OPC HDA SimApi User Guide

2020-11-16

sartorius.com/umetrics

Contents

1 Introduction

This document is the user guide for the **OPC HDA SimApi** from Sartorius Stedim.

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 Easy Analytics, SIMCA, or SIMCA-online.

For more information on available SimApis, see 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.
- Node name aliases can be used to keep exposing old node names to SIMCA-online, even though node names have changed in the OPC HDA server. This simplifies migration to a new OPC HDA server on a SIMCA-online server with many project configurations that use the old names. See below.
- Support for tags with array data. If configured for it, the SimApi will expand array data to multiple individual tags.
- Configurable cache functionality for specified tags. This can be used as a workaround to prevent the SIMCA-online server from ending a phase or batch because of sporadic and short communication problems with the OPC server.
- Always interpret data for specified tags as strings. This can be used for SIMCA-online to be able to read batch id's that are represented with a number outside the range -16777216 to +16777216[1].

Note that this SimApi does not implement a batch node for batch data. For this you can use the Batch Context Generator within SIMCA-online (15 or later) or the Batch Table Wrapper SimApi.

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

1.1.1 Renaming the root node and using node aliases to migrate to a new HDA server for SIMCA-online

If you have a SIMCA-online server that uses an old HDA server but want to migrate to a new HDA server with, you can do as follows, in order to not have to change tag paths in all project configurations in SIMCA-online:

1. Make a backup of the existing OPC HDA SimApi XML-file.
2. In SIMCA-online Server Options:
    a. Delete the old SimApi instance
    b. Create a new SimApi instance, reusing the old name.
    c. Configure the SimApi to connect to your new server and select the nodes you like to expose. Here you see the new server's node names. Save settings and close Server Options.
3. Manually edit the XML file for the new SimApi.
    a. The root node in the XML file is your new server's name. **NewServer** in the below example. If this has changed, you should change it to the old name, and save the file. Note that if you use the graphical configuration utility, it will write the new name to the XML file, forcing you to redo this step.

        <setting key="URL" value="opchda:///**NewServer** /{F8582CF2-88FB-11D0-B850-00C0F0104305}" />

    For each node whose name has changed on the new server, you can now configure an alias that will be used in SIMCA-online instead. For example: this changes the name of **NewName** to **OldName:**

    <nodes>

        <setting key="Node_0_name" value="**NewName**" />

[1] See the SIMCA-online Technical Guide for details.

```
        <setting key="Node_0_path" value="Simulation Items.Bucket Brigade" />
        <setting key="Node_0_alias" value="OldName" />
    </nodes>
```

4. Start the SIMCA-online server and verify that node names are correct.

Note: Aliases are only for node names not for tag names. Also does not work for array data tags.

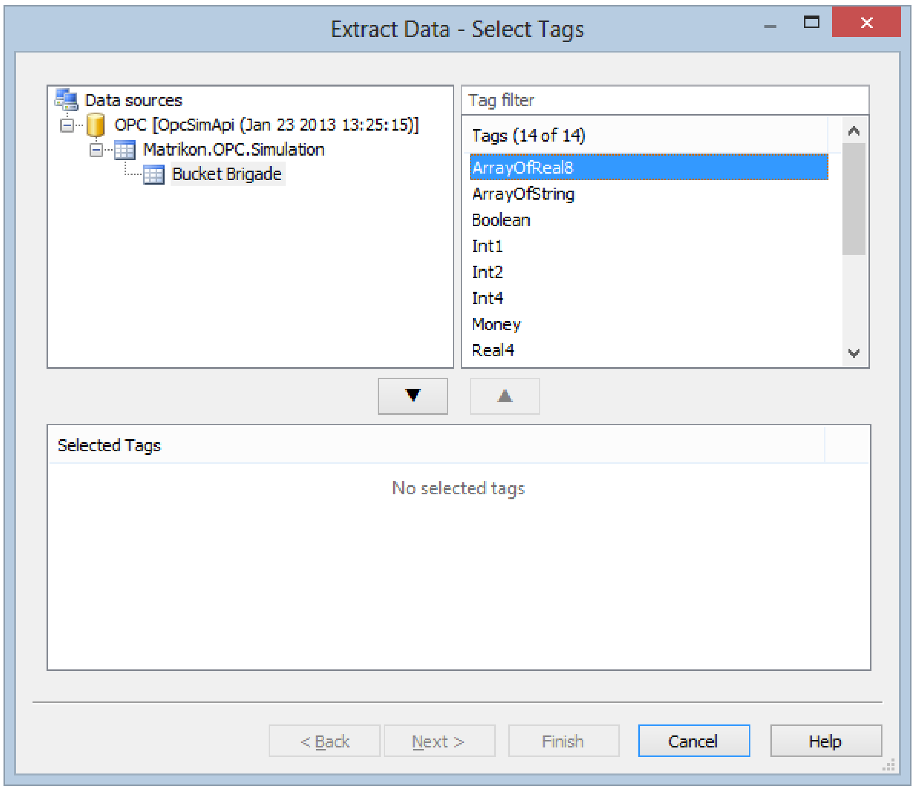
1.1.2 Support of Array data in tags

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[2] 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 0 below.

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



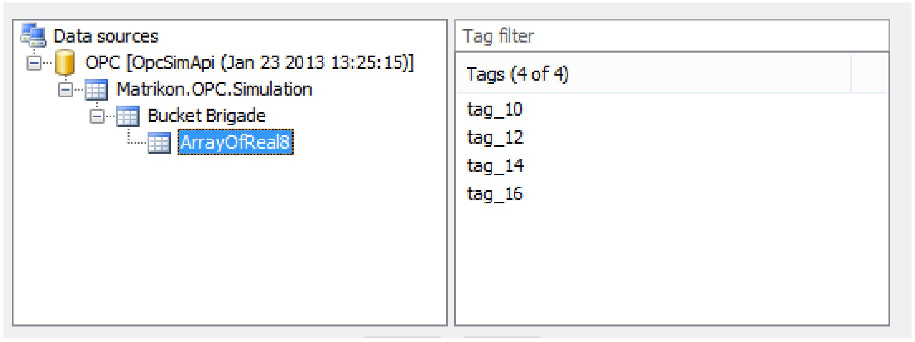
This is how the same dialog looks after configuring the tag as holding Array data: Notice that four individual tags now are listed. These don't exist in OPC but rather are generated by the SimApi for use in SIMCA-online.

[2] Only SIMCA-online can 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.

1.1.3 Cache functionality

1.1.3.1 For real-time execution

SIMCA-online will end a phase or batch if the batch id read is missing or the status is bad or uncertain. This can happen if there is a temporary communication problem with the OPC server. To work around this problem the SimApi can be configured to implement a cache of the last read that can be used instead for a configurable amount of time. If the SimApi doesn't get a valid value within this time the cache will be cleared and SIMCA-online will stop the batch.



1.1.3.2 For historical data execution, repredict in SIMCA-online

For historical data this functionality works a bit different from real-time execution. If bad status is read for a data value the SimApi will take the previous known good value within the historical time period. If the first value within that period starts with bad status, missing data (empty) will be returned until a valid value is read. So this functionality could be inconsistence, that is why we recommend that no raw data with bad status is stored in the process historian database.

Note that the configurable amount of time used for real-time execution is not applied for the historical data.

1.1.4 Interpret tag data as strings

In SIMCA-online the batch id should be a string tag or numerical tag containing integers (ranging from -16777216 to +16777216, for a larger range SIMCA-online loses precision and batch IDs won't be unique).

Use this functionality to configure the tag(s) that should be interpreted as strings, that way numbers outside the range -16777216 to +16777216 can be represented as text and no precision will be lost.

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).

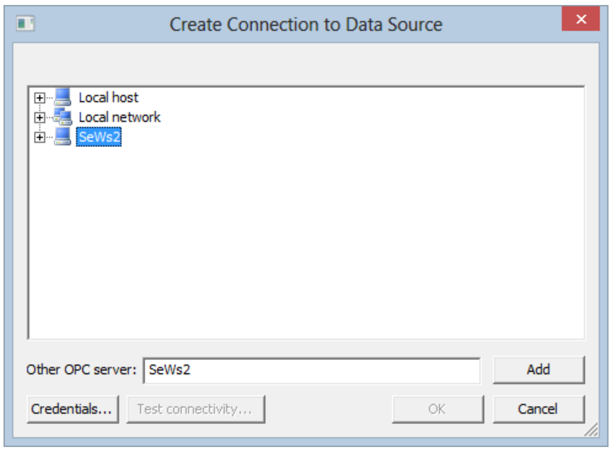
3 Installation and setup

Refer to the **SimApi Guide** located at umetrics.com/simapi for how to install and setup this SimApi.

3.1 Opc SimApi Configuration

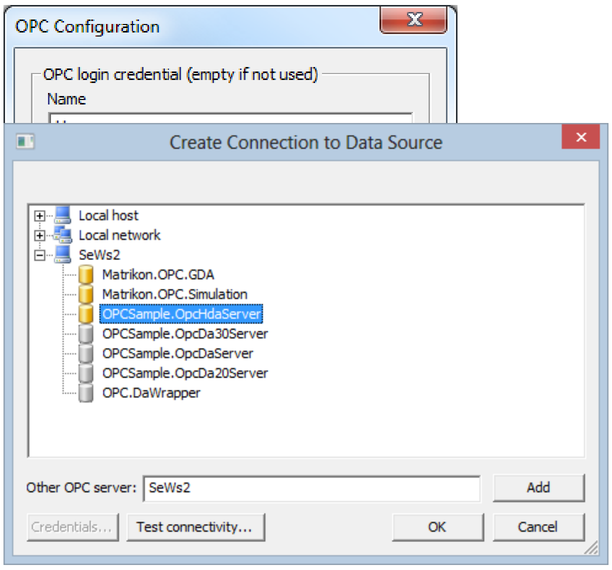
A range of settings are required for the SimApi to connect to your OPC server. All settings that you make using the graphical configuration described below are saved in the XML-file.

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:



Other OPC server: SeWs2

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.



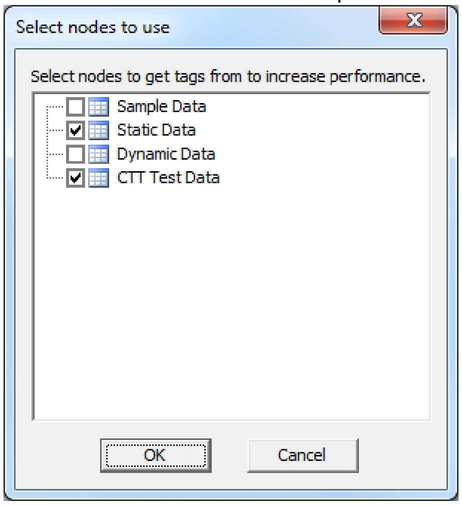
Other OPC server: SeWs2

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.


Select nodes to use

Select nodes to get tags from to increase performance.

- [ ] Sample Data
- [✓] Static Data
- [ ] Dynamic Data
- [✓] CTT Test Data

3.2 The XML Configuration File

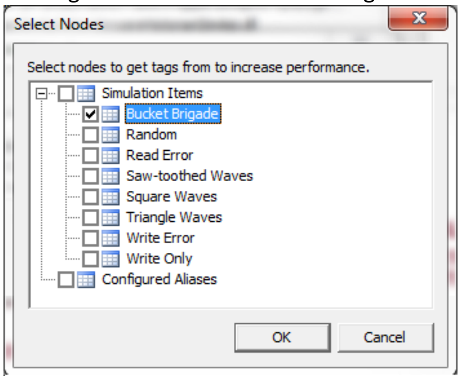
Below a sample XML file is shown. Many of the settings are made using the graphical interface above, but some has to be made by hand in the file. These settings are marked bold below, and described in the sub sections.

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

    <setting key="LogFileSize" value="1048576" />
    <setting key="LogLevel" value="4" />
    <setting key="Connection" value="10443d89df125d92" />
    <setting key="URL" value="opchda:///MyServer/{F8582CF2-88FB-11D0-B850-00C0F0104305}" />
    <setting key="UseNullTimeValues" value="0" />
    <setting key="CacheResetTime" value="60" />
    <TagsWithCache>
        <Tag name="Simulation Items.Bucket Brigade.String" />
        <Tag name="Simulation Items.Bucket Brigade.Real4" />
        <Tag name="Simulation Items.Bucket Brigade.Int1" />
    </TagsWithCache>
    <TreatAsString>
        <Tag name="Simulation Items.Bucket Brigade.Real4" />
        <Tag name="Simulation Items.Bucket Brigade.Real8" />
    </TreatAsString>
   <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>
    </arraynode>
    <nodes>
         <setting key="Node_0_name" value="NewName" />
         <setting key="Node_0_path" value="Simulation Items.Bucket Brigade" />
         <setting key="Node_0_alias" value="OldName" />
         <setting key="Node_1_name" value="NewName2" />
         <setting key="Node_1_path" value="Simulation Items.Bucket Brigade2" />
         <setting key="Node_1_alias" value="OldName2" />
    </nodes>
</settings>
```

3.2.1 XML Settings

Log file settings	**Explanation**
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. Maximum value is 4 and minimum value is 0. (0=Critical, 1=Error, 2=Warning, 3=Information, 4=Debug).
Configuration settings	
UseNullTimeValues	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" meaning null time values will be used when reading current data. `<setting key="UseNullTimeValues" value="0" />`
CacheResetTime	Specify the time interval in seconds that the cache will keep its last known good value, after that time the cache will be cleared.

TagsWithCache	Specify the tags that will use the cache functionality. The tags should contain the entire path with nodes. Here is an example:
	`<TagsWithCache>`
	`    <Tag name="Simulation Items.Bucket Brigade.String" />`
	`    <Tag name="Simulation Items.Bucket Brigade.Real4" />`
	`    <Tag name="Simulation Items.Bucket Brigade.Int1" />`
	`</TagsWithCache>`
	The tags are located in the following node:
	
	Select Nodes
	Select nodes to get tags from to increase performance.
	☐ Simulation Items
	☑ Bucket Brigade
	☐ Random
	☐ Read Error
	☐ Saw-toothed Waves
	☐ Square Waves
	☐ Triangle Waves
	☐ Write Error
	☐ Write Only
	☐ Configured Aliases
	OK Cancel
	If you want all tags to be cached use the following syntax:
	`<TagsWithCache>`
	`    <Tag name="*" />`
	`</TagsWithCache>`

TreatAsString	Specify the tags whole values should be interpreted as strings, even if they are numeric. The tags should contain the entire path with nodes in the same way as TagsWithCache are specified.
Array node settings	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.
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.
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.

Node aliases: Node_0_alias, Node_1_alias, etc	Configure aliases to use instead of the node names of the HDA server. See 1.1.1 for how to use node aliases to simplify migration to a new OPC HDA server.

3.3 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

3.4 Troubleshooting

OPC depends on DCOM and there are a range of security settings that might have to be changed in Windows for the OPC connection 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;

- 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

[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.
[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

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**.**

4 Support

This SimApi is developed by Sartorius Stedim Data Analytics. For support, please visit https://umetrics.com/support