Unified Data Manager Configuration Guide

October 2015
An ICONICS Whitepaper
www.iconics.com

CONTENTS

1 About This Document

1.1 Scope of the Document

This document is intended to quickly guide users through the capabilities of the Unified Data Manager (UDM). It describes configurations which can be easily reproduced by first time users who have a basic understanding of OPC. The set of example configurations provide a better understanding of how projects can benefit from using the innovative capabilities. For a complete description of the product's menus and dialogs please refer to the UDM Online Help

1.2 Revision History

Version 9 – Raymond Van der Tas, August 2006
Version 9.01 – Raymond Van der Tas, July 2007
Version 9.1 – Martin Jonas, December 2007

2 Introduction

The Unified Data Manager enables the centralized management of commonly used expressions, value sets, groups, alarm subscriptions, filters, recipes, registers and event triggers across GENESIS32 and BizViz products.

The UDM allows the configuration of the following:

- Expressions
- Groups
- Values Sets
- Recipes
- Registers
- Event Triggers
- Alarm Subscriptions
- Alarm Filters

Click **Start, Programs, ICONICS Tools, Unified Data Manager** in order to start the configuration.

The UDM configuration is stored in a database. When starting the UDM, the title bar indicates the currently loaded configuration database. Configurations can be stored in MS Access, MSDE or SQL Server

When starting the configurator it should be noted that many examples are already configured. In order to create a brand new configuration select **File, New** from the menu, click **Next** and then specify a logical name for the Access configuration database such as **myDataManager.mdb**

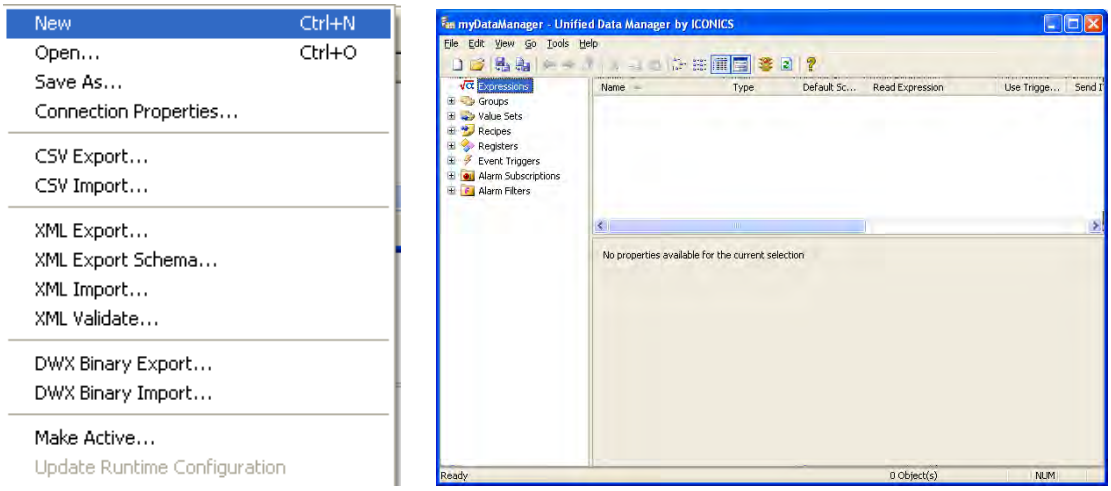

Figure 1: New Configuration

Select **File, Make Active** in order to verify that this configuration is the actively used configuration.

3 Expressions

You can now right-click **Expressions** in the left window pane and select **New, Folder**.

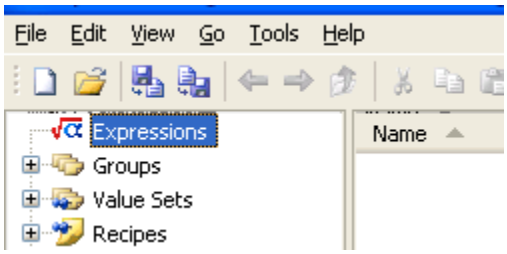

Figure 2: New Expression

3.1 Add Folders

Folders are used to logically organize the expressions. For our examples we will add a folder called **myexpression**

1. Right-click **Expressions** and select **New, Folder** and name it **myexpression**
2. Right-click the **myexpression** folder and select **New, Expression Item**.

3.2 Simple expression

As a first example we will create an expression that calculates the total level of five (5) tanks. This expression only needs a name, **TotalLevel,** and a read expression
x = {{tag1}}+{{tag2}}+{{tag3}}+{{tag4}}+{{tag5}}

OPC tag names always need to be enclosed with the curly brackets {{ }}. An example of our configuration:

Name	TotalLevel
Parameters	
Read Expression	x= {{ICONICS.Simulator.1\PLC.Tank1.DATA001}} +{{ICONICS.Simulator.1\PLC.Tank2.DATA001}} +{{ICONICS.Simulator.1\PLC.Tank3.DATA001}} +{{ICONICS.Simulator.1\PLC.Tank4.DATA001}} +{{ICONICS.Simulator.1\PLC.Tank5.DATA001}}
Write Expression	
Output Tag	

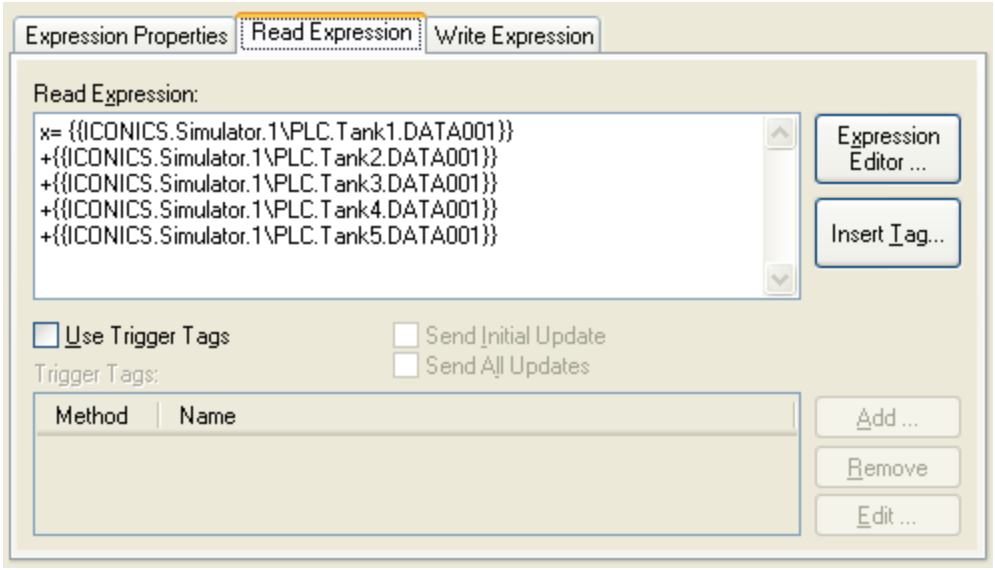

Figure 3: Read Expression

3.2.1 Using GraphWorX32

1. Start **GraphWorX32** and place a **process point** (PPT/DE) on the display
2. Click **Data Tags...** and use Unified Data Browser (UDB) to browse **Expressions**

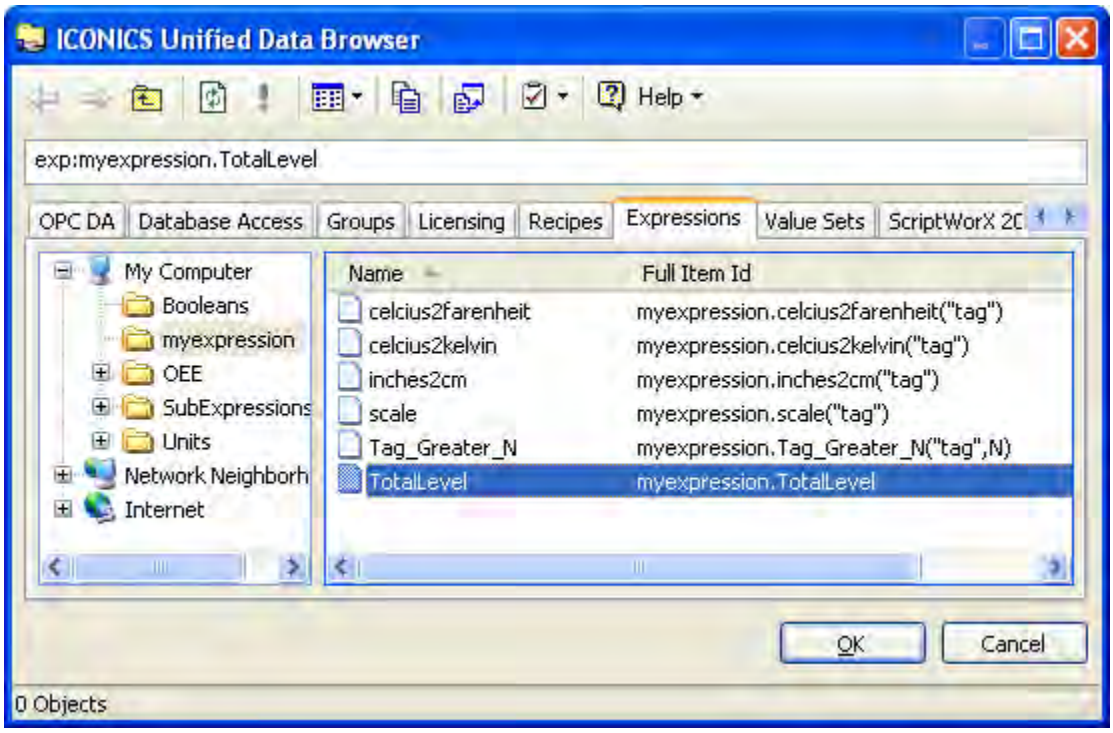

Figure 4: Unified Data Browser

3. Select the **TotalLevel** expression and click OK

   Note: The syntax of the tag is exp:myexpression.TotalLevel

3.3 Scaling values

Typically scaling device values to engineering units is performed by the OPC Server itself. You could also use the ICONICS Language Configurator to perform device values to engineering unit conversion. However, for this example we want to demonstrate the scaling with the UDM. In this example we will configure a SCALE expression which takes a tag name as the parameter and divides the OPC value by 10.

Name	Scale
Parameters	Tag
Read Expression	x= {{<<tag>>}} / 10
Write Expression	x= {{input}} * 10
Output Tag	<<tag>>

The tag parameter is of the STRING type since it will need to accept the OPC Tag name.

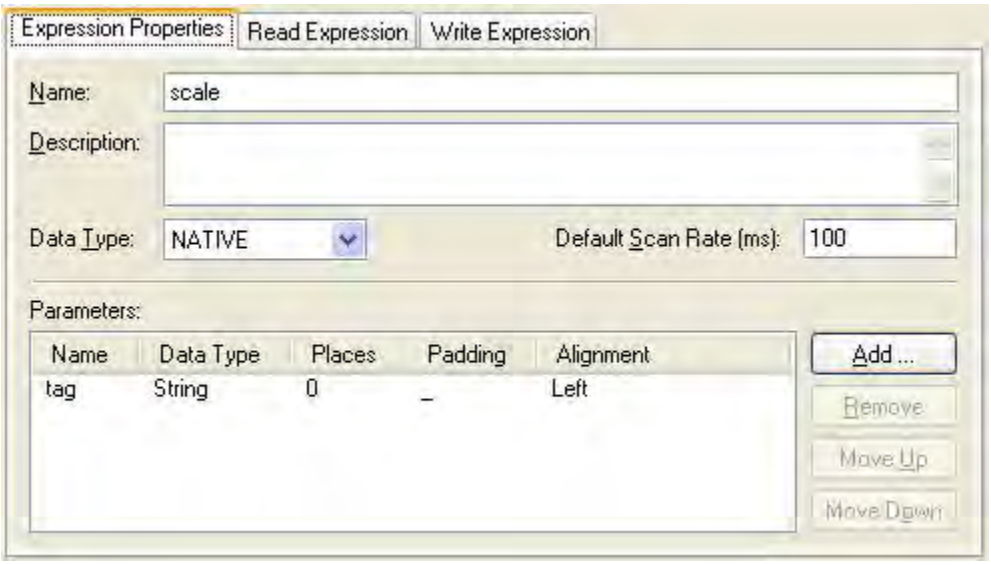

Figure 5: Expression Properties with Parameter

The read expression should enclose the tag parameter with alias characters << >> to indicate that it will be assigned a real alias value by the client. The curly brackets {{ }} is the standard method in the Expression engine to indicate that the string is to be treated as an OPC tag.

Figure 6: Read Expression – Use of a Parameter

The write expression specifies {{input}}. This is an internal parameter of which the value is provided by GraphWorX32 when a value is written.

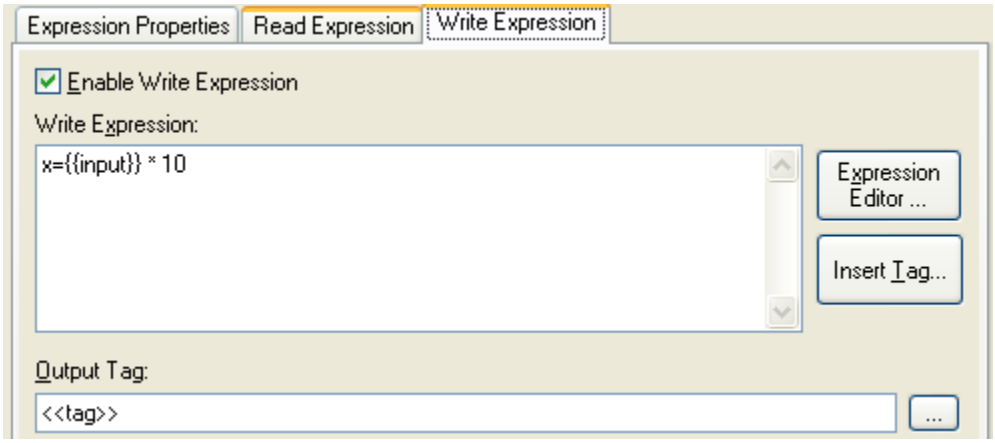

Figure 7: Write Expression configured

3.3.1 Using GraphWorX32

1. Start **GraphWorX32** and place a **process point** on the display
2. Click on **Data Tags...** and use the UDB to browse the Expressions.
3. Select **scale** and replace "tag" with a real OPC tag name. For instance "ICONICS.Simulator.1\SimulatePLC.PumpSpeed" The complete syntax is **exp:myexpression.scale("ICONICS.Simulator.1\SimulatePLC .PumpSpeed")**
4. Click **OK** and start **Runtime**.

3.4 Converting Temperatures

The next example demonstrates how we configure a Celsius to Fahrenheit expression.

Name	celsius2fahrenheit
Parameters	Tag
Read Expression	x= ({{{<<tag>>}}*1.8)+32
Write Expression	x= ({{{input}}-32)/1.8
Output tag	<<tag>>

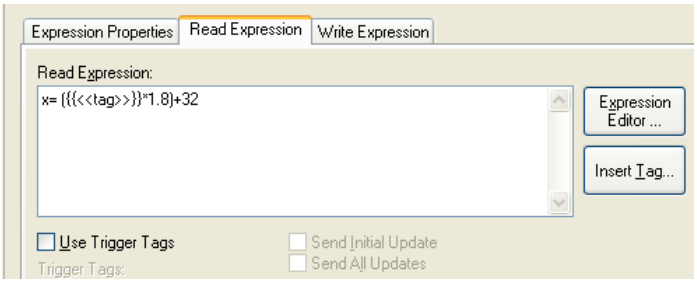

Figure 8: Celsius to Fahrenheit Expression - Read Expression tab

3.5 Using multiple parameters

Also more complicated expressions can be created which take multiple parameters in its calculations. The next example checks if the tag value is greater than the number parameter N.

Name	Tag_Greater_N
Parameters	tag (string) N (number)
Read Expression	x= if({{{<<tag>>}}>{{<<N>>}},1,0)
Write Expression	
Output tag	

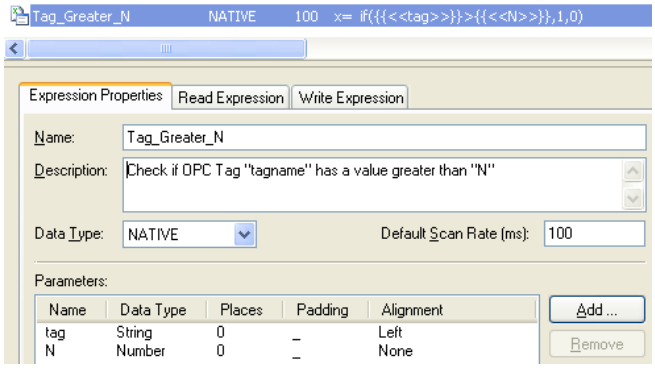

Figure 9: Expression properties - Multiple Parameters

3.6 OEE Expressions

Overall **E**quipment **E**fficiency is a calculation to analyze a machine or process in real-time. OEE is calculated by three major components: Quality, Availability and Performance. Placing OEE expressions in the Unified Data Manager allows us to use the expressions when needed without having to worry about the calculations itself.

The formula: **OEE = Quality * Availability * Performance %**

The next example display in GraphWorX32 shows an OEE panel where the color coding Red, Yellow and Green would indicate if company standards are met.

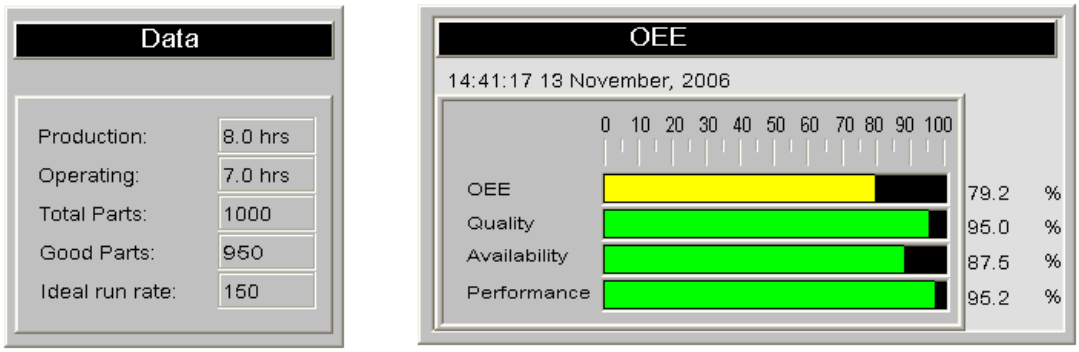

Figure 10: OEE indicators

Before creating OEE panels in GraphWorX32, we would first add a folder called **OEE** to the Unified Data Manager.

1. Right-click **Expressions** and select **New, Folder** and name it **OEE**
2. Right-click the **OEE** folder and select **New, Expression Item**.
3. The first expression to add is **Quality** which is a ratio between good produced parts and the total amount of produced parts.

Name	**Quality**
Parameters	good_parts (string) total_parts (string)
Read Expression	x= {{<<good_parts>>}}/{{<<total_parts>>}}
Write Expression	
Output tag	

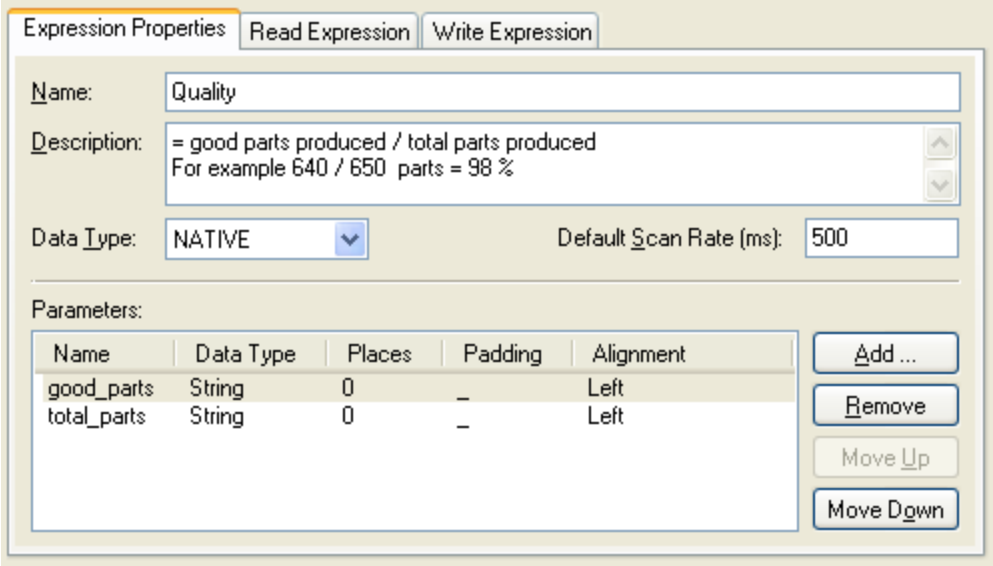

Figure 11: Expression Properties - Parameters

A Process Point in GraphWorX32 should then have the following syntax to present the value as a percentage:

x= {{exp:OEE.Quality("<<good_parts>>","<<total_parts>>")}} *100

The parameters are local GraphWorX32 aliases and can be assigned to the OPC Tags providing the true number of good_parts and total_parts.

4. The next expression to add is **Availability,** the ratio between the time a machine or equipment was actually operated (operating time) and the time the equipment theoretically could have been used (production time).

Name	Availability
Parameters	operating_t (string) production_t (string)
Read Expression	x= {{<<operating_t>>}}/{{<<production_t>>}}
Write Expression	
Output tag	

A Process Point in GraphWorX32 should then have the following syntax to present the value as a percentage:

x= {{exp:OEE.Availability("<<operating_t>>","<<production_t>>")}} *100

The parameters are local GraphWorX32 aliases and can be assigned to the OPC Tags providing the true operating time and production time.

5. The third expression to add is **Performance** which is the amount of time required to produce a product divided by the ideal time to produce the product.

Name	Performance
Parameters	operating_t (string) total_parts (string) ideal_run_rt (string)
Read Expression	x= ({{<<total_parts>>}}/{{<<operating_t>>}})/{{<<ideal_run_rt>>}}
Write Expression	
Output tag	

A Process Point in GraphWorX32 should then have the following syntax to present the value as a percentage:

x= {{exp:OEE.Performance("<<operating_t>>","<<total_parts>>", "<<ideal_run_rt>>")}}*100

The parameters are local GraphWorX32 aliases and can be assigned to the OPC Tags providing the true operating_time, total_parts and ideal_run_rate.

6. The last expression to add is **OEE = Quality * Availability * Performance %**

Name	OEE
Parameters	operating_t (string) production_t (string) goor_parts (string) total_parts (string) ideal_run_rt (string)
Read Expression	x= ({{<<total_parts>>}}/{{<<operating_t>>}})/{{<<ideal_run_rt>>}} * ({{<<good_parts>>}}/{{<<total_parts>>}}) * ({{<<operating_t>>}}/{{<<production_t>>}})
Write Expression	
Output tag	

A Process Point in GraphWorX32 should then have the following syntax to present the value as a percentage:

x= {{exp:OEE.OEE("<<operating_t>>","<<total_parts>>", "<<ideal_run_rt>>","<<production_t>>","<<good_parts>>")}} * 100

The parameters are local GraphWorX32 aliases and can be assigned to the OPC Tags providing the true operating_time, production time, total_parts, good parts, ideal_run_rate.

4 Groups

Groups can be used to distribute a value to a group of tags. Groups within groups are also allowed.

4.1 Configure a Group of tags

Folders can be used to logically organize the groups. For our examples we will add a folder called **setpoints.**

1. Right-click **Groups** in the left window pane, select **New, Folder** and name it **setpoints**
2. Right-click the **setpoints** folder and select **New, Group Item**.
3. Specify the group name: **set** and add a couple of tags as shown below.

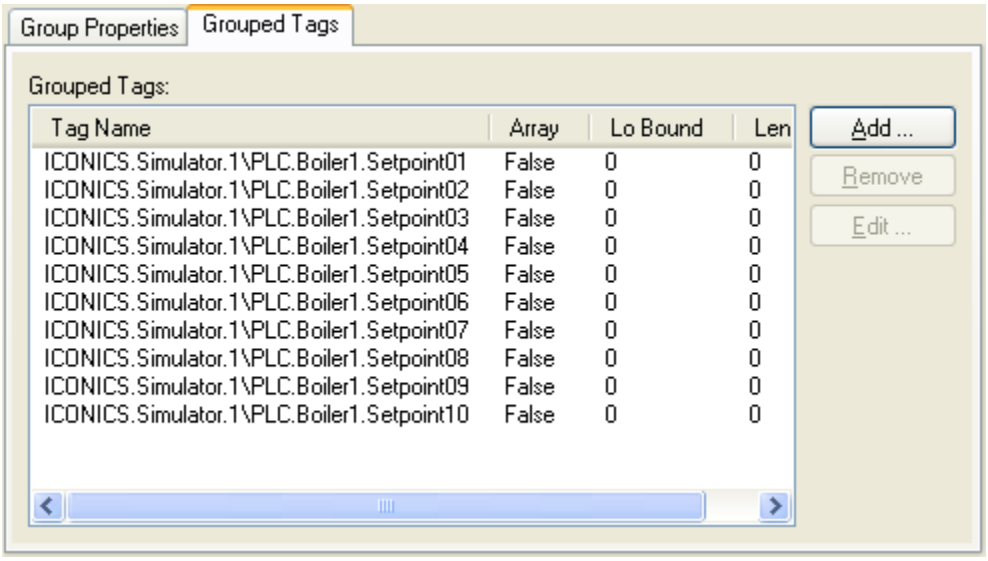

Figure 12: Grouped Tags

In the OPC Client you can now use the ICONICS Unified Data Browser (UDB) and select the group. Writing a value to this tag will write the value to all tags in the group.

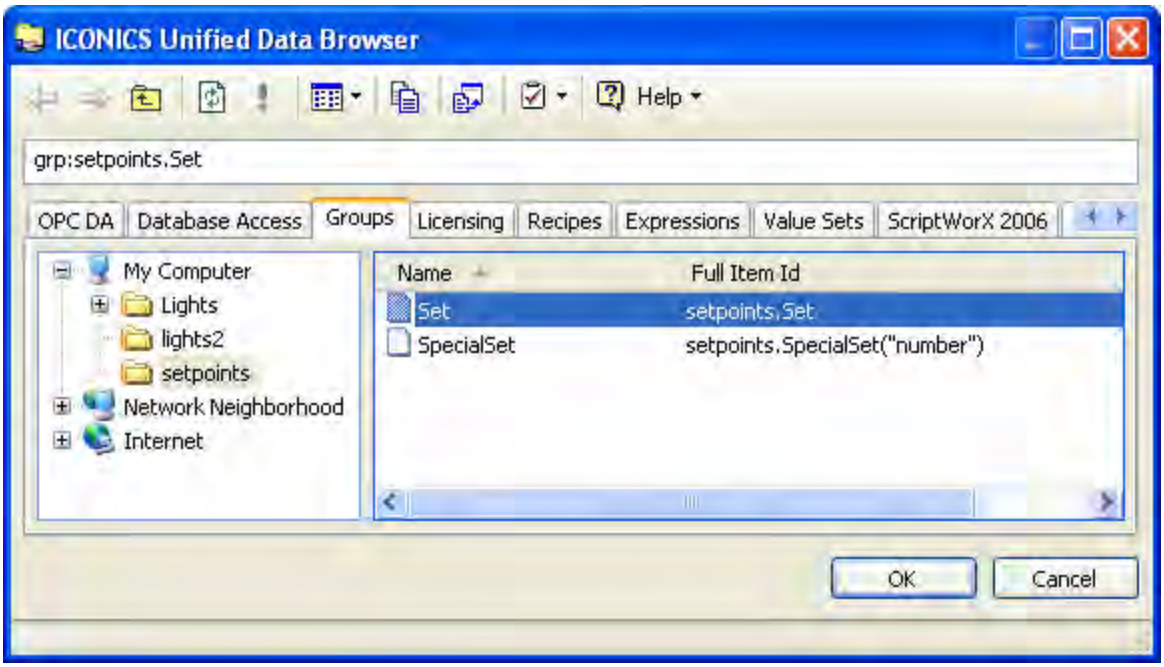

Figure 13: Selecting Group Process point in ICONICS Unified Data Browser (UDB)

4.2 Using a parameterized group of tags

Imagine you have multiple groups where a parameter could be used to identify the group

1. Right-click the **setpoints** folder and select **New, Group Item**.
2. Specify the group name: **SpecialSet** and add a parameter called **number** as shown below.

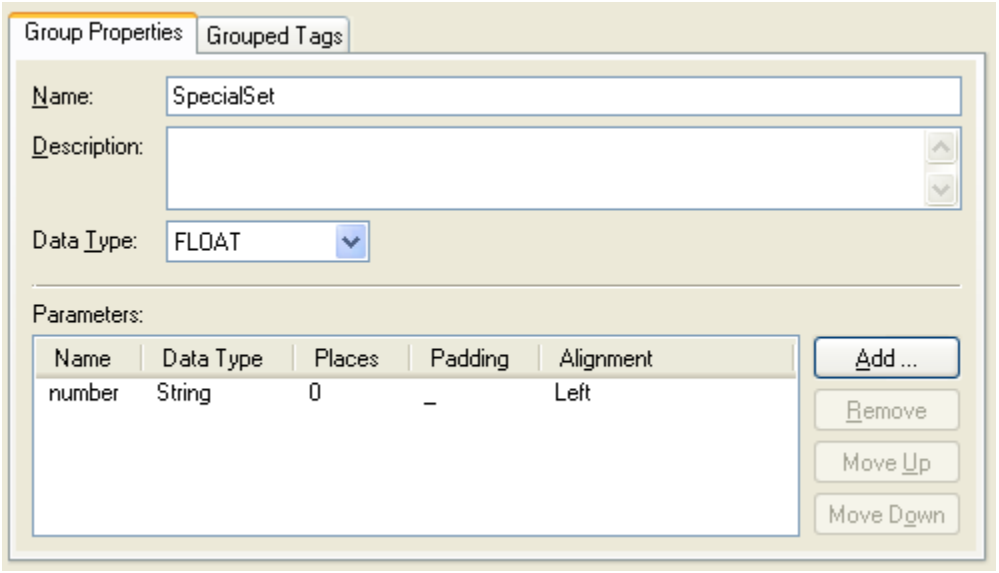

Figure 14:Group Properties - Using parameter

3. Specify the Grouped Tags to use this parameter in the tag names. The parameter has to use the **<< >>** alias identifier.

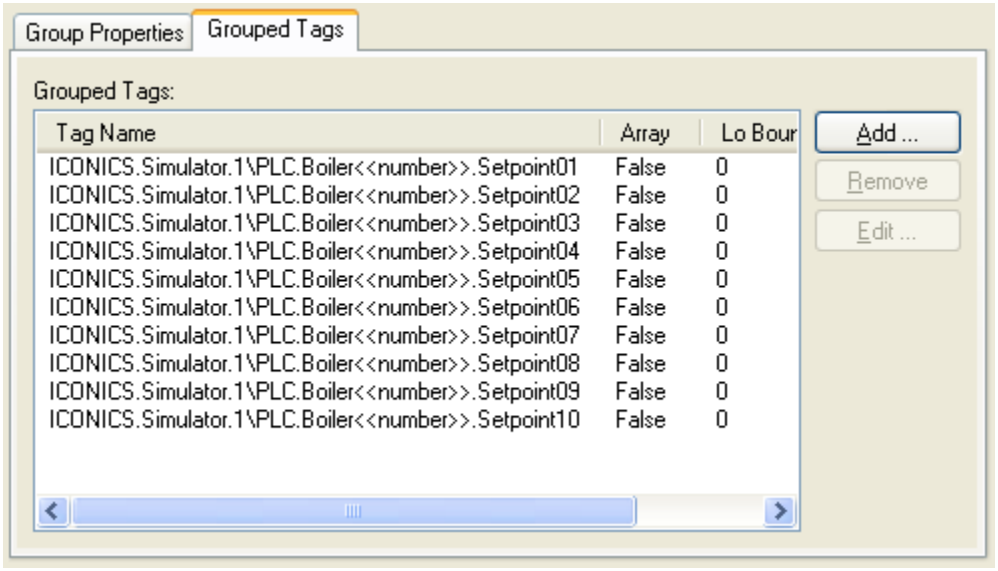

Figure 15: Grouped Tags with use of a Parameter

When using this special group in GraphWorX32, you can substitute **number** with the number you want to address. For instance a tag in GraphworX32 which connects to **grp:setpoints.SpecialSet("3")** will distribute the written value to the grouped tags in **Boiler3**.

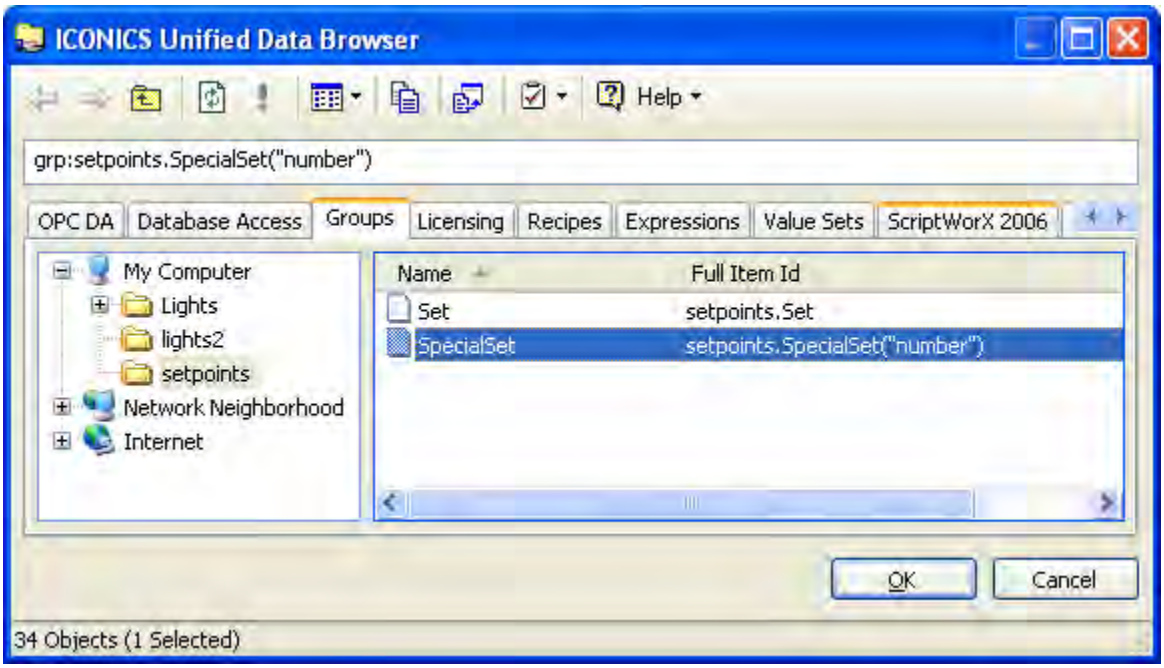

Figure 16: Selecting a Group Process Point with Parameter

4.3 Grouped groups

In a project you may have several groups. Sometimes you would like to create a group that contains other groups. As an example we will create two groups. One called **even numbers** and the other called **odd numbers**

The even numbers group contains these tags:

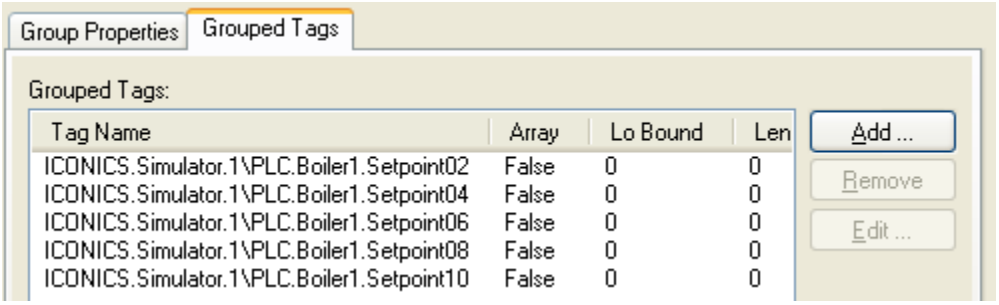

Figure 17: Even Numbers Group

The odd numbers group contains these tags:

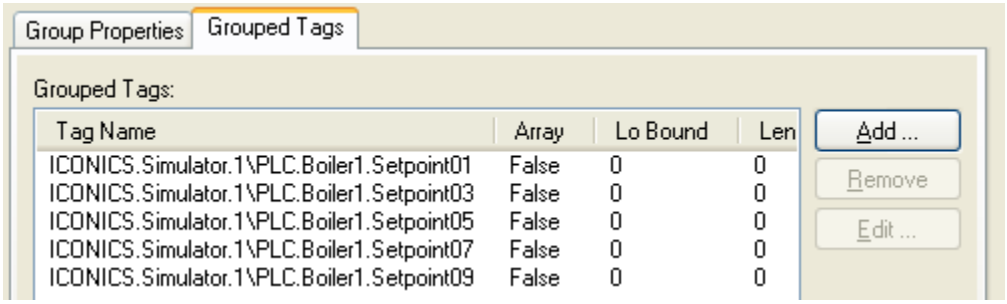

Figure 18: Odd Numbers Group

The **odd and even numbers** group contains the following tags:

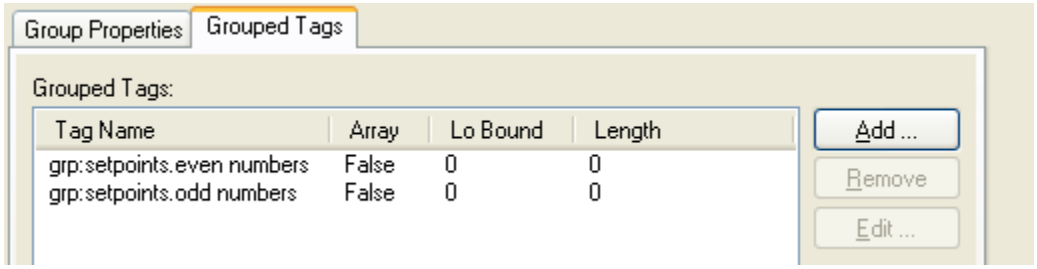

Figure 19: Two Groups in another Group

When using this group in GraphWorX32 **grp:setpoints.even and odd** will distribute the written value to the tags of the two groups.

5 Value Sets

Value sets allows you to write a set of values to a group of tags. This is similar to the concept of groups, but with more flexibility.

5.1 Value Set Configuration

In this example we will create a value set which can be used to set values to one of our boilers. Each boiler contains the same set of tags and only the boiler number needs to be known to point at the right set of tags.

Value Set Folders can be used to logically organize the value sets. For our examples we will add a folder called **SET**

To configure a Value Set:

1. Right-click **Value Sets** in the left window pane, select **New, Folder** and name it **SET**
2. Right-click the **SET** folder and select **New, Value Set Item**.
3. Configure the Value Set **SetBoiler** as shown below and add a parameter which you rename to **number**

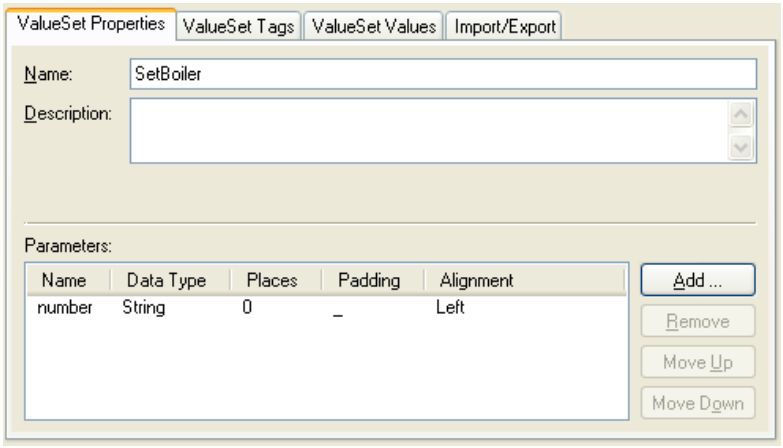

Figure 20: Value Set Properties with a Parameter

4. Select the Value Set Tags tab and add the list of tags from BOILER1.

5. Edit the tag for each tagname in the list to use the <<number>> parameter to identify any of our multiple boilers.

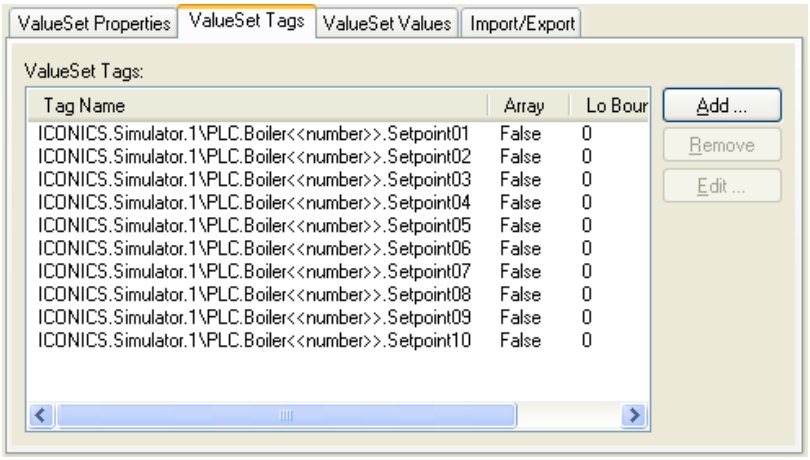

Figure 21: Value Set Tags with a Parameter

6. Then select the Value Set Values tab, click the **new** button and specify a number for the value set for instance **1**. For each tag in the list you will need to select it and click the **Set as Value** button.

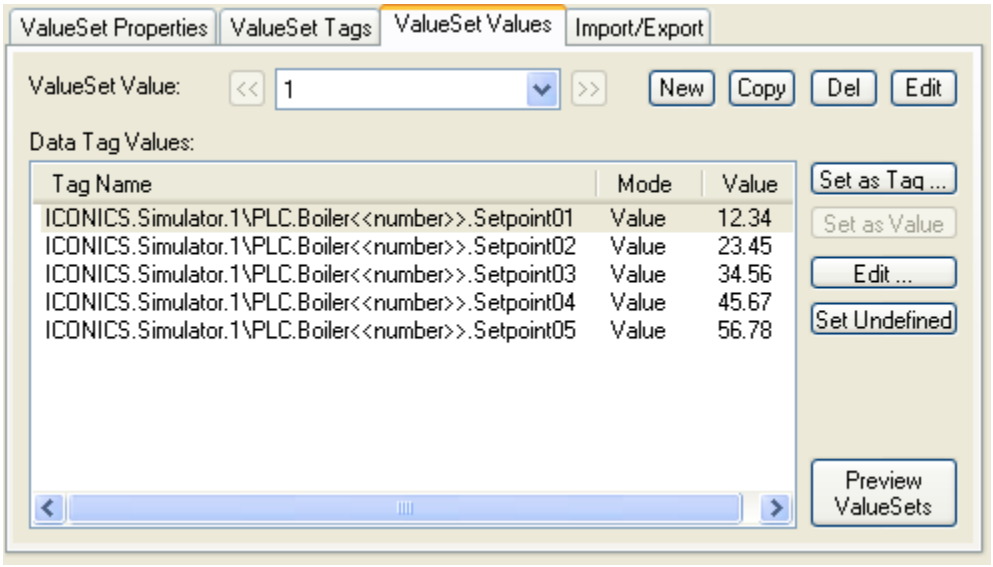

Figure 22: Defining Value Set Values

7. Specify a Value for instance 12.34, select the Data Type for instance VT_R8 and click **OK**.

When all tags in the list have a proper value assigned, you can click the COPY button to duplicate this value. When clicking COPY you will be asked to enter a unique number for the value set.

In GraphWorX32 you would use the Unified Data Browser to select the Value Set and to replace “number” with the desired boiler number, for instance “1”

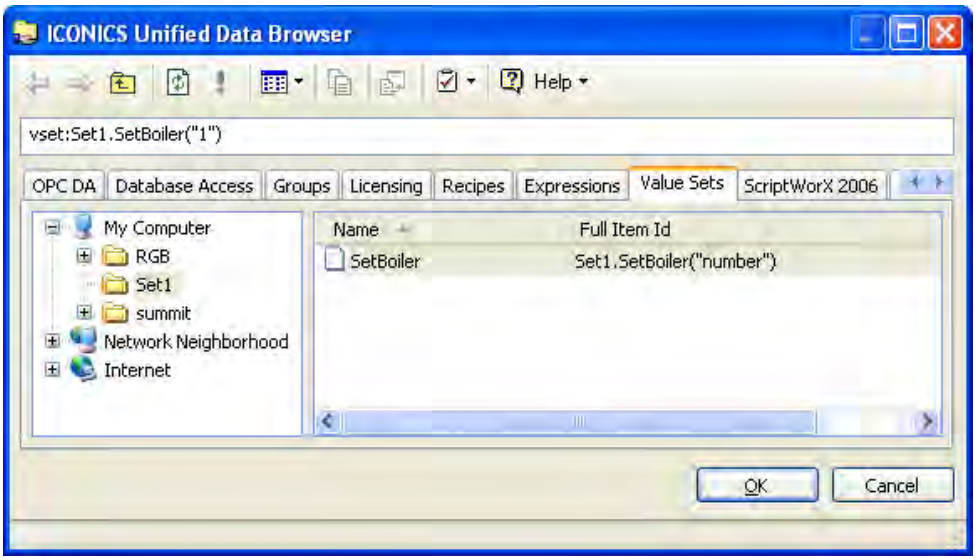

Figure 23: Selecting Value Sets in Unified Data Browser

If you have a Process Point/Data Entry on the Display, you can write a value to this tag. The value “1” will cause the values for value set 1 to be downloaded to the tags. The value “2” will cause the values for value set 2 to be downloaded to the tags, and so on.

> **Note:** If you want to avoid remembering your value sets IDs you can use the State Field feature of GraphWorX23 Process point to substitute them with comprehensible names.

5.2 Advanced Value Set Configuration

Rather than having value set values statically assigned, we can also have the value to be set directed by OPC tags.

In the previous example you created two or more sets of values. Simply edit one of the value sets and change “set as value” into **Set as Tag**

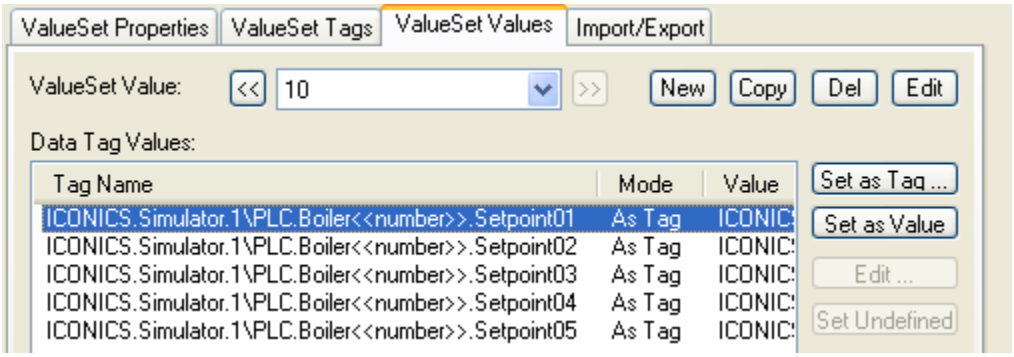

Figure 24: Setting Value Set Values as Tags

6 Recipes

Recipes allow you to create a predefined list of parameters (ingredients) connect these to OPC tags and to add recipe items for each flavor of your product you plan to create. The recipe system is first configured in the Unified data manager and can then be used on PDAs (Pocket GENESIS) or in GENESIS32 using the Tree Control ActiveX and the Recipe Control ActiveX. You can also create your own controls based on Recipe Management interface as it is demonstrated in GraphWorX32 Symbol Library.

6.1 Recipe Configuration

In this example we will create a recipe for a paint line. The Name Recipe of the recipe will be called **Paint Line**

To configure a Recipe:

1. Right-click **Recipes** in the left window pane, select **New, Folder** and name it **PAINT**
2. Right-click the **PAINT** folder and select **New, Recipe**.

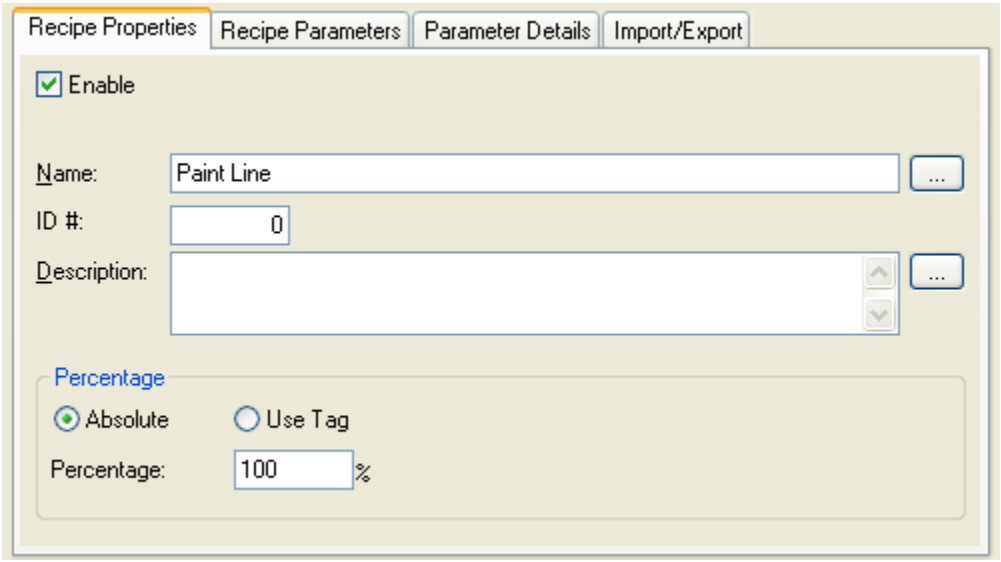

Figure 25: Recipe Properties

3. The Recipe Parameters tab needs to be assigned a list of OPC tags for the paint line:

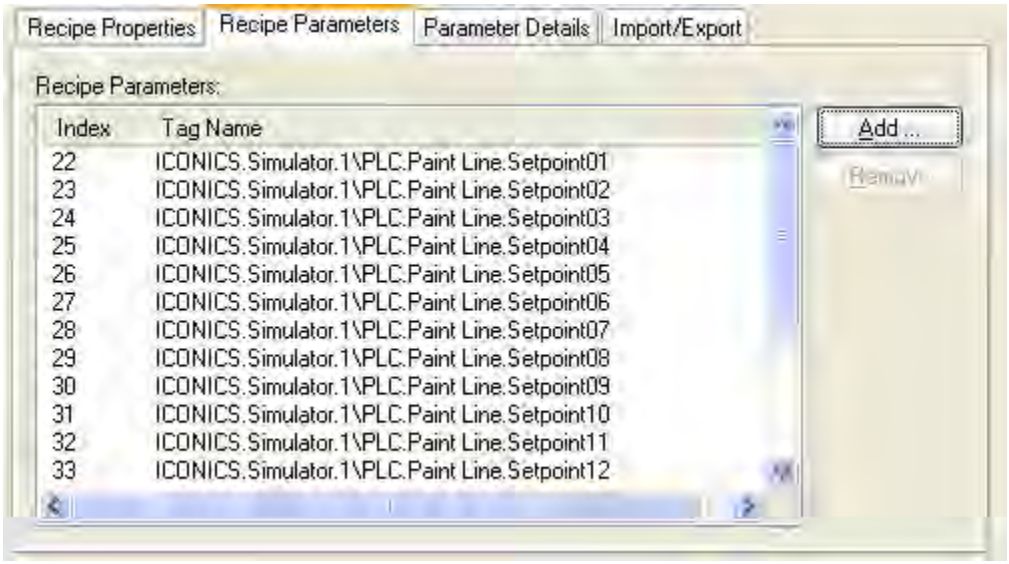

Figure 26: Recipe Parameters

The Parameter Details tab allows us to give a logical "ingredient" names to each of the OPC tags. In the next example we will simply name them **a, b, c,…**

After you entered a name for the ingredient you click the **>** button until all parameters have been assigned a name.

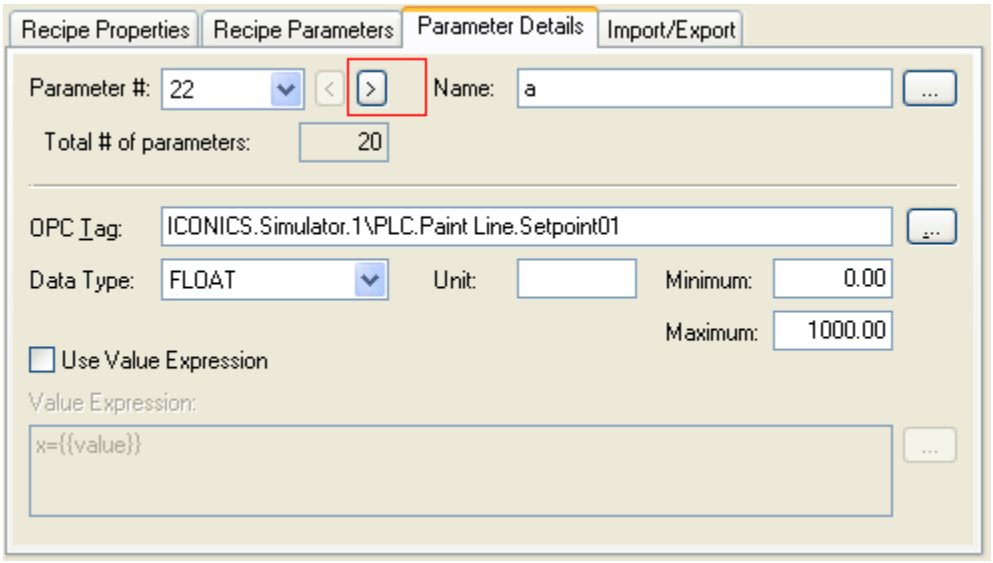

Figure 27: Configuring Parameters

6.2 Recipe Item Configuration

Once a recipe is created, you can add one or more recipe items to the Recipe.

To add a recipe item, right-click the **Paint Line** recipe and select **New, Recipe Item** You can name the recipe item **Havana White** and click the **Recipe Items Values** tab in which you can assign the proper values for each of the ingredients:

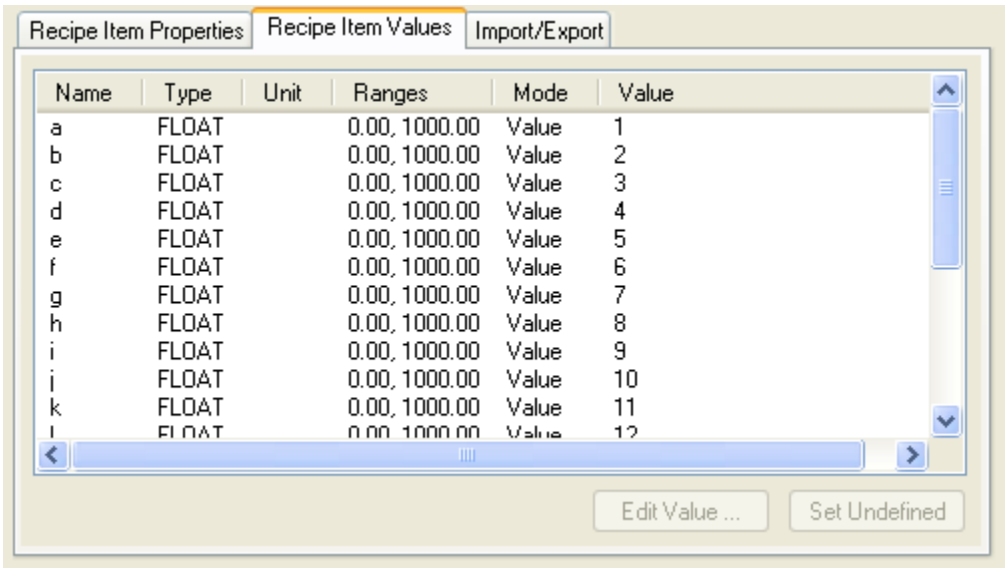

Figure 28: Defining Parameter Values of Recipe Item

You can create as many Recipe Item as you want in Unified Data Manager, however, you will only need to add one recipe item since GraphWorX32 has recipe controls and pick actions to allow the operator to copy, edit and download recipes.

7 Registers

In general, registers are named variables that all GENESIS32 clients and OPC clients can use. If you have used registers inside DataWorX32, then the features of UDM registers are a subset of the features available for registers in DataWorX32.

The most significant difference between UDM and DataWorX32 registers is that UDM registers are **not real OPC tags**, and UDM registers cannot be accessed without GenClient and the appropriate GenBroker plug-in. That means that no 3rd party clients may access them. The main differences between UDM and DataWorX32 registers are noted in the table below.

	UDM Registers	DataWorX32 Registers
Available through OPC	No	Yes
Available through Automation	No	Yes
Switch register	No	Yes
Alias register	No	Yes
Redundancy register	No	Yes
Online changes	Automatic	Manual
Data Propagation	Limited, can be based on tag	Limited, can be delayed
Ranges	No	Yes

7.1 Bridging registers

The default setting of a new register is for **Bridging** – reading and propagating an input tag to an output tag where the two tags are not the same.

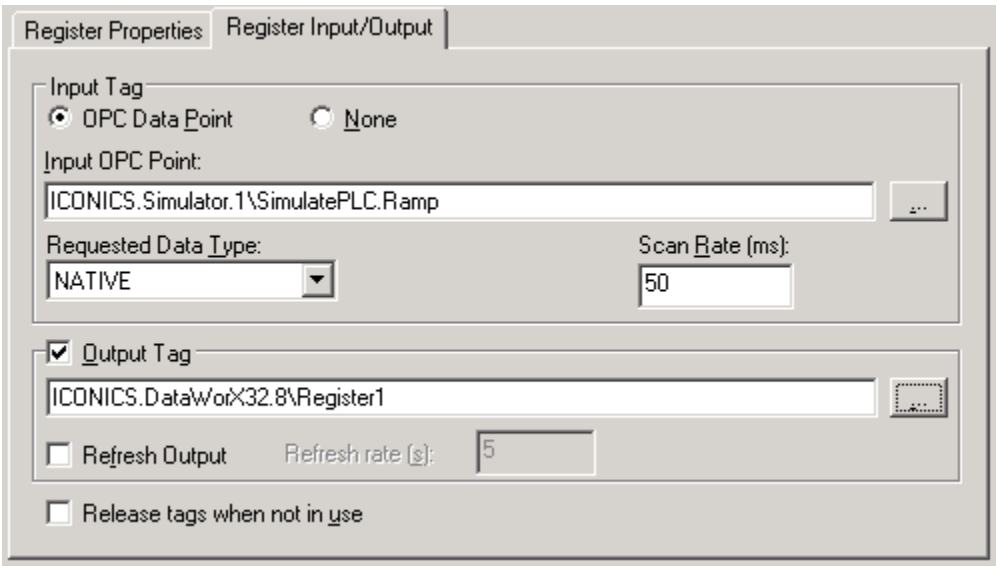

Figure 29: Bridging Register Setting

For the purpose of bridging you only need to select **Input OPC Point** and **Output Tag.** Both can be performed by clicking on the appropriate **...** button, which opens ICONICS Unified Data Browser (UDB).

> **Note:** You must not check **Release tags when not in use** checkbox otherwise bridging will not work. This feature disconnects the input as well as the output tag when no client is connected to the register.

7.2 Aggregation registers

Aggregation registers are those where Input OPC Point is the same as Output Tag. It is usually used when you have more OPC servers but you want all clients connect to only one server.

For this use you need to **disable** input updates propagation under Register properties tab.

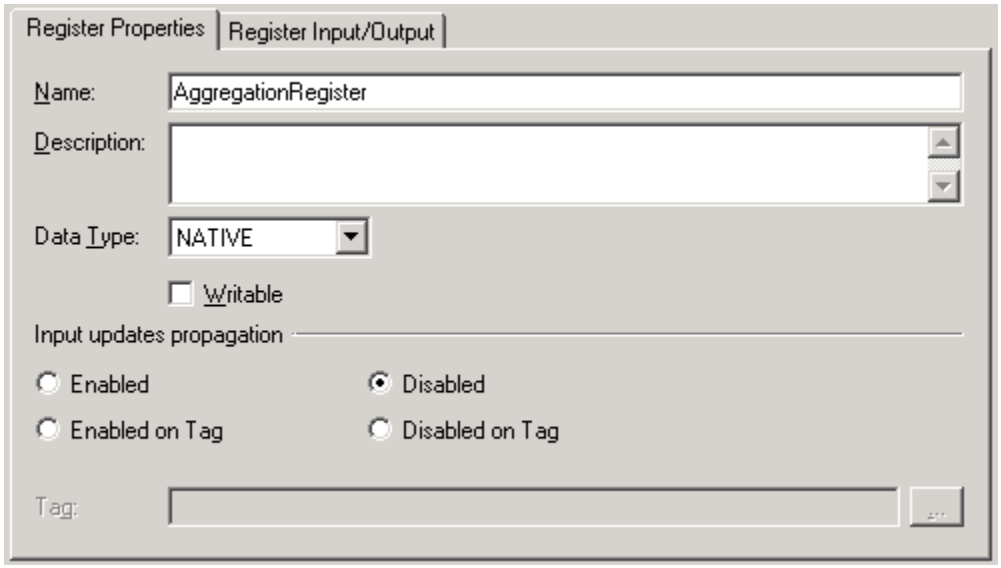

Figure 30: Aggregation Register Properties

7.3 Input registers

If not combined with an output tag, registers usually serve as holding registers or “read only” registers connected to another OPC point.

In order to create an Input-only register you need to uncheck Output tag checkbox

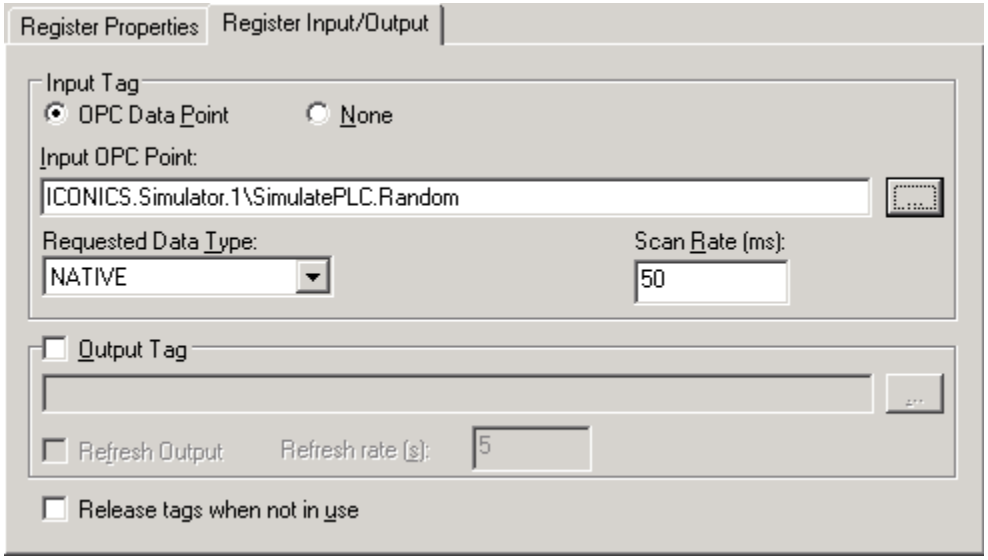

Figure 31: Input OPC Data Point Setting

Now you have a register that you can use to read an OPC point defined as **Input OPC point**

If you want to create a holding register, to store values you entered, select the **None** radio button. You can decide to define an **Initial value**.

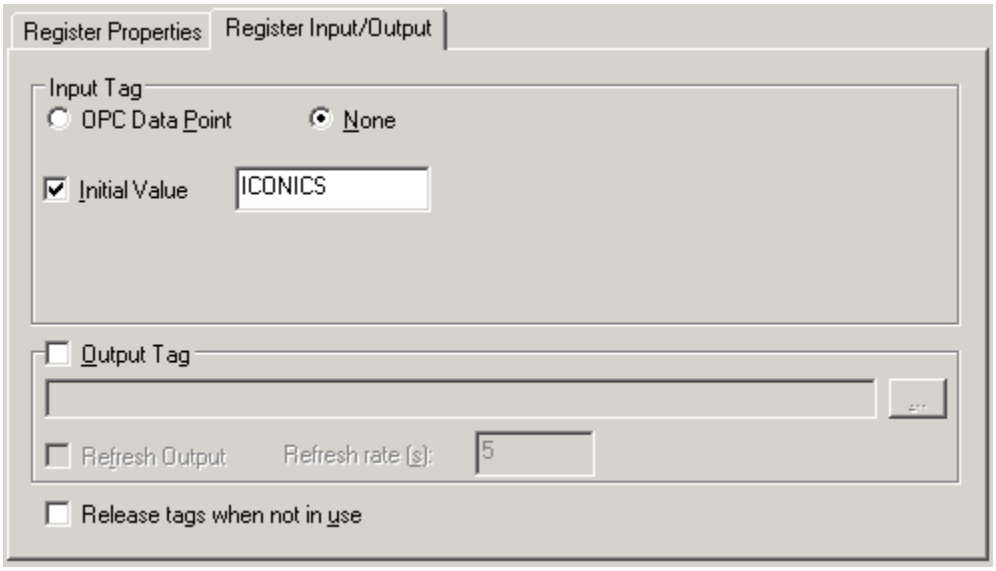

Figure 32: Input Holding Register Setting

It is not recommended to make a register writeable that has an OPC Data Point for input but no output tag. If you enter a value in such register it is stored in the register but the input OPC Data Point value does not change. The entered value is propagated nowhere because there is no tag defined as the register's output.

7.4 Using UDM registers

You may have noticed that there is a traffic light icon in the Unified Data Manager's toolbar. All UDM items but registers are available and work regardless the status of this light. If you want the registers to work you need to start this traffic light. When the light is green all your registers are enabled, but when the light is red your registers will not show data, bridge, or aggregate.

You can use the UDM registers in the same way as any other OPC points in your clients with the limitation described at the beginning of this document. When browsing for your registers in Unified Data Browser you should look under **UDM Data** tab.

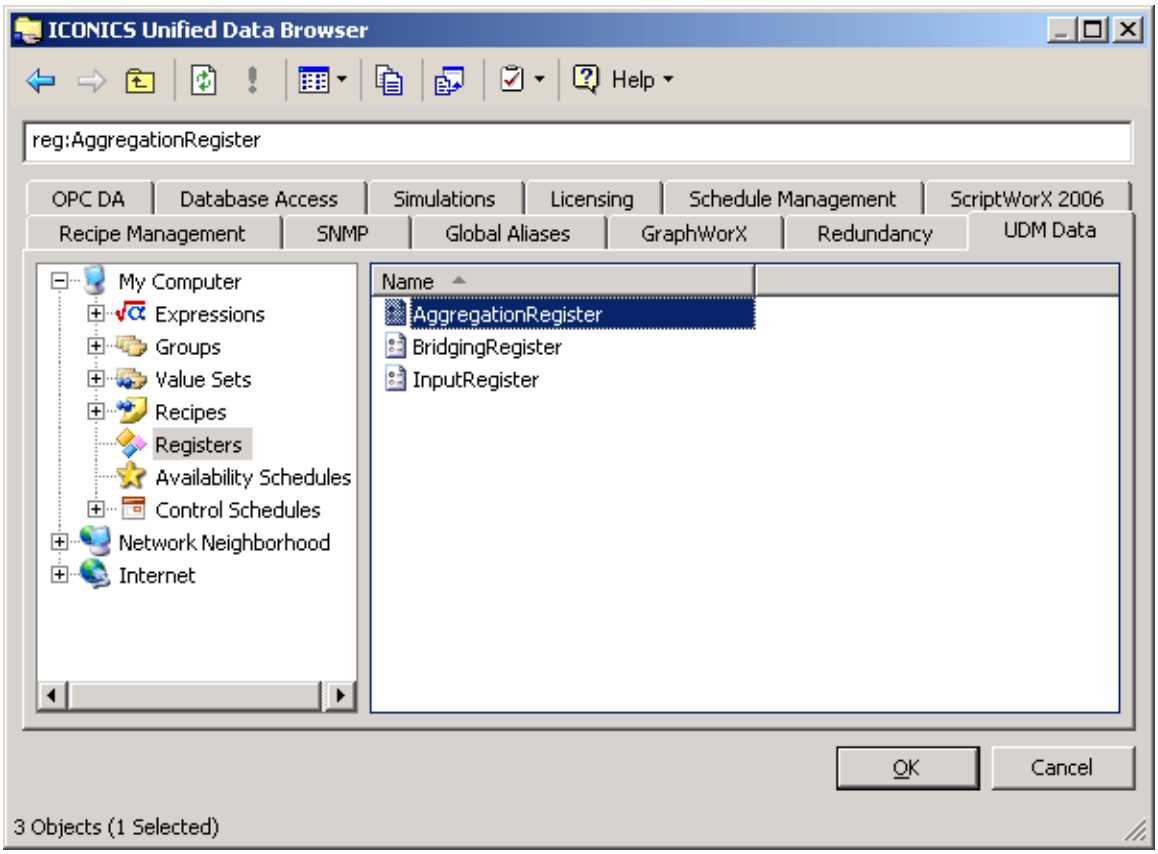

Figure 33: Browsing Registers in UDB

8 Global Alarm Subscriptions and Filters

8.1 Alarm Subscriptions

Subscriptions are used to inform the OPC Alarms/Events Server which alarms we want to be informed about. The global subscriptions can be used by alarm clients such as the AlarmWorX32 Viewer and AlarmWorX32 Multimedia.

Folders can be used to logically organize the Subscriptions. For our examples we will add a folder called **myalarms**.

1. Right-click **Alarm Subscriptions** in the left window pane, select **New, Folder** and name it **Summit**
2. Right-click the **myalarms** folder and select **New, Alarm Subscription Item** and configure the subscription to connect to the AlarmWorX32 Alarm Server. The next image shows an example where we added also a subscription to two other OPC A&E Servers.

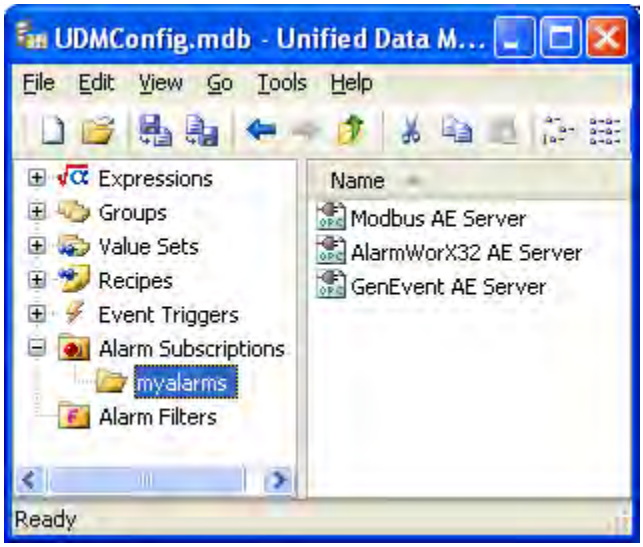

Figure 34: Alarm Subscriptions

8.2 Alarm Filters

Filters are used to present a subset of the alarms provided by the alarm server. The filters are used by alarm clients such as the AlarmWorX32 Viewer and AlarmWorX32 Multimedia.

Folders can be used to logically organize the Filters. For our examples we will add a folder called **myfilter.**

1. Right-click **Alarm Filters** in the left window pane, select **New, Folder** and name it **myfilter**
2. Right-click the **myfilter** folder and select **New, Alarm Filter Item**.

The next configuration demonstrates shows how we filter “A-Alarms” (alarm tags which start with the letter A).

Name	A Alarms
Filter String	X= (like({{Source}},"A*",0))

This configuration demonstrates shows how we filter alarm tags which have a priority higher than 500 but lower than 600.

Name	500-600 priority
Filter String	x=({{Severity}} >=500) && ({{Severity}} <600)

8.2.1 Using Alarm Filters in the Alarm Client

To use the configured filters, you can add an **ICONICS AlarmWorX32 Viewer** to a display and select **Filter** from the properties dialog.

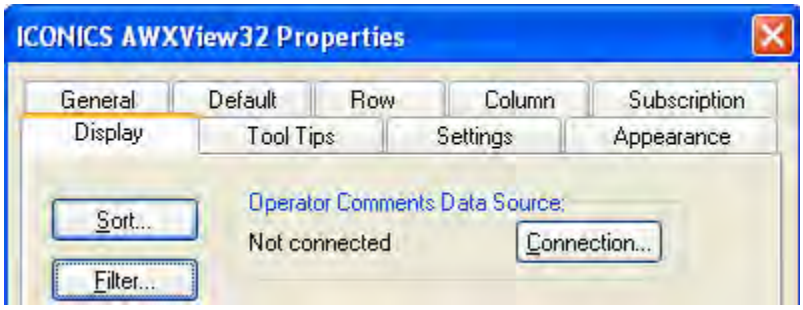

Figure 35: AlarmWorX32 Viewer ActiveX Properties

Select the "**Use Global**" checkbox. Click **Add,** Click **Edit…** and browse for the filter.

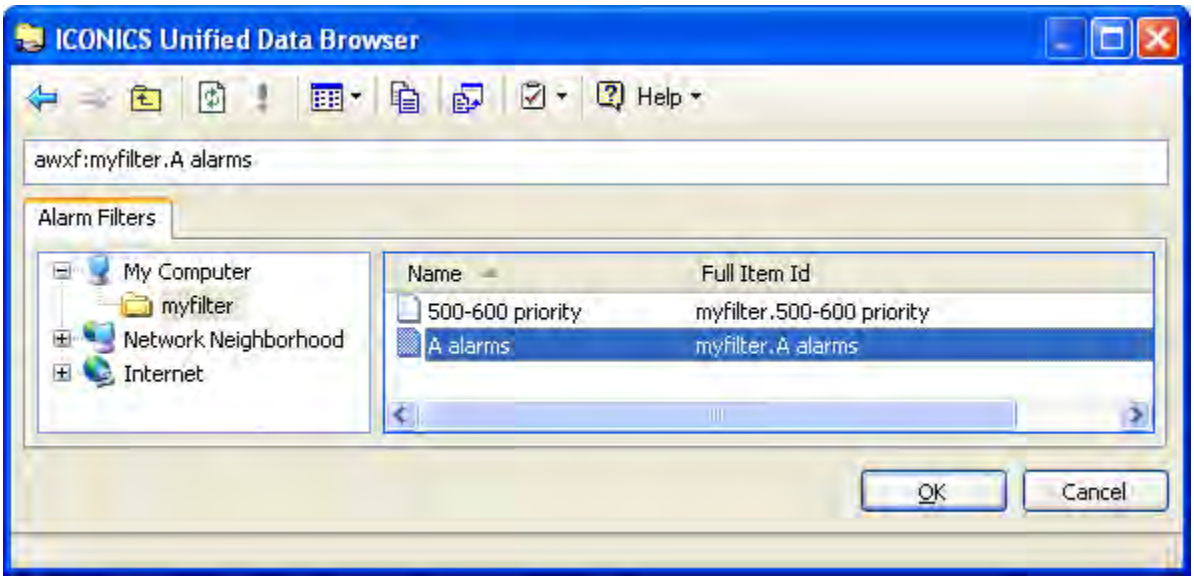

Figure 36: Selecting UDM Alarm Filter in Unified Data Browser

During runtime this AlarmWorX32 Viewer only filters alarms where the alarm tag name starts with: A

Figure 37: Filtered Alarms

Founded in 1986, ICONICS is an award-winning independent software developer offering real-time visualization, HMI/SCADA, energy, fault detection, manufacturing intelligence, MES and a suite of analytics solutions for operational excellence. ICONICS solutions are installed in 70% of the Fortune 500 companies around the world, helping customers to be more profitable, agile and efficient, to improve quality and be more sustainable.

ICONICS is leading the way in cloud-based solutions with its HMI/SCADA, analytics, mobile and data historian to help its customers embrace the Internet of Things (IoT). ICONICS products are used in manufacturing, building automation, oil & gas, renewable energy, utilities, water/wastewater, pharmaceuticals, automotive and many other industries. ICONICS' advanced visualization, productivity, and sustainability solutions are built on its flagship products: GENESIS64™ HMI/SCADA, Hyper Historian™ plant historian, AnalytiX® solution suite and MobileHMI™ mobile apps. Delivering information anytime, anywhere, ICONICS' solutions scale from the smallest standalone embedded projects to the largest enterprise applications.

ICONICS promotes an international culture of innovation, creativity and excellence in product design, development, technical support, training, sales and consulting services for end users, systems integrators, OEMs and Channel Partners. ICONICS has over 300,000 applications installed in multiple industries worldwide.

World Headquarters
100 Foxborough Blvd.
Foxborough, MA, USA, 02035
Tel: 508 543 8600
Email: us@iconics.com
Web: www.iconics.com

European Headquarters
Netherlands
Tel: 31 252 228 588
Email: holland@iconics.com

Czech Republic
Tel: 420 377 183 420
Email: czech@iconics.com

France
Tel: 33 4 50 19 11 80
Email: france@iconics.com

China
Tel: 86 10 8494 2570
Email: china@iconics.com

Italy
Tel: 39 010 46 0626
Email: italy@iconics.com

UK
Tel: 44 1384 246 700
Email: uk@iconics.com

India
Tel: 91 22 67291029
Email: india@iconics.com

Germany
Tel: 49 2241 16 508 0
Email: germany@iconics.com

Australia
Tel: 61 2 9605 1333
Email: australia@iconics.com

Middle East
Tel: 966 540 881 264
Email: middleeast@iconics.com

Microsoft Partner
Gold Application Development

Microsoft Partner
2014 Partner of the Year Winner
Public Sector: CityNext

64™ | Windows 8 Compatible | Windows 10 Compatible | Windows Server 2012 Certified | BTL | CERTIFIED OPC | vmware READY

VISUALIZE | ANALYZE | MOBILIZE | CLOUD

www.iconics.com