating a new user account specifically to run called tasks. Windows authentication is not supported in this tool.

Server: mymachine

□ Use Localhost

Username: a_user

Password: ●●●●●●●●●●

In **Server**, enter the hostname or IP address of the BPA Platform server hosting the required tasks.

> **NOTE:** Do not enable **Use Localhost** or type `localhost` as the **Server** address if the BPA Platform client and server are located on separate computers. If you are using a remote client, use only the hostname or IP address of the BPA Platform server. Using **localhost** causes the task to fail as it tries to locate and run a task not present on the server computer.

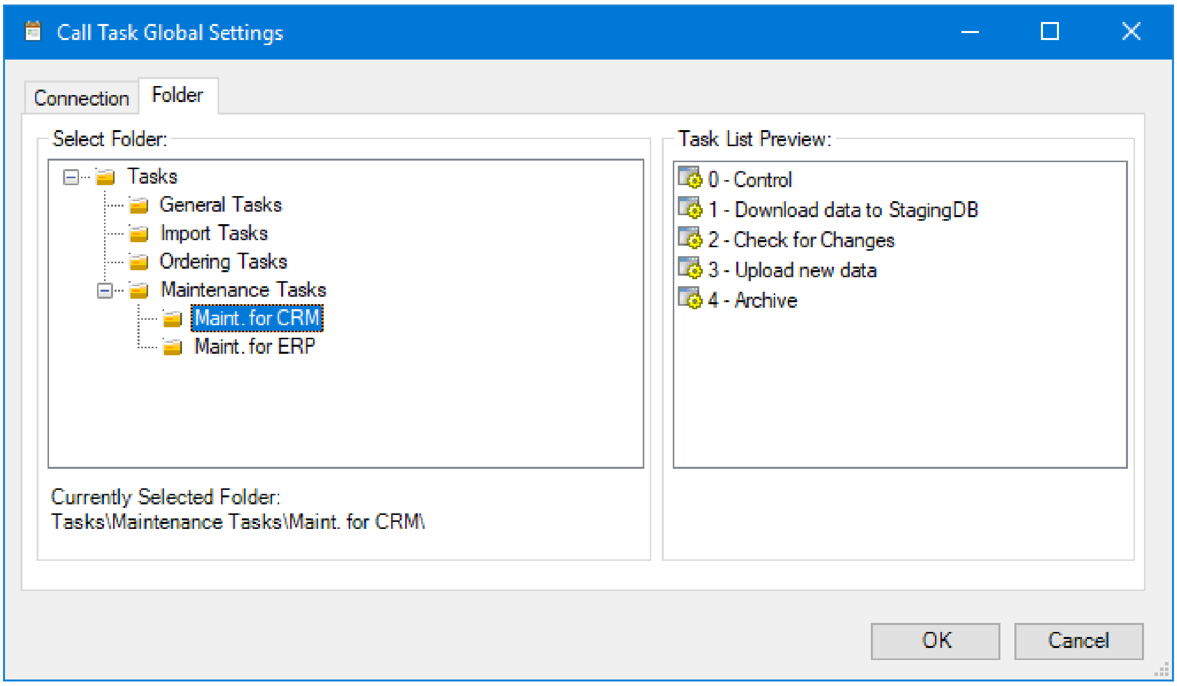
Enter the credentials (**Username** and **Password**) of a BPA Platform account that has permissions to:

- Use the **Call Task** tool
- Access the folder and required task inside

TIP: Create a separate BPA Platform account specifically for running called tasks — this prevents clashes with logged in users. Windows accounts are not supported.

About the Folder Tab

From the **Folder** tab, select the relevant task folder that contains the task(s) to be called.



Currently Selected Folder:
Tasks\Maintenance Tasks\Maint. for CRM\

NOTE: You cannot select an individual task at this point. The layout of the **Folder** tab is only to allow you to view the folder contents. Individual task selection is made when adding a **Call Task** step — see About the Mapping Tab.

Step Configuration

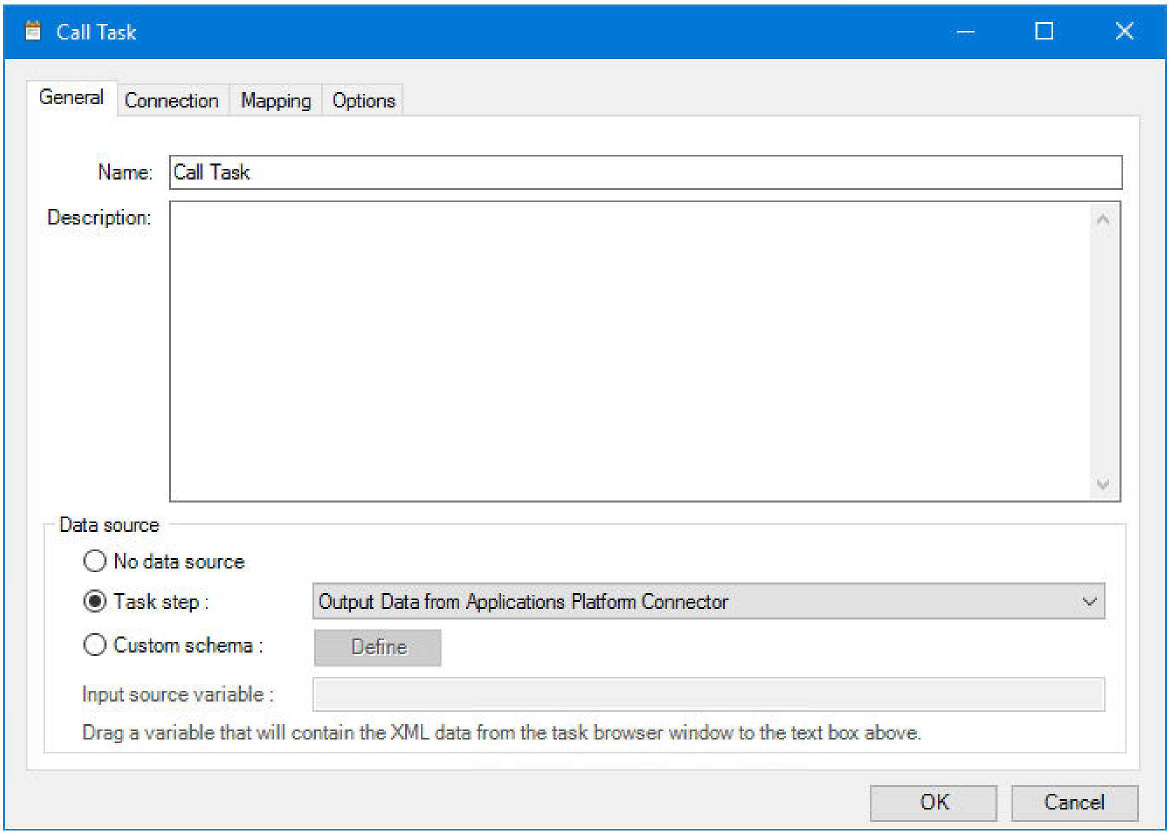
To add a new **Call Task** step to an existing task, you either:

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

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

About the General Tab

Use the **General** tab to name the BPA Platform data source to be mapped to the called task's objects. The data source must be in XML form. To map BPA Platform recordsets, configure a **Convert Recordset to XML** step in the task before the **Call Task** step, then select this as **Task step** (see below).



Call Task

Name: Call Task

Description:

Data source

○ No data source

● Task step : Output Data from Applications Platform Connector

○ Custom schema :

Input source variable :

Drag a variable that will contain the XML data from the task browser window to the text box above.

Provide a meaningful **Name** and **Description** for this step.

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

Data source can either be:

- **No data source** — If you don't make use of a dedicated XML input source, select this option to use BPA Platform variables in place of the XML or recordset objects' fields.

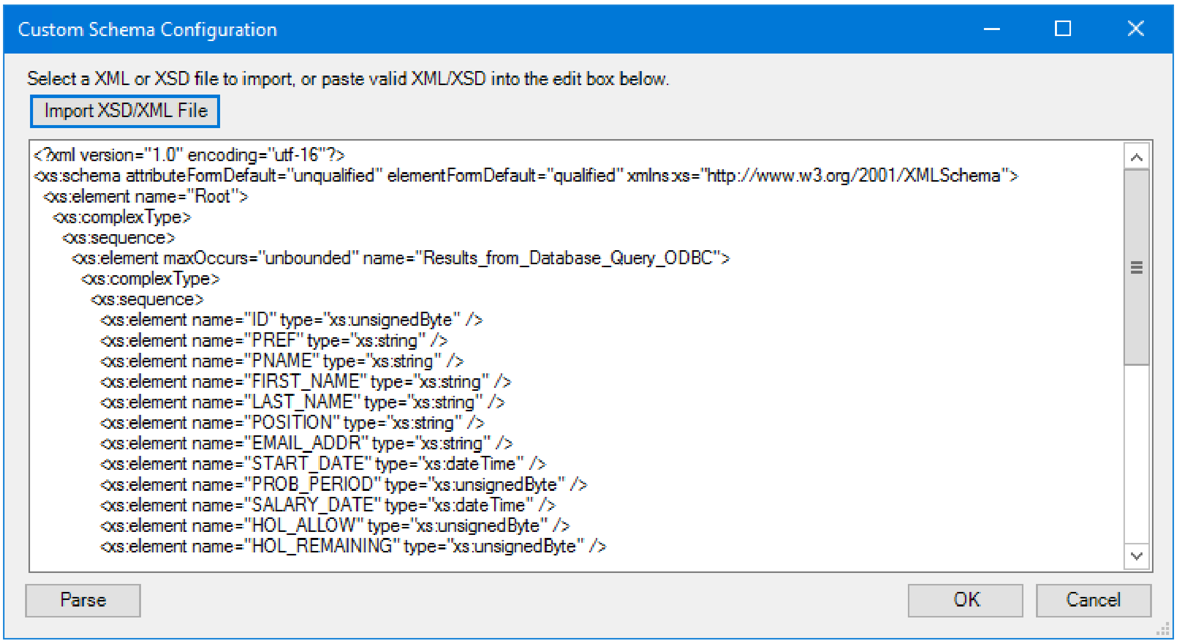
  For example, you can extract various bits of information from an email and store them in BPA Platform variables. The variables are mapped to the called task's operational objects and fields for use by the called task during its run time — .

- **Task step** — The data source can be set to an available BPA Platform XML data source.

  Only those steps that are capable of natively exposing an XML document at runtime are listed. This may be another **Call Task** step, or a tool, such as **Convert Recordset to XML**.

- **Custom schema** — An XML schema defines the structure of the parsed XML: what tags are present, and the nesting of the tags. You **Define** the schema of the XML that is used as the input data source for this step. The **Call Task** tool uses the industry standard XSD format. Any XML processed by this step must conform to this schema else an error will be reported.

  If the XSD schema is available, either import it into the **Custom Schema Configuration** (use the **Import XSD/XML File** button), or copy and paste it into the configuration box.



Select a XML or XSD file to import, or paste valid XML/XSD into the edit box below.

```
<?xml version="1.0" encoding="utf-16"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="Root">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="unbounded" name="Results_from_Database_Query_ODBC">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="ID" type="xs:unsignedByte" />
       <xs:element name="PREF" type="xs:string" />
       <xs:element name="PNAME" type="xs:string" />
       <xs:element name="FIRST_NAME" type="xs:string" />
       <xs:element name="LAST_NAME" type="xs:string" />
       <xs:element name="POSITION" type="xs:string" />
       <xs:element name="EMAIL_ADDR" type="xs:string" />
       <xs:element name="START_DATE" type="xs:dateTime" />
       <xs:element name="PROB_PERIOD" type="xs:unsignedByte" />
       <xs:element name="SALARY_DATE" type="xs:dateTime" />
       <xs:element name="HOL_ALLOW" type="xs:unsignedByte" />
       <xs:element name="HOL_REMAINING" type="xs:unsignedByte" />
```

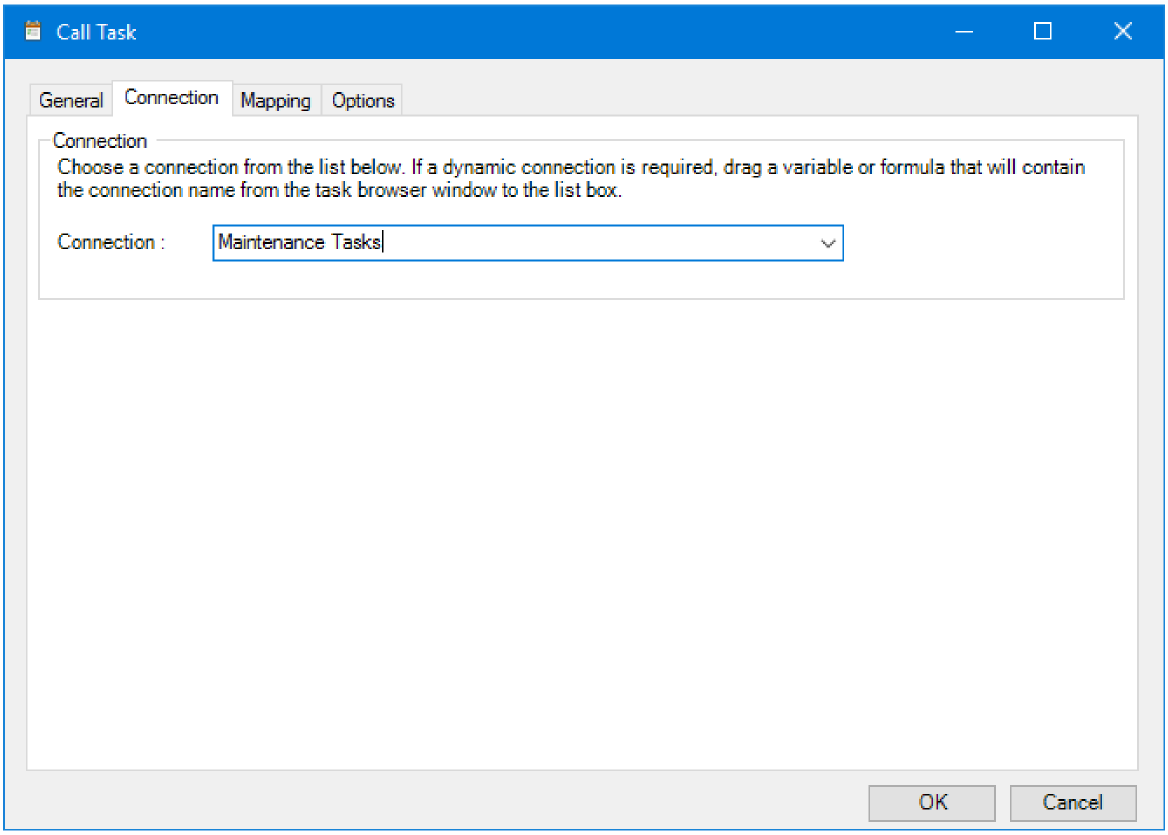
  If the XSD schema is not available, you can import an example of the runtime XML (**Import XSD/XML File**), or copy and paste it into the configuration box. Use the **Parse** button to create the schema.

  □ **Input source variable** — As well as defining the schema, specify the BPA Platform variable that contains the XML data at runtime.

About the Connection Tab

All connections created in the Global Configuration are presented here.

Alternatively, you can use a BPA Platform variable to create a dynamic connection, where the connection used is determined by runtime circumstances. At runtime, the contents of the variable must match the name of one of the Global Configuration connections — this is case-sensitive.

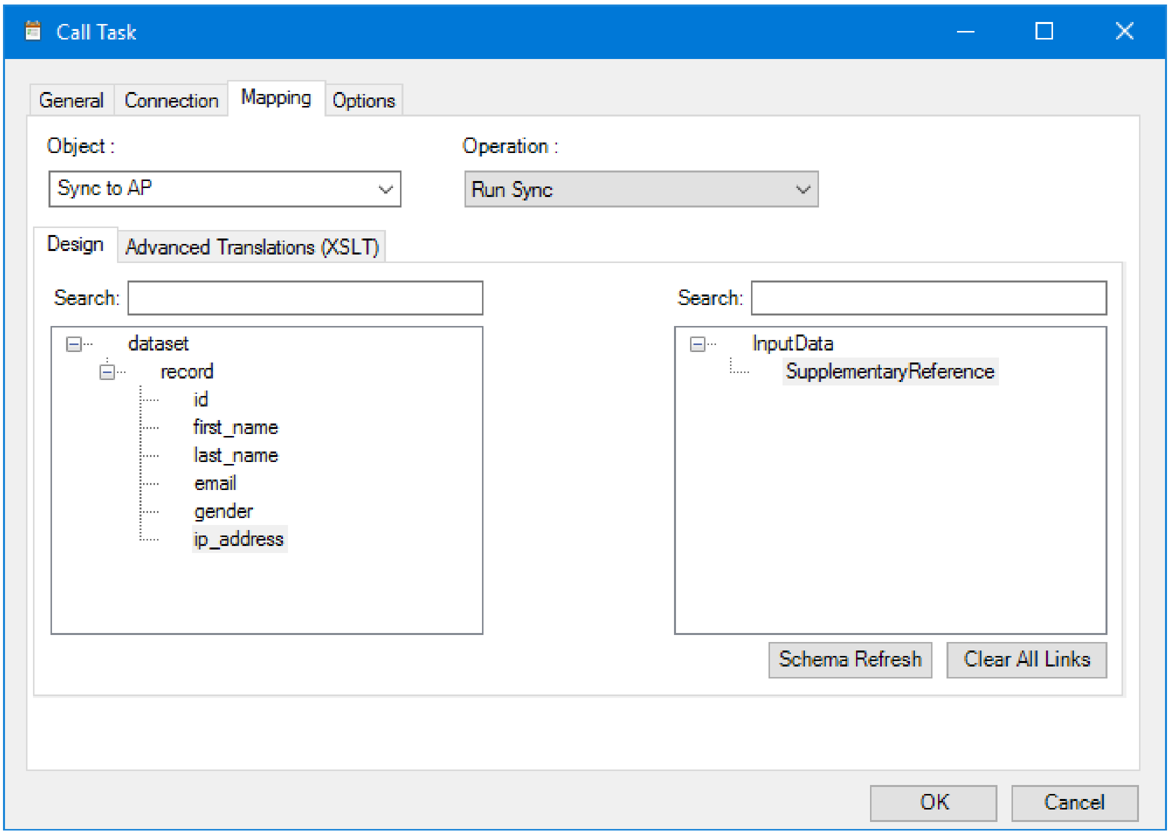


Choose a connection from the list below. If a dynamic connection is required, drag a variable or formula that will contain the connection name from the task browser window to the list box.

Connection : Maintenance Tasks

About the Mapping Tab

The **Mapping** tab is used to select the task to run. However, you do not need to map data if the called task does not require any input data.



Object : Sync to AP

Operation : Run Sync

The **Object** drop-down shows the available tasks from the selected task folder. The **Operations** drop-down shows the operations available for the selected **Object**

Further down, the left-hand XML tree structure shows those data source fields available for mapping . The right-hand XML tree structure displays those input fields for the selected **Object** and **Operation**.

TIP: If you have selected **No data source** in the **General** tab, you must add BPA Platform formulas and variables to the left-hand data source XML tree structure before mapping.

Creating Mappings

Create links by dragging and dropping a data source field (left) onto its corresponding **Call Task** input field (right). Only linked fields are used in the output XML.

NOTE: You do not need to map data (create links) if the called task does not require any input data from this step during run-time.

BPA Platform formulas and variables can be included in the source data even when using a **Task step** or **Custom schema** — drag them from the BPA Platform Browser to the data source's XML; these can then be linked to the relevant input fields. BPA Platform recordsets must first be converted to XML using either the **Convert Recordset to XML** or **Transform Data** tool before they can be used here.

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.

The tool uses eXtensible Stylesheet Language Transformations (XSLT) to translate the received XML. The **Advanced Translations (XSLT)** tab shows the XSLT generated for the links created for the object and operation. Use **Enable Free Type Mode** to directly edit the XSLT — this is particularly useful when translating a non-standard requirement.

Refreshing the Called Task's Schema

If new task data, such as variables, are added to the called task after this step was created, you must refresh the called task's schema in order to see the new data in the **InputData** schema — click **Schema Refresh**.

> **NOTE:** Only **Parameter**-type variables can be processed by the **Call Task** tool. **Datafile** and **Password** type variables are not supported; nor are variable arrays.

Refreshing the Called Task's Schema

If new task data, such as variables, are added to the called task after this **Call Task** step was created, you must refresh the called task's schema in order to see the new data in the **InputData** schema — click **Schema Refresh**.

Supported Operations

The **Object** drop-down shows the available tasks. The **Operations** drop-down shows the operations available for the selected **Object**:

- **RunSync** — Runs the called task when called by the original task; the original task waits for the called task to complete before continuing. Parameter-type variables can be used to pass data between the tasks; with RunSync operations, the original task cannot complete until the data is passed back.

  The data passed back is store in the output XML of the original task. This in turn can be consumed by another step capable of processing XML, such as the **Convert XML to Recordset** tool.

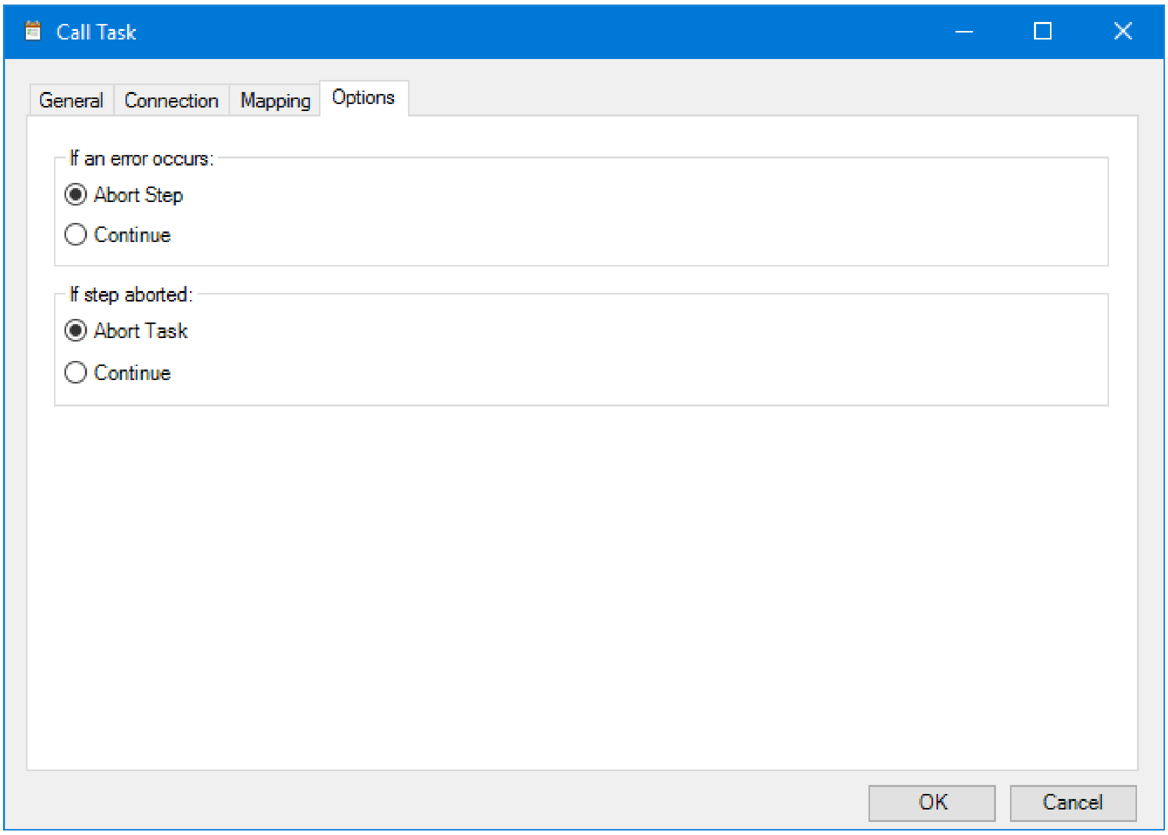
  Use this operation if the original task requires data from the called task.

- **Queued** — Queues the second task but continues to run the original task to completion; both tasks run independently of each other. Data can be passed to a parameter-type variable in the called task. However, this operation does not allow data to be passed back as the original task has already completed running before the second task initialises.

  Use this option when only triggering a task to run, with or without data from the originating task.

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:

● Abort Step

○ Continue

If step aborted:

● Abort Task

○ Continue

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