Opc HDA SimApi User Guide

1. Introduction

This document is the user guide for the *Opc HDA SimApi* from Umetrics.

It uses *OPC HDA* (Historical Data Access) to connect to an OPC HDA Server. For more information about OPC HDA, see http://en.wikipedia.org/wiki/OPC_Historical_Data_Access.

For a detailed list of changes in different versions of this SimApi, see the *Version Info.txt* file that comes with the installation.

This SimApi can be used by SIMCA-online, SIMCA-Batch On-Line, SIMCA-4000, or SIMCA.

For more information on available SimApis, see www.umetrics.com/simapi.

1.1 Features

- Reading of both historical and current data through OPC HDA. Note that OPC DA is not used in this SimApi but that HDA supports reading *current* data as well as historical.
- Support for reading many observations at once.
- Write back of data from SIMCA-online to the OPC server.
- Basic filtering of which branches (nodes) will be exposed through the SimApi.
- Support for tags with array data. If configured for it, the SimApi will expand array data to multiple individual tags.

Note that this SimApi does not implement a batch node for batch data. For this you can use the Batch Table Wrapper SimApi.

The sub sections describe some of the above features in detail.

1.1.1 Support of Array data in tags

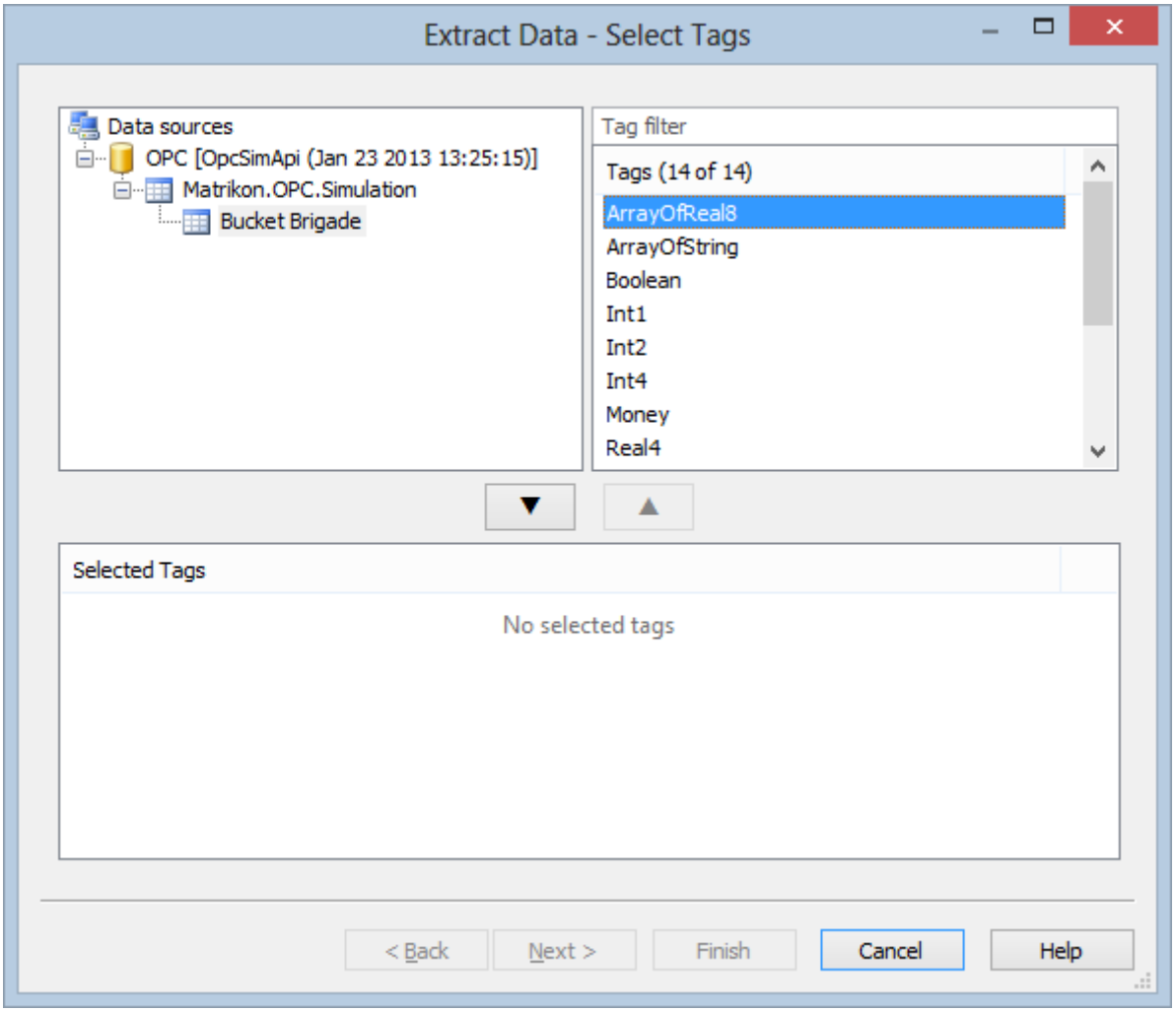
Tags in OPC can hold array data; i.e. multiple values stored in one tag. The SIMCA products cannot deal with array data so this SimApi works around this by expanding array data for specified tags into individual values put in multiple new virtual tags that SIMCA-online[1] can use. These virtual tags are created in a new node with the same name as the tag with array data.

As SIMCA-online reads tag names when the server starts, the array is *not* allowed to change size. If the array has a different size from when configuring it, a message will be written to the log file and *missing values* will be returned for the whole array.

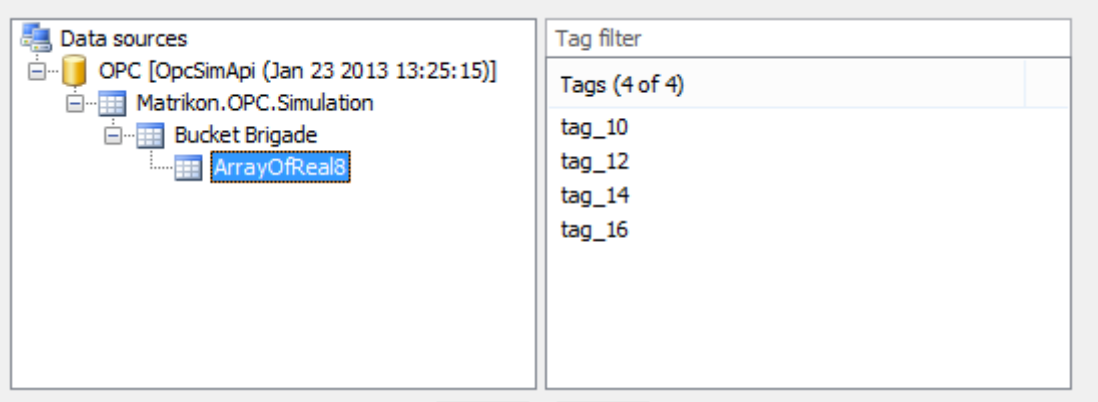
You have to explicitly configure each tag that contains array data in order for SIMCA-online to be able to recognize it. This configuration is made in the XML-file and the necessary syntax is given in 2.4 below.

This is how an array tag looks in SIMCA-online before being configured as array data:

[1] Only SIMCA-online an handle array data like this. SIMCA and previous products such as SIMCA-Batch On-Line and SIMCA-4000 does not support nodes inside nodes which is a requirement.



This is how the same dialog looks after configuring the tag as holding Array data: Notice that four individual tags now are listed. These don't exist in OPC but rather are generated by the SimApi for use in SIMCA-online.



1.2 Prerequisites

This SimApi requires the **OPC Core Components Redistributable** to be installed on the computer where you want to use the SimApi.

Go to opcfoundation.org and download the latest available OPC Core Components. You should download the 32-bit (x86) or 64-bit (x64) version that matches the *Windows version* of the computer you will use the SimApi on.

Note that the 64-bit version of the Core Components Redistributables installs the 32-bit version as well, so it is the only package that needs to be installed on 64-bit systems (even if you plan to use a 32-bit SimApi there).

1.3 Installation

The SimApi comes in a 32-bit (x86) and a 64-bit (x64) version. Use the one that matches the program (for example SIMCA-online or SIMCA) you will use the SimApi in.

Run the SimApi setup program to install the SimApi on your computer where you want to use it.

Don't forget to install the OPC Core Components as described in 1.2 above.

Then you need to set up the SimApi as described in 2 below.

Also refer to the following sections for information on important locations for log- and settings files and troubleshooting tips.

1.4 Configuration file and log file locations

The SimApi stores its settings in a XML settings file in the *Program Data* folder[2]. This folder is located in different places depending on the Windows version:

Window Vista and Windows Server 2008 and later: C:\ProgramData\Umetrics\SimApi

Windows XP: C:\Documents and Settings\All Users\Application Data\Umetrics\SimApi

This folder also contains the log file that the SimApi writes. This log file is useful for troubleshooting.

1.5 Troubleshooting

It can be tricky to get OPC working, because it depends on DCOM and there are a range of security settings that might have to be changed in Windows for it to work.

Refer to the following OPC support documents for information about this. These files are included in the SimApi installation and put in its program folder[3].

- *OPC_and_DCOM_5_things_you_need_to_know.pdf*
- *OPC_and_DCOM_Troubleshooting.pdf.*

When testing SimApis on freshly installed computers[4], Umetrics has found the following relating to user accounts and security;

- The *same user account and password* must exist on both the SimApi computer and the OPC server computer. Otherwise it will result in problems when configuring the SimApi, specifically enumerating the OPC servers on the remote server computer so that the branches (nodes) can be selected. If both the server and SimApi computer is on the same domain this isn't a problem, but if one of the computers are not part of a domain it is very important.
- However, even if configuring the SimApi isn't possible because of the above mentioned reason, the SimApi can still work if the user account that the SimApi runs as has sufficient permissions on the OPC server itself. This means the copying a SimApi configuration file from another computer will work.

More tips;

---

[2] This folder is normally hidden in Windows so in order to see it in Windows Explorer you should configure Windows Explorer to show hidden files. Note that you still can navigate to a hidden folder by copying and pasting the folder path to Explorer's address bar.

[3] This typically means a folder in c\program files (x86)\Umetrics\SimApi (or c:\program files\Umetrics\SimApi).

[4] Specifically we tested on Windows XP SP3 and Windows 7 computers that are not part of a domain, connecting to a remote Matrikon OPC server on a computer part of a domain. In this instance we had to create a local account with the same name on both the OPC server and test computers, and be logged on as that account on the test computers.

- If you experience issues when configuring the SimApi, such as not seeing any OPC servers on a computer where you expect them you can use the Credentials-button in the dialog to setup the user name and password to use to connect to the OPC server computer. Use the Test-button to test your connection.
- Even if the configuration of the SimApi is successful (this runs in the security context of the user that performs the configuration), it can still happen that SIMCA-online service cannot start. The reason for this can be that the service runs as the user Local System on the computers, and that the computer account[5] of the SIMCA-online server computer doesn't have the appropriate rights on the OPC server. In this case specifying the Credentials in the SimApi configuration can help.
- If OPC Core Components isn't installed you will get the error message "Class not registered" (or "Interface not registered") when expanding an OPC server in the SimApi configuration dialog.

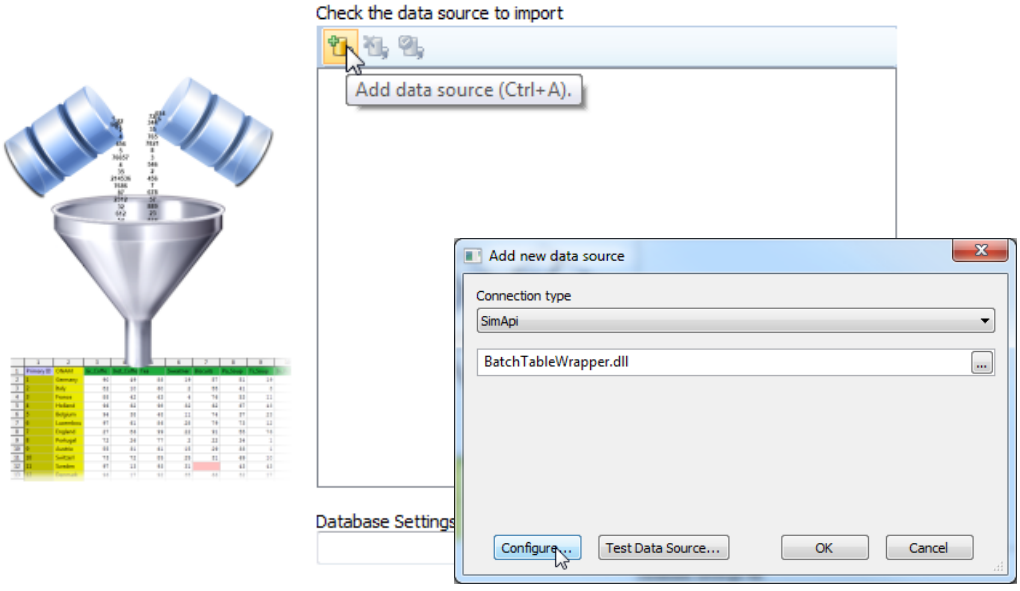
A good way to test OPC connectivity is to use the free software MatrikonOPC HDA Explorer which you can download from https://www.matrikonopc.com/products/opc-desktop-tools/index.aspx.

2. Setting up

In order for SIMCA or SIMCA-online to be able to use the SimApi you need to configure it as described here.

2.1 Setting up the SimApi for use in SIMCA

Start SIMCA-import by *File->New Regular Project->New Batch Project*. If the database import wizard is not opened automatically, open it from *File->New Spreadsheet->From Database*. Click on *Add data source*.

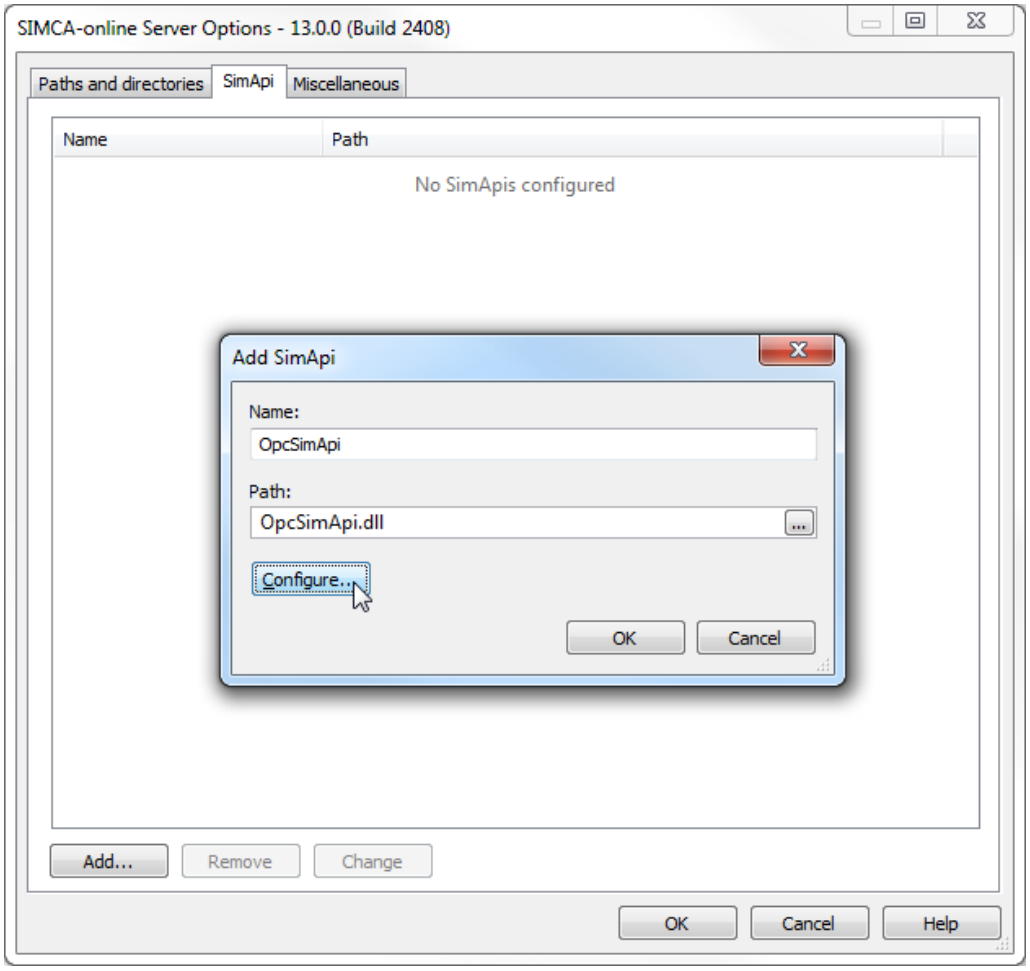


Select *SimApi* as connection type, find the installed BatchTableWrapper.dll and click on *Configure…* button and the configuration dialogs are opened. Continue to 2.3 below.

2.2 Setting up the SimApi for use in SIMCA-online

Start the SIMCA-online Server Options utility from the start menu. Go to the SimApi tab and click *Add...*

[5] A computer account is similar to a user account but for a specific computer. It is of the form ComputerName$. For more information about Local System and computer accounts, see http://msdn.microsoft.com/en-us/library/windows/desktop/ms677973(v=vs.85).aspx

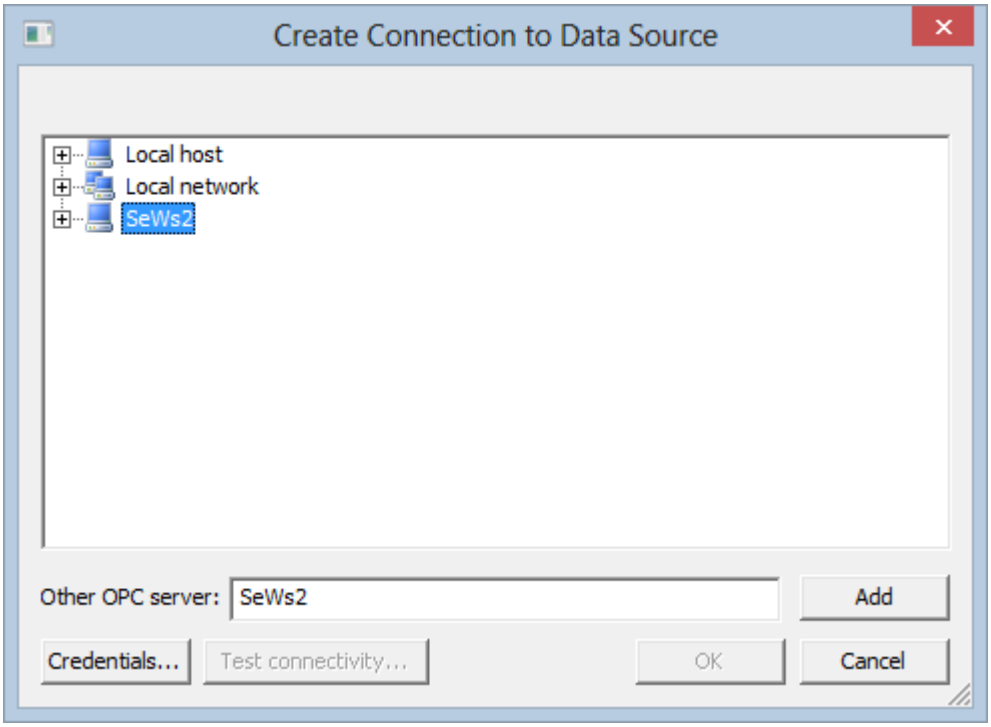


Enter a name, for example OpcSimApi, and then click the …-button to browse to and select the OpcSimApi.dll. Click the *Configure* button and the Opc SimApi Configuration is opened. This is described in the next section.

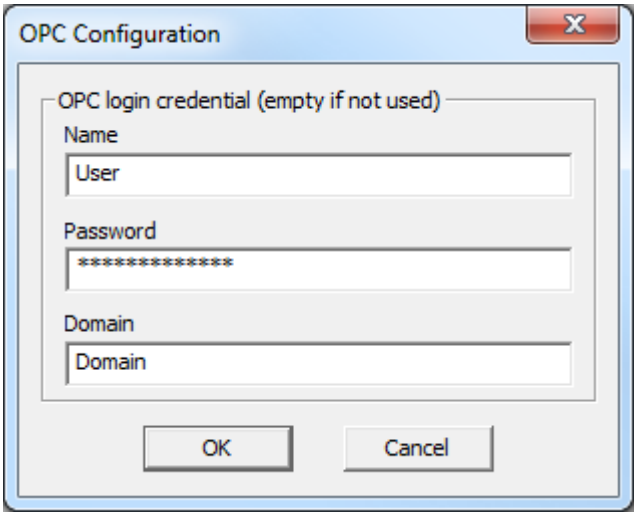
2.3 Opc SimApi Configuration

A range of settings are required for the SimApi to connect to your OPC server. All settings that you make using the graphical configuration described below are saved in the XML-file (see 1.3 above).

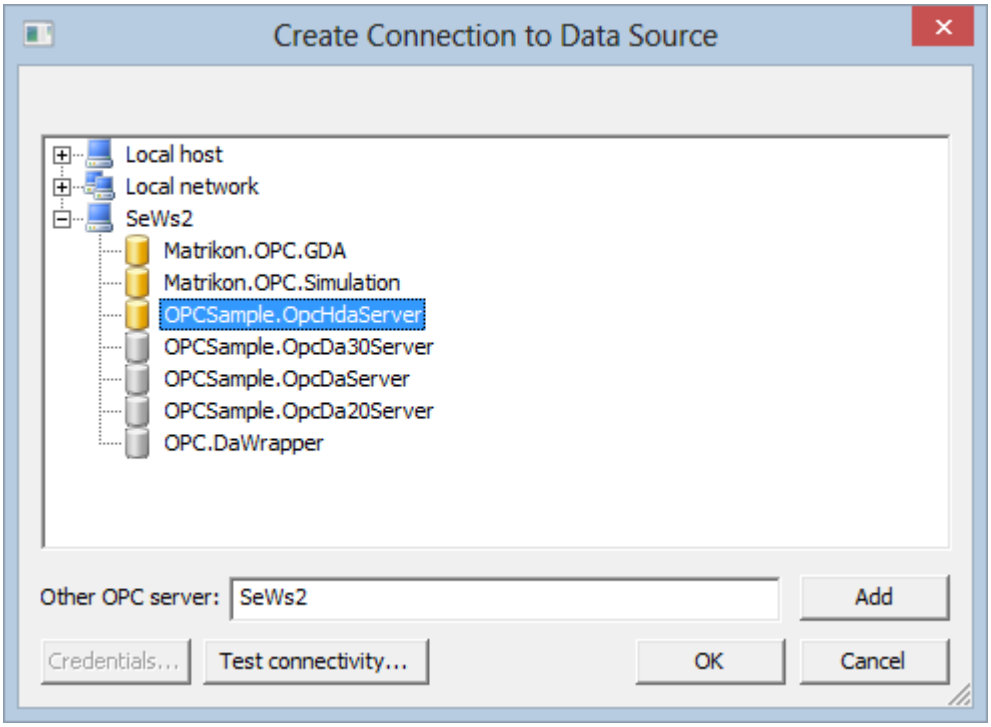
Browse to the computer where the OPC Server is located using the + before Local network shown in the pictures below, or type a computer name and click the Add button. This is how the dialog looks after adding a computer:



After selecting a server, you can optionally use the Credentials-button to specify a user, domain and password that you want to connect to the computer with. If not provided the account you are currently logged on as will be used for the configuration.



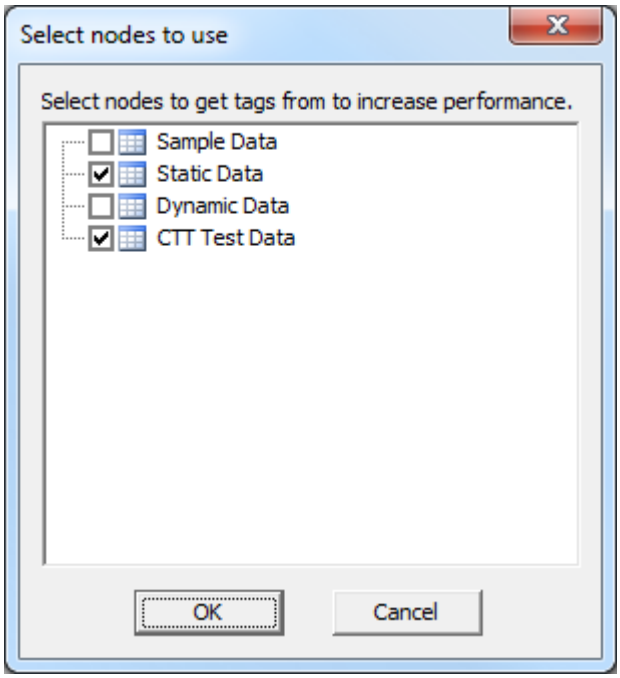
Click the +-icon to expand the computer to see all OPC Servers on the machine. If there are some problems with rights, or communication to the server, an error message will be presented. Otherwise a list of available OPC servers is presented. Servers listed with yellow cylinder icons are available for selection.



Select an OPC Server and use the *Test connectivity*-button to test communication with the OPC Server (by sending an OPC ping packet). A message will be displayed that shows the result of this test.

After the connection have been tested, select the OPC HDA Server you want to use, and click on the *OK* button.

If the server supports branches, the following *Select nodes to use* dialog will list the available branches. Select the branches that are relevant to the monitoring. The fewer branches selected, the better performance will be because fewer tags will be enumerated each time the SimApi is initialized by SIMCA-online or SIMCA.



2.4 The XML Configuration File

Below a sample XML file is shown. Many of the settings are made using the graphical interface above, but some has to be made by hand in the file. These settings are marked bold below, and described in the sub sections.

```
<?xml version="1.0" ?>
<settings>

    <setting key="LogFileSize" value="1048576" />
    <setting key="LogLevel" value="3" />

    <setting key="Connection" value="10443d89df125d92" />
    <setting key="URL"
value="opchda://SeWs2/OPCSample.OpcHdaServer/{6A5EEDEC-1509-4627-997F-
993CCB65AB7C}" />
    <setting key="UseNullTimeValues" value="0" />
    <setting key="BatchID" value="" />
    <nodes>
        <setting key="Node_0_0" value="Sample Data" />
        <setting key="Node_0_1" value="Sample Data" />
    </nodes>

   <arraynode tag="" size="4" prefix="tag_">

        <arraytagnamesbystartandinterval start="10" interval="2" />

        <arraytag>tag_10</arraytag>
        <arraytag>tag_12</arraytag>
        <arraytag>tag_14</arraytag>
        <arraytag>tag_16</arraytag>
        <!-- This is a sample ArrayNode showing some available settings.
You need to add a tag name and configure the xml properly for it to work. -
->
    </arraynode>
    <!-- You should add one ArrayNode for each tag that holds array data. -
->
</settings>
```

2.4.1 SimApi Log File Specific Settings

- LogFileSize – The maximum allowed size of the log file before the file is truncated
- LogLevel – The higher the value the more information is printed to the log file (log level 3 = information). Maximum value is 4 and minimum value is 0

2.4.2 ArrayNode settings – for dealing with array data in OPC tags

For an introduction to array data, see 1.1.1 above. The xml file is used to configure the array data as follows:

Multiple arraynode elements can be created, one for each tag that holds array data that you want to expose through the SimApi.

The purpose of an arraynode is to hold the necessary settings so that the SimApi can expand the array data into individual virtual tags created below a new node with the name of the tag holding the array exposed by the SimApi.

Attributes for the arraynode element:

- **tag** – should contain the full tag name of the tag that has array data.
- **size** – should contains the size of the array data that should be read. The size must match the number of elements in the array on each read through the SimApi, otherwise no values will be

returned by the SimApi and errors logged to the SimApi log file. This is by design; if the size is variable, most likely the tag names would not match anymore, and data inconsistencies could occur.

- **prefix** - added in the beginning of all generated tag names (see below), can be empty, defaults to "tag_" if not provided.

Many **arraytag** elements – each giving a name of a tag. These are optional. If used you must provide the same number of arraytags elements as the size attribute set in the arraynode. If you don't provide arraytag names, or if the number of names is wrong the default will automatically generated names as follows:

<arraytagnamesbystartandinterval start="10" interval="2" />. This tells the SimApi it should generate names for the created tags using the prefix from above together with start and interval. The resulting names are shown above in the xml sample above. The default values for start and interval are 1 if not specified.

2.4.3 UseNullTimeValues – option for how current data is read

The following option can be used to control how current data is read from the OPC server. If you don't get the expected current values, you can change the value attribute to "1".

```
<setting key="UseNullTimeValues" value="0" />
```

3. Interfaces and functions used by the SimApi

The SimApi use the following OPC HDA interfaces and functions.

- IOPCServerList2 interface
  - EnumClassesOfCategories function
  - GetClassDetails function
- IOPCEnumGUID interface
  - Next function
- IOPCHDA_Server interface
  - CreateBrowse function
  - GetItemHandles function
  - ReleaseItemHandles function
- IOPCHDA_Browser interface
  - GetEnum function
  - ChangeBrowsePosition function
  - GetItemID function
- IOPCHDA_SyncRead interface
  - ReadRaw function
- IOPCHDA_SyncUpdate interface (optional)

- InsertReplace function

4. Support

See http://www.umetrics.com/support