GraphWorX32 – Using Extended Point Syntax

APPLICATIONS NOTE

October 2013

Description: Guide to using Extended Point Syntax provided by applications in GENESIS32 and BizViz suites.
OS Requirement: Win 2000, XP Pro, Server 2003, Vista, Server 2008, Windows 7
General Requirement: Basic knowledge of GENESIS32, BizViz, and GraphWorX32.

Introduction

The GENESIS32 and BizViz suites use OPC Data Access Servers and their items to read and write information to the various subsystems visualized and controlled by the applications. Typically, the values provided by the OPC items are all that is necessary; however in some cases more information is needed.

Aside from the value of a device, you may want to know information such as quality and timestamp from the OPC server. You can send specific parameters to the OPC server to obtain more detailed information about the values that are returned by the OPC Server. Sometimes, you can even combine parameters together. This could, for example, be used to request a particular property using a specific data type at a given scan rate.

The Extended Point Syntax Editor

All OPC items using the EPS (Extended Point Syntax) will be prefixed with ***tag:*** It is possible to manually enter the Extended Point Syntax for an OPC item. However for those who are not familiar with the syntax, the Extended Point Syntax Editor may be the better option. It is available from the Unified Data Browser. Once an OPC item has been selected, you can click on the EPS Editor... button to open the editor as depicted in Figure 1.

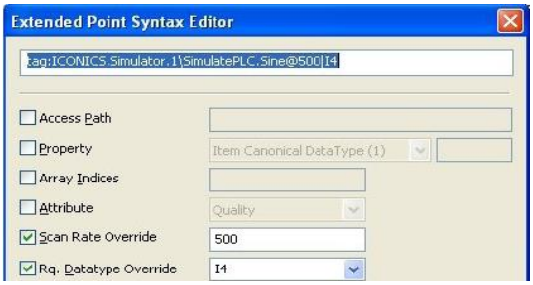

Figure 1 - Extended Point Syntax Editor

Access Path

The access path allows the OPC Server to specify different methods to access the specified OPC item. The actual parameters, when supported, are fully determined by the OPC Server vendor. To add an access path to an OPC item, use the Extended Point Syntax dialog or enter the data item using the following format: `tag:OPCServer\OPCTag::AccessPath`.

Property

In addition to the standard value provided by all OPC Servers, some OPC Servers also expose a set of additional properties, which may be requested as if they were OPC items from the ICONICS applications. When an OPC Server supports this feature, it can expose a number of common properties as defined by the OPC Foundation. In addition, it can also expose any number of custom properties. The pull down list in the Extended point Syntax Editor provides access to the common properties, but will also allow you to specify your own property IDs.

The properties provided in the list are:

- **Item Canonical Datatype (1) -** The internal data type of the OPC item in the OPC Server
- **Item Access Rights (5) -** The access rights to the OPC item as defined by the OPC Server, typical values are: 1-read and 3-read/write.
- **Server Scan Rate (6) -** The scan rate used by the OPC Server
- **EU Units (100) -** The engineering units defined for this OPC Item.
- **Item Description (101) -** The description provided for this OPC Item.
- **High EU (102) -** The high value in engineering units defined for this OPC Item.
- **Low EU (103) -** The low value in engineering units defined for this OPC Item.
- **High Instrument Range (104) -** The high value in instrument units defined for this OPC Item.
- **Low Instrument Range (105) -** The low value in instrument units defined for this OPC Item.
- **Contact Close Label (106) -** The label for the contact close state defined for this OPC Item.
- **Contact Open Label (107) -** The label for the contact open state defined for this OPC Item.
- **Item TimeZone (108) -** The time zone of the OPC Item.
- **Other -** Any other custom property for which the ID can be entered manually.

To request a property either use the Extended Point Syntax dialog or enter the data item using following format: tag:OPCServer\OPCTag-->PropertyID.

Scan Override Rate

OPC items requested by an application are added to an OPC Group and each OPC Group has its own scan rate. The default scan rate used for a group is determined by the application, for example, the display update rate in GraphWorX32 and the data

collection rate in TrendWorX32. To support multiple scan rates in a single display, we allow overriding the scan rate at the OPC item level. This will result in one or more additional OPC Groups, each with their own appropriate scan rate.

To override the scan rate for the OPC item, either use the Extended Point Syntax dialog or enter the data item using following format: `tag:OPCServer\OPCTag@RateInMilliSec`.

Requested Data Type Override

In the majority of cases, an OPC item is requested using its native data type. For this there is no special action required in any of the ICONICS applications. Nevertheless, in some cases it is useful to force a data type for the OPC item.

The Extended Point Syntax supports the following data types, using the names defined in the Variant Type enumerator:

- I1 (8 bit signed integer)
- I2 (16 bit signed integer)
- I4 (32 bit signed integer)
- I8 (64 bit signed integer)
- UI1 (8 bit unsigned integer)
- UI2 (16 bit unsigned integer)
- UI4 (32 bit unsigned integer)
- UI8 (64 bit unsigned integer)
- R4 (32 bit floating point)
- R8 (64 bit floating point)
- BOOL (boolean)
- BSTR (string)
- DATE (date/time format)
- DECIMAL (decimal)
- HRESULT
- ERROR

To change the requested data type of the OPC item, use the Extended Point Syntax dialog or enter the data item using the following format: `tag:OPCServer\OPCTag|DataType`.

Array Indices

When OPC Servers provide array data it may be necessary to extract a specific element from the array, this can be achieved using the Array Indices of the Extended Point Syntax. It supports single and multiple dimensional arrays. To extract elements from a multi-dimensional array, separate the index for each dimension with a comma (,).

To obtain specific elements from an OPC item either use the Extended Point Syntax dialog or enter the data item using the following format: `tag:OPCServer\OPCTag[index]` or `tag:OPCServer\OPCTag[index1,index2]`.

Array elements are always read-only, and cannot be written to, only the array tag as a whole (without any indices) is writeable.

Attribute

When an OPC Server sends a callback with the most up to date value to the OPC client, it also sends two other bits of information, the quality and timestamp. These attributes can be visualized using the Extended Point Syntax.

The attributes provided are:

- **Quality -** The exact quality and sub quality provided by the OPC Server. This attribute shows the value from the OPC Server, not the value internally determined in GenClient. This means that when the connection the OPC Server breaks, this attribute will still show the last received value from the OPC Server. To accommodate for this scenario use the quality() function of the expression engine instead.
- **Good quality -** A convenience value returning 0 or 1.
- **Timestamp -** The timestamp sent with the last callback.
- **Valid -** A 0 or 1 value indicating if the tag is valid.

To request a property either use the Extended Point Syntax dialog or enter the data item using the following format: `tag:OPCServer\OPCTag#attribute`.

Examples

If you want to use the EPS functionality, you can simply create your own display with a couple of PPTs in GraphWorX32 and connect to your OPC Server (e.g. ICONICS Simulator). Then GraphWorX32 allows you to display the property of the tag by using the Extended Point Syntax Editor.

Follow these steps to display the most useful property such as Quality or Timestamp:

1. Add a process point in a GraphWorX32 display.
2. Browse for the `ICONICS.Simulator.1\SimulatePLC.Sine` tag.
3. Click on the EPS Editor... button in the top-right corner of the Unified Data Browser to open the Extended Point Syntax Editor.
4. Select the **Attribute** Quality and check the EPS syntax, which should be: `tag:ICONICS.Simulator.1\SimulatePLC.Sine#quality` and click **OK** until you are back in the GraphWorX32 display.
5. Go into runtime. You should see the display similar Figure 2. Note the third line that shows the current example of returning the Quality Attribute.

Figure 2 - EPS in GraphWorX32