



Your comments and suggestions on the operation of this software are welcome. Please address them to:

ICONICS

100 Foxborough Blvd.
Foxborough, MA 02035

Phone: 508-543-8600

Fax: 508-543-1503

E-mail: support@iconics.com

Web: www.iconics.com

[ICONICS Technical Support](#)

© ICONICS, Inc. All rights reserved.

Introduction to Modbus OPC Server

The ICONICS Modbus OPC Server 3.1 is an OPC-compliant server that serves data to OPC clients. The Modbus OPC server was implemented using advanced programming concepts of the current version of the OPC specification for use in developing next generation industrial software applications.

Key features of the Modbus OPC Server 3.1 include:

- Flexible configuration for numerous tags.
- Tag values are persistent.
- Tags can be grouped in logical folders for manageability.
- Tags support configurable range information.
- Tags have configurable alarm conditions.
- Tag multiplier allows you to create hundreds of tags in seconds.
- Supports OPC Data Access (DA) and Alarm and Events (AE) specifications.
- OPC XML-DA wrapper installs with Modbus OPC Server
- TraceWorX32 diagnostics support (logging data into XML file)
- Sample OPC Client
- OPC Admin Utility

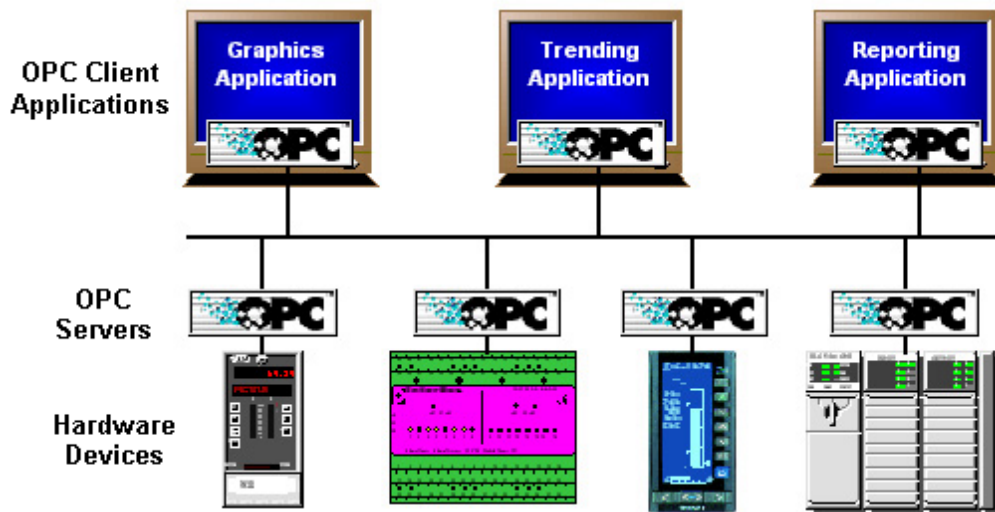
The Modbus OPC Server product contains two parts: a user interface configuration module and the actual OPC server, the runtime module.

The **configuration module** allows you to create a database that holds configuration data of the tags. Such as the tag name, the ranging, and the alarm settings. The **runtime module** uses a runtime database to access the actual values of the tags configured. Structures of both databases are indicated in the following sections.

Changes made to the configuration are accepted only after a restart of the OPC server. However, changes made to the runtime database are accepted online.

What Is OLE for Process Control?

OLE™ for Process Control (OPC) is a standards-based approach for connecting data sources (e.g., PLCs, controllers, I/O devices, databases, etc.) with HMI client applications (graphics, trending, alarming, etc.). It enhances the interface between client and server applications by providing a universally supported and well-documented mechanism to communicate data from a data source to any client application. Included are not only a detailed guide on how to pass the data, but also specific information on other attributes to supplement those data, such as range information, data type, quality flags, and date and time information. The figure below shows the OPC Architecture, which was introduced by the OPC Foundation. By following the OPC Architecture, a device needs only one standard driver, which is an OPC-compliant server. All OPC-compliant client applications can then be connected to that device, either locally or over a network. Furthermore, connections can be made to more than one OPC server at the same time.



OPC-Based Client-Server Architecture

Any OPC client application can connect to any OPC server. In other words, OPC offers true Plug-and-Play capability in the fields of HMI and industrial automation. OPC server types include OPC Data Access (DA), OPC Alarm and Events (AE), and OPC Historical Data Access (HDA).

Installing Modbus OPC Server

This section describes the steps for installing ICONICS Modbus OPC Server. Be sure to close any other applications before installing Modbus OPC Server. For general guidelines and requirements for GENESIS32™ installation, please refer to the ICONICS Getting Started user's guide.

System Requirements

To use the Modbus OPC Server, you must have the following *minimum* system requirements:

- 233 MHz processor
- 64 MB RAM
- 100 MB disk space
- 8X speed CD-ROM drive
- VGA video card (256 or more colors for best results)
- Microsoft® Windows® 2000 with the latest service packs, Windows XP with the latest service packs, Windows NT® 4.0 with the latest service packs, or Windows Embedded NT and XP.

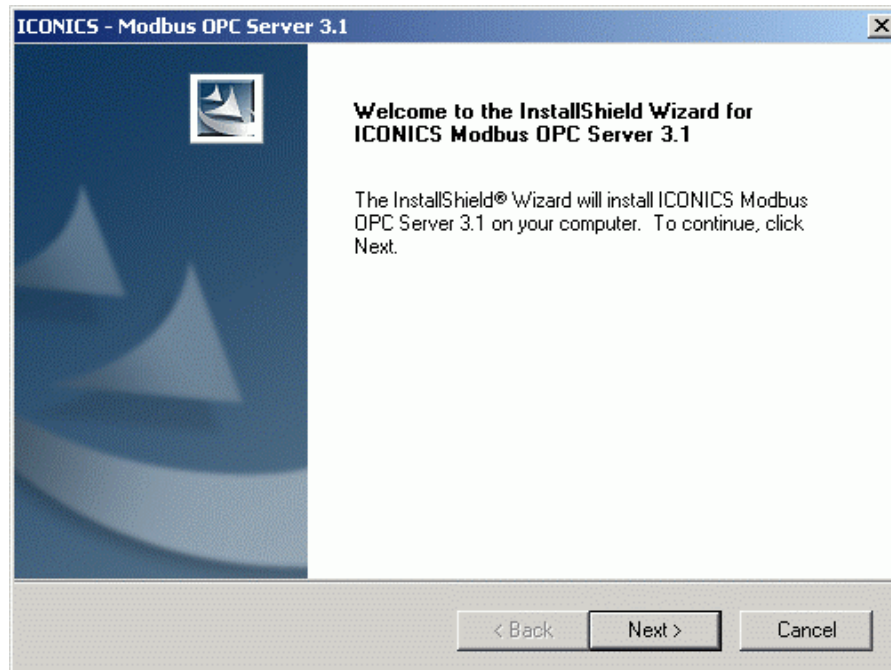
Performance will improve with higher specification hardware (faster data updates).

Installation Procedure

The following steps detail the Modbus OPC Server installation:

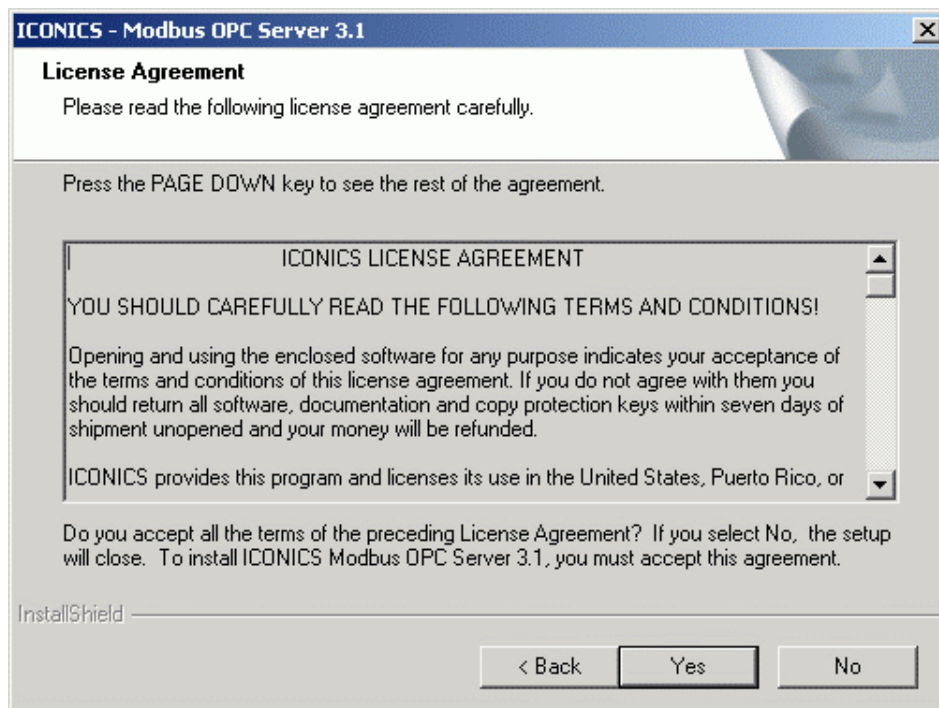
Note: If your operating system (e.g., Windows NT) requires a login name, you must log in with administrator capability before installing Modbus OPC Server.

1. Before installing the software, be sure that all other applications are closed and/or disabled.
2. Insert the ICONICS product CD and select **Modbus OPC Server**. The **Welcome** screen will appear, as shown below.



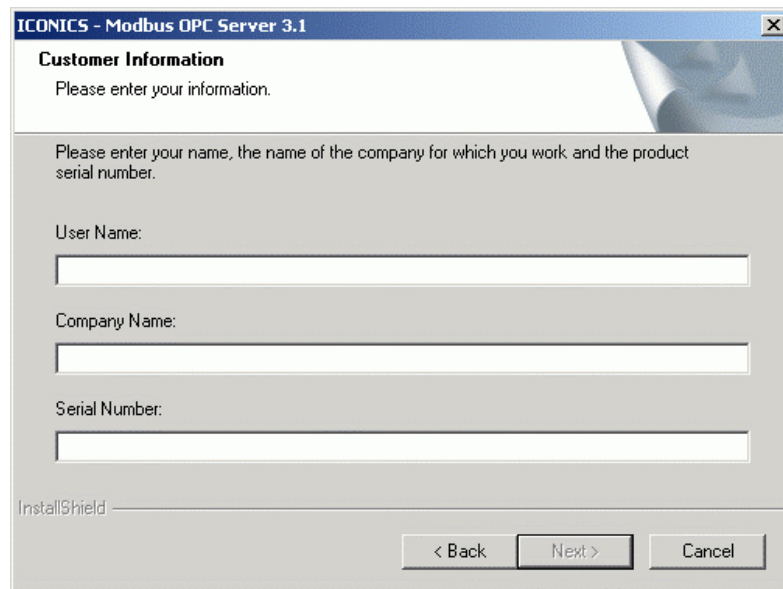
Installation Welcome Screen

3. The **ICONICS License Agreement** dialog box will appear, as shown below. Carefully read the ICONICS License Agreement. If you agree with the terms and conditions, click the **Yes** button to continue.



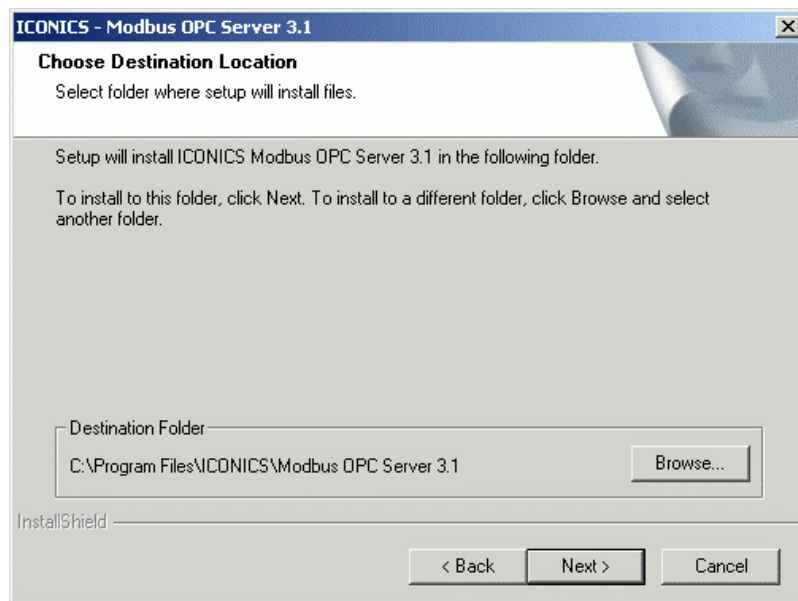
ICONICS License Agreement

4. The **User Information** dialog box will appear, as shown below. Type your (the user's) name in the **Name** field and your company's name in the **Company** field. Click the **Next** button to continue.

The image shows a Windows-style dialog box titled "ICONICS - Modbus OPC Server 3.1". The main heading is "Customer Information" with a subtext "Please enter your information." Below this, there is a larger text area that says "Please enter your name, the name of the company for which you work and the product serial number." There are three input fields: "User Name:", "Company Name:", and "Serial Number:". At the bottom left, it says "InstallShield". At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

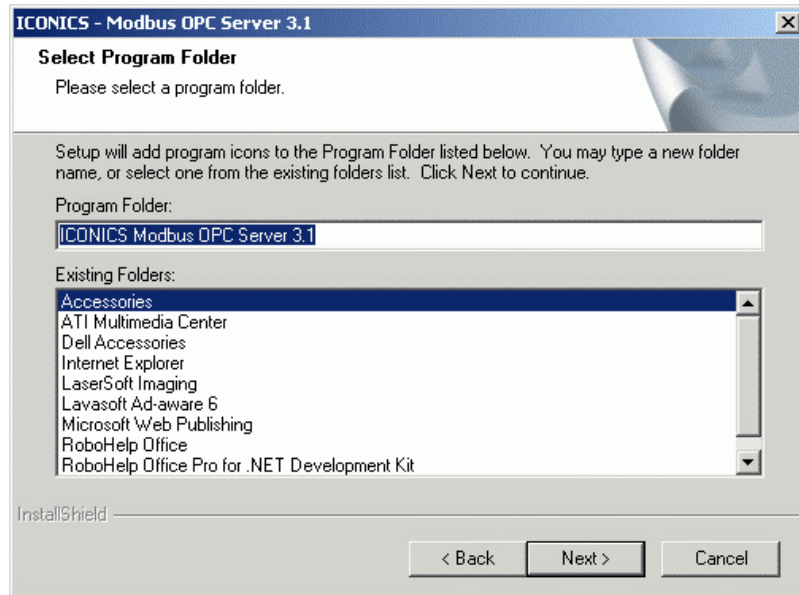
User Information Dialog Box

5. The **Installation Destination** dialog box will appear, as shown below. The default location is "C:\Program Files\ICONICS\Modbus OPC Server 3.1." If you wish to specify a different directory for installation, click the **Browse** button and choose the appropriate installation folder. After you have chosen a location, click the **Next** button to continue the installation.

The image shows a Windows-style dialog box titled "ICONICS - Modbus OPC Server 3.1". The main heading is "Choose Destination Location" with a subtext "Select folder where setup will install files." Below this, there is a text area that says "Setup will install ICONICS Modbus OPC Server 3.1 in the following folder." and "To install to this folder, click Next. To install to a different folder, click Browse and select another folder." There is a text box labeled "Destination Folder" containing the path "C:\Program Files\ICONICS\Modbus OPC Server 3.1". To the right of this text box is a "Browse..." button. At the bottom left, it says "InstallShield". At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

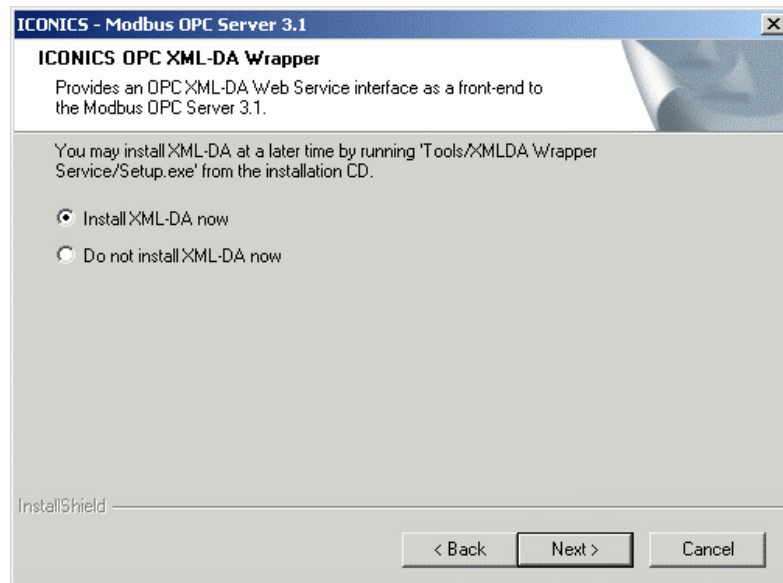
Choosing the Installation Location

6. The **Select Program Folder** dialog box will appear, as shown below. Either type the name of a new program folder in the **Program Folders** field or select the program folder of your choice from the list in the pane below. Click the **Next** button to continue.



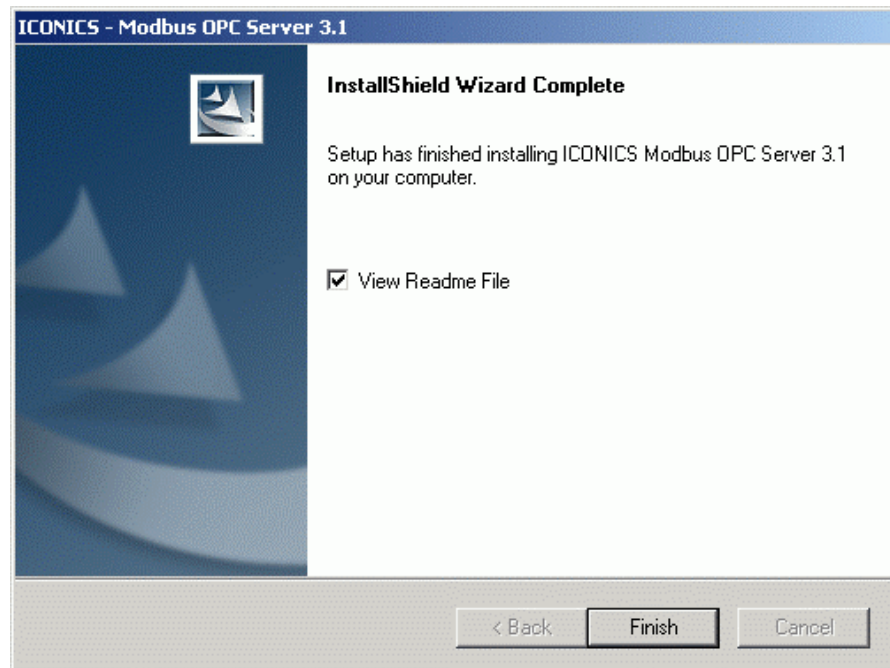
Selecting the Program Folder

7. The **ICONICS OPC XML-DA Wrapper** dialog box will appear, as shown below. This gives you the option on installing the OPC XML DA (Data Access) wrapper. This provides an OPC XML DA Web service interface as a front end to the Modbus OPC Server. If you do not wish to install the wrapper at this time, you may install it later from the installation CD. Click the **Next** button to continue.



OPC XML-DA Wrapper Installation

8. After all files and components have been installed, the **Setup Complete** dialog box appears, as shown in the figure below. Click the **Finish** button to complete the installation. Restart your computer before using the product.



Setup Complete Dialog Box

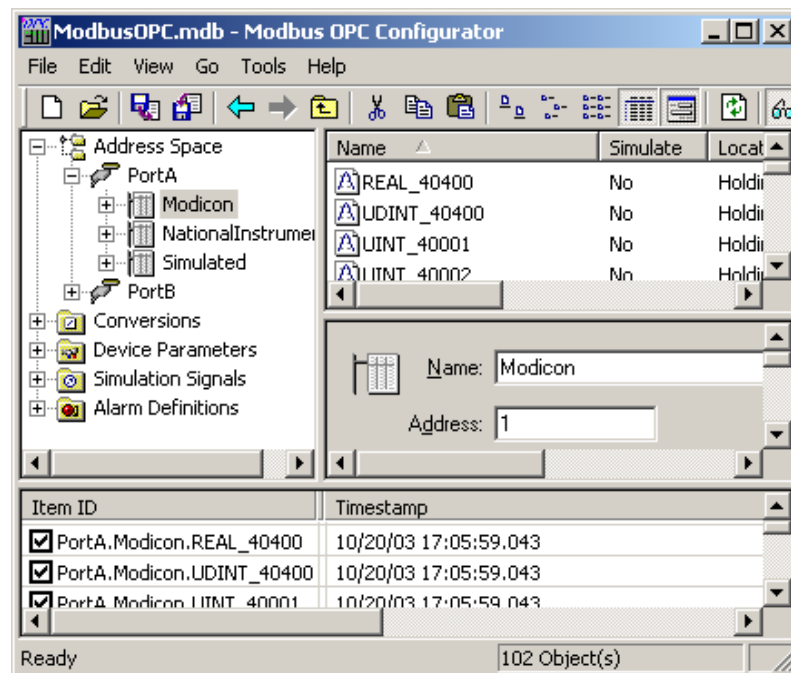
Modbus Configurator

The ICONICS Modbus Configurator module is used to configure the data items (tags). This configuration is stored to the configuration database (Microsoft Access database file *.mdb). The Modbus OPC Server Runtime module will read this database on startup and creates its **Address Space** according to this database.

Starting the Modbus Configurator

To open the Modbus Configurator from the Windows **Start** menu, select **Programs > ICONICS Modbus OPC Server 3.1 > Modbus OPC Configurator**. Note that a **Sample OPC Client** is also available from this menu.

The Configurator screen will open, as shown in the figure below. The screen consists of a split window with a tree control view in the left-hand pane and a configuration view in the right-hand pane. The installation provides a default, configured Microsoft Access database file ("ModbusOPC.mdb") that provides a standard format for the configuration database. The Configurator also includes a toolbar and menus with many command functions. For a description of the toolbar and menu functions, please see the sections below.



Modbus Configurator Screen

Toolbar

The Modbus Configurator toolbar, shown below, contains the following command buttons. To show or hide the toolbar, select **Toolbars - Standard Buttons** from the **View** menu. For more information on menu functions, please refer to the **Menus** section below.

Standard Toolbar

	New: Creates a new configuration database.
	Open: Opens an existing database.
	Import Data: Imports configuration data from a .txt or a .csv file.
	Export Data: Exports configuration data to a .txt or a .csv file.
	Back: Moves the cursor back to the previously selected item.
	Next: Moves the cursor to the next item in the tree control.
	Up One Level: Moves up one level in the tree view.
	Cut: Deletes current selection, sending it to the clipboard.
	Copy: Copies the current selection to the clipboard.
	Paste: Pastes the current contents of the clipboard.
	Large Icons: Displays items as large icons.
	Small Icons: Displays items as small icons.
	List: Displays items as a list.
	Details: Displays items as a list with details.
	Dialog View: Displays additional configuration options.
	Global Refresh: Refreshes the data for the entire Configurator screen.
	Monitor View: Displays OPC server data in a separate pane.
	About: Displays information about the application.

Data Manipulation Toolbar

	New Port: Inserts a new port into the configuration.
	New Device: Inserts a new device into the configuration.
	New Folder: Inserts a new folder into the configuration.
	New Data Item: Inserts a new data item into the configuration.
	Multiply: Duplicates the selected item in the tree control.
	Delete: Deletes the selected item in the tree control.

Menus

The menu bar of the Configurator contains the following menus:

- File
- Edit
- View
- Go
- Tools
- Help

Note: You can also access many of the menu commands by right-clicking items in the tree control of the Configurator and selecting command functions from the pop-up menus.

File Menu

The File menu commands are listed in the table below.

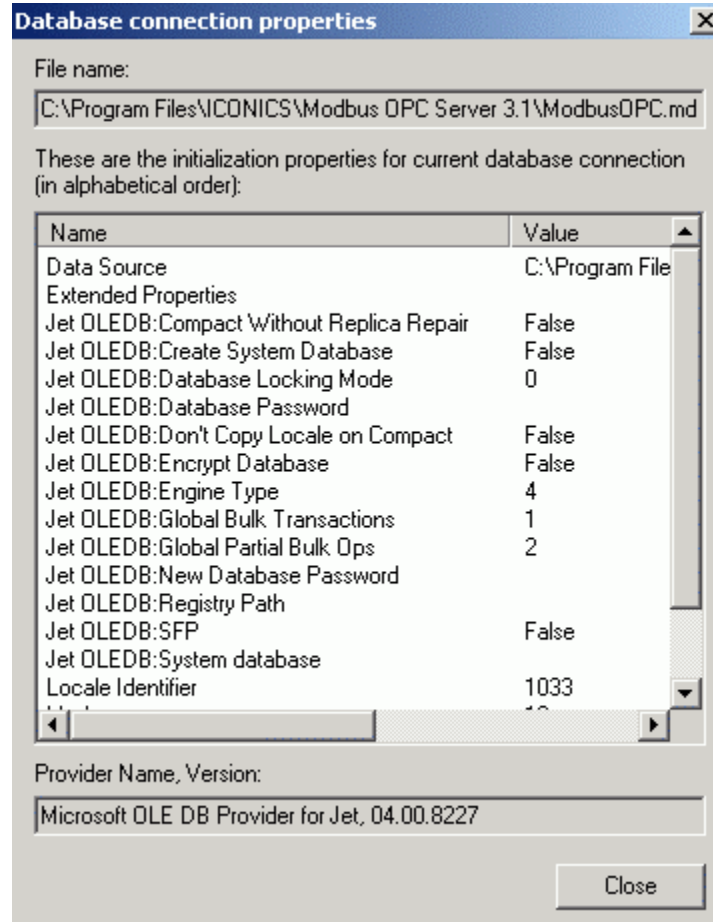
File Menu Commands

Command	Shortcut Keys	Function
New	CTRL+N	Creates a new configuration database.
Open	CTRL+O	Opens a Microsoft Access (.mdb) or Microsoft Data Link (.udl) file, which allows you to connect to any OLE database source, such as a SQL Server database.
Save As		Saves the current database under a different name as a Microsoft Access (.mdb) or Microsoft Data Link (.udl) file.
Connection Properties		Displays the current database connection properties.
Windows CE Export		Exports all data into an extended .csv file. This file can then be transferred to a Windows CE device and used by the server running there. This additional export step is necessary, as Windows CE does not support .mdb or other database formats.

Export CSV		Exports configuration data from your database to a text file (.txt) or a Microsoft Excel file (.csv). You can specify the delimiters and what to export.
Import CSV		Imports data into your configuration database from a text file (.txt) or a Microsoft Excel file (.csv). You can then specify the delimiters and choose from the import settings.
XML Export		Exports configuration data to an XML file.
XML Export Schema		Exports configuration data to an XML Schema file.
XML Import		Imports configuration data from an XML file.
XML Validate		This feature does not import an XML data file, but it will try to validate its structure using stored XML schema. Once it passes this validation, the XML file is acceptable for import by the Configurator.
Make Active		Makes the current database active. If this command is not available, then the current database is already the active database.
Exit		Closes the application. The current configuration database is automatically saved.

Database Connection Properties

Selecting **Connection Properties** from the **File** menu opens the **Database Connection Properties** dialog box, shown below, which lists the initialization properties for the current database connection.

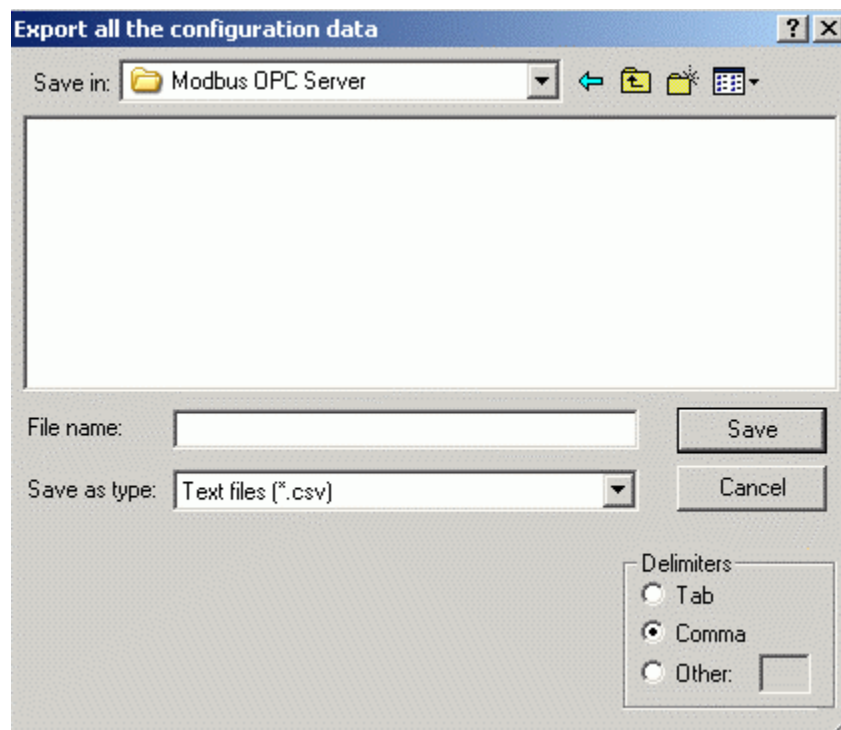


Database Connection Properties Dialog Box

Exporting Configuration Data

Exporting Data to a Text or CSV File

The Configurator offers the flexibility of exporting data from your configuration database to a text (.txt) file or a Microsoft Excel (.csv) file. To export data, select **Export** from the **File** menu. This opens the **Export Configuration Data to File** dialog box, as shown in the figure below. You can then specify the delimiters for exporting the data. Unless you specify delimiters in the **Export Configuration Data to File** dialog box, the file uses **Commas** as delimiters by default. Each group contains headings and columns that provide information about each item, such as descriptions and associated translations and expressions. It also provides the "tree" pathway for each item. Choose the directory to which you want to export the data from your database. In the **Save As Type** field, choose the file type (.txt or .csv) that you would like to save.



Exporting Configuration Data

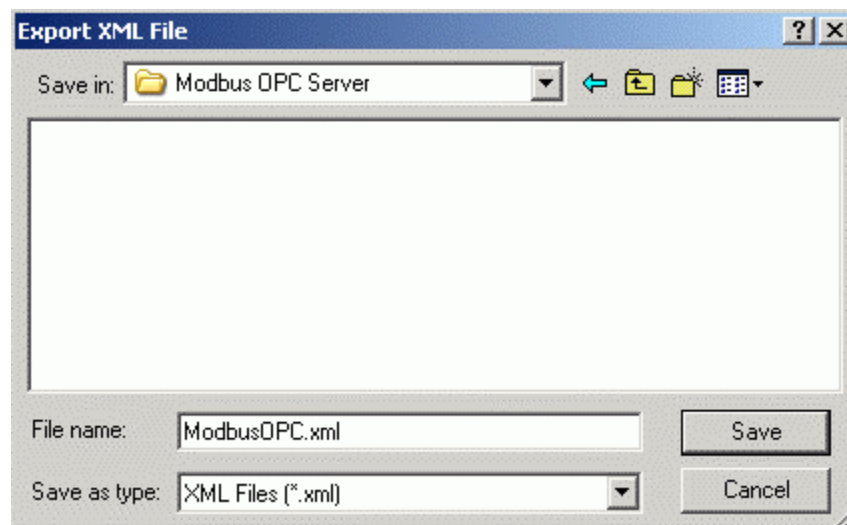
Exporting Data to an XML File

The Configurator also allows you to export data from your configuration database to an XML file. The XML export/import functionality was mainly developed for Windows platforms that do not support databases (e.g. Windows CE and Windows Embedded). XML has the following advantages over the CSV import/export function:

- XML has a standardized format, unlike the text/CSV format, which uses various delimiters (e.g. TAB instead of commas, strings could not accept all characters, etc.)
- XML is language-independent, whereas CSV converts date/time, floats, and currency fields according to local settings in Windows. For example, using CSV, you cannot export data on German Windows and import it on English windows without making changes
- Windows has an installed automation object that has the capability to work with XML. Thus, programmers can create/modify their configurations outside the Configurator using Visual Basic, if desired.
- XML supports schemas. A **schema** is a special XML file that specifies the data structure of an XML data file.

To export data, select **XML Export** from the **File** menu. This opens the **Export XML File** dialog box, as shown in the figure below. Give the file a name, and then choose the directory to which you want to export the data from your database. Click **Save**.

Note: You can also export configuration data to an XML Scheme file by selecting **XML Export** from the **File** menu.



Exporting Configuration Data to an XML File

Importing Configuration Data

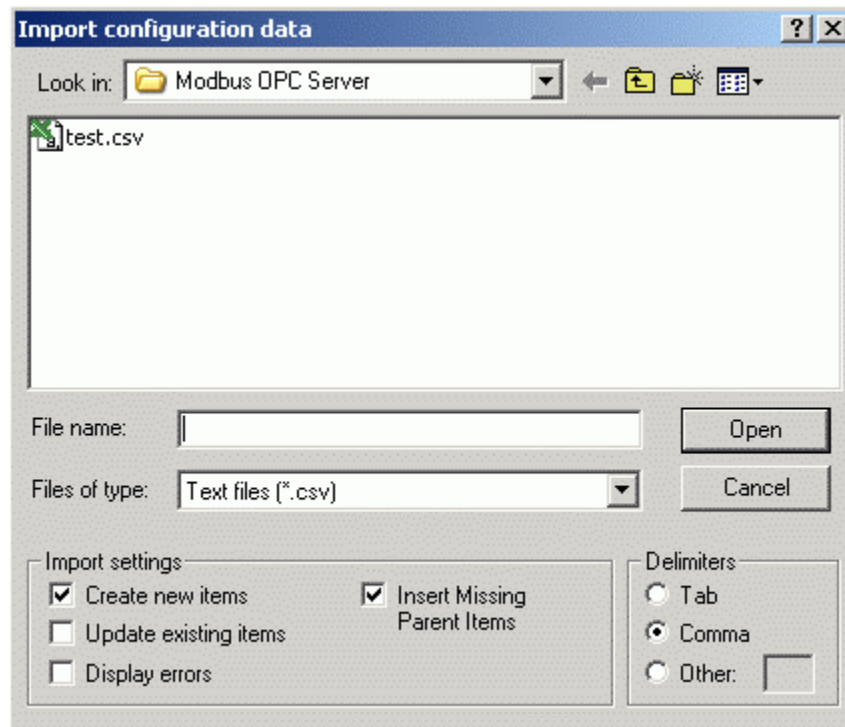
Importing Data From a Text or CSV File

The Configurator offers the flexibility of importing data from a text (.txt) file or a Microsoft Excel (.csv) file to your configuration database. To import data, select **Import CSV** from the **File** menu. This opens the **Import Configuration Data From File** dialog box, shown below. You can then specify the delimiters and choose from the following import settings:

- **Create new items.** When the import file contains items that are not yet in the configuration database, then it creates them. Otherwise it skips these items.
- **Update existing items.** When the import file contains items that are in the configuration database, then it updates them using data from the import file. Otherwise it skips these items.

Note: Either **Create new items** or **Update existing items** must be selected. Otherwise there is nothing to import.

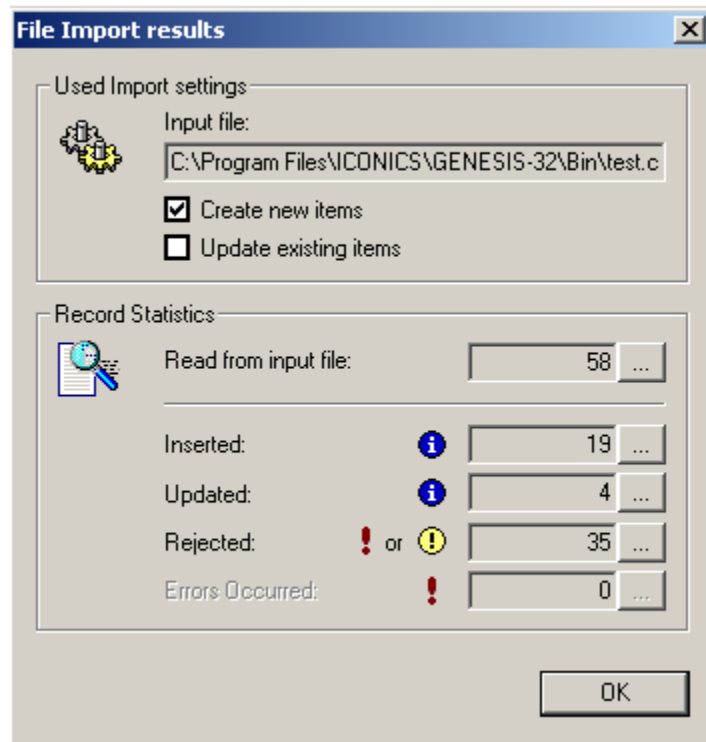
- **Display errors.** When this item is checked, the Configurator shows a dialog box if an error occurs, and then asks you if you want to proceed with the import. When it is not checked, it skips all items where an error occurred.



Importing Configuration Data

When you have selected a file to import, click **Open**. When the import is completed, the **File Import Results** dialog box opens, as shown below. This shows the import settings, including the input file name. It also provides a summary of the import, including how many items were inserted, updated, or rejected, and shows how many errors occurred.

Click the ... button to the right of each field to get the details view of the import results, as shown below. This view shows the specific items that were inserted, updated, or rejected, as well as a description of any errors that occurred.



File Import Results Dialog Box

Importing Data From an XML File

The Configurator allows you to import data from your configuration database to an XML file. The XML export/import functionality was mainly developed for Windows platforms that do not support databases (e.g. Windows CE and Windows Embedded). XML has the following advantages over the CSV import/export function:

- XML has a standardized format, unlike the text/CSV format, which uses various delimiters (e.g. TAB instead of commas, strings could not accept all characters, etc.)
- XML is language-independent, whereas CSV converts date/time, floats, and currency fields according to local settings in Windows. For example, using CSV, you cannot export data on German Windows and import it on English windows without making changes
- Windows has an installed automation object that has the capability to work with XML. Thus, programmers can create/modify their configurations outside the Configurator using Visual Basic, if desired.

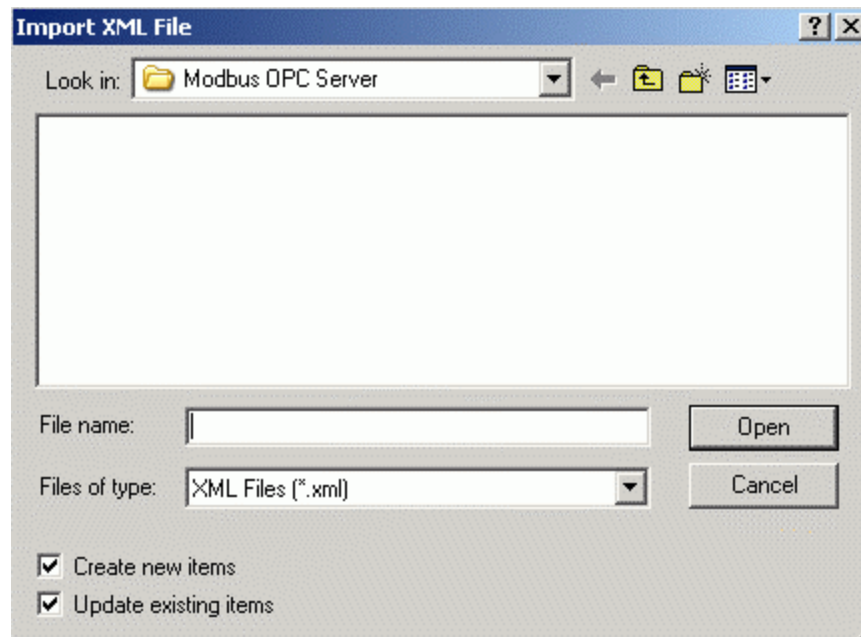
- XML supports schemas. A **schema** is a special XML file that specifies the data structure of an XML data file.

To import data, select **XML Import** from the **File** menu. This opens the **Import XML File** dialog box, as shown in the figure below. Give the file a name, and then choose the directory from which you want to import the data. You can then specify the delimiters and choose from the following import settings. Click **Open**.

- **Create new items.** When the import file contains items that are not yet in the configuration database, then it creates them. Otherwise it skips these items.
- **Update existing items.** When the import file contains items that are in the configuration database, then it updates them using data from the import file. Otherwise it skips these items.

Note: Either **Create new items** or **Update existing items** must be selected. Otherwise there is nothing to import.

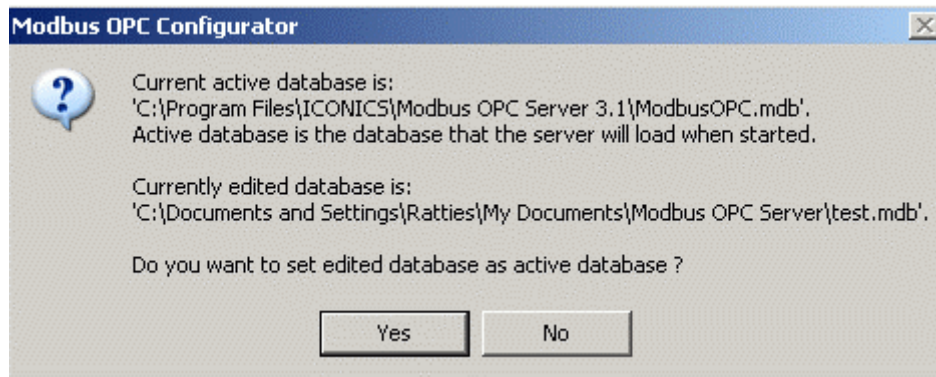
Note: Selecting **XML Validate** from the **File** menu does not import an XML data file, but it will try to validate its structure using stored XML schema. Once it passes this validation, the XML file is acceptable for import by the Configurator.



Importing Configuration Data From an XML File

Activating the Database

Once your configuration is complete, you need to make sure that it is the active database. The database that is currently active is the one that the server uses. To make the current database active, select **Make Active** from the **File** menu. A dialog box appears showing both the current active database and database that is currently being edited, as shown in the figure below. To set the edited database as the active database, click the **Yes** button. Next time the server goes into runtime, it will use this active database for all of its operations.



Activating the Database

Edit Menu

The **Edit** menu commands are listed in the table below.

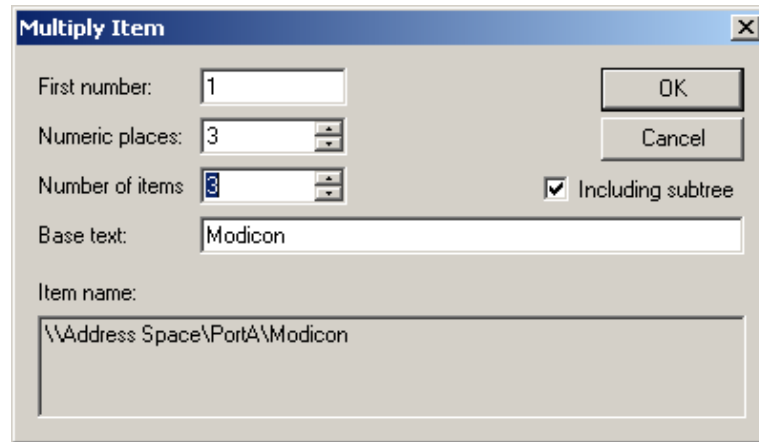
Edit Menu Commands

Command	Shortcut Keys	Function
New		Creates a new item depending on what is selected in the tree control.
Rename	CTRL+R	Renames the selected item.
Multiply	CTRL+M	Opens the Multiply Item dialog box, which allows you to duplicate the selected item in the tree control.
Delete	CTRL+DEL	Deletes the selected object.
Cut	CTRL+X	Cuts the selected object from the view and places it on the clipboard.
Copy	CTRL+C	Copies the selected object to the clipboard.
Paste	CTRL+V	Pastes the last object placed on the clipboard.
Select All	CTRL+A	Selects all objects in a list. The selection is shown in the upper-right-hand section of the viewer.
Invert Selection		Unselects all selected items and selects all unselected items.

Multiplying Items

The Configurator allows you to multiply items in the tree control. Multiplication provides a simple way of developing configurations where there are many similar items in a given category. To multiply an item:

1. Select the item in the tree control that you wish to multiply.
2. Either right-click the item and select **Multiply** from the pop-up menu, or select **Multiply** from the **Edit** menu. This opens the **Multiply Item** dialog box, shown below.



Multiply Item Dialog Box

3. When the items are multiplied, they are all given a base name followed by a number. The default base text is the name of the item selected for multiplication. To modify the base text, change the **Base Text** field appropriately.
4. In the **First Number** field, specify the number to appear next to the first multiplied item.
5. In the **Number of Items** field, specify how many items you wish to create.
6. In the **Numeric Places** field, specify the minimum length of each number to append. Values that take up less space than the specified amount of numeric places will have zeros before the number.
7. If you want to multiply all subfolders as well, check the **Including Subtree** check box.
8. Click the **OK** button to do the multiplication. The example configuration shown in the **Multiply Item** dialog box above creates three new items with the following names:
 - Modicon001
 - Modicon002
 - Modicon003

View Menu

The **View** menu commands are listed in the table below.

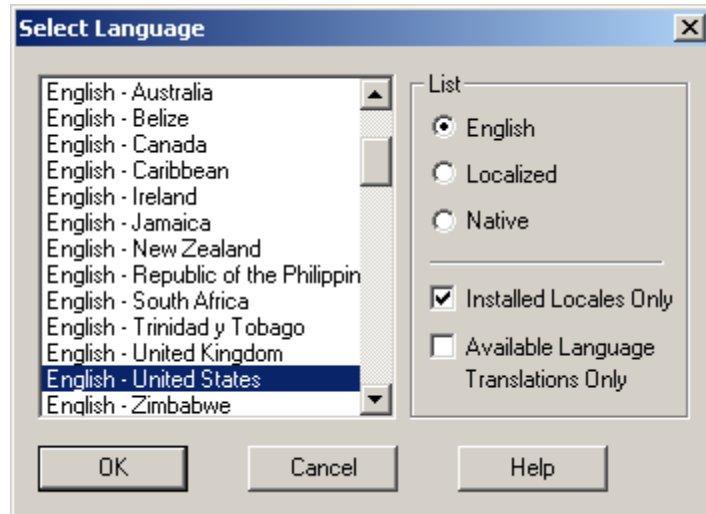
View Menu Commands

Command	Shortcut Keys	Function
Toolbar		Shows/hides the Configurator toolbar buttons.
Status Bar		Toggles the status bar at the bottom of the Configurator window.
Large Icons	F7	Displays items as large icons.
Small Icons	F8	Displays items as small icons.
List	F9	Displays items as a list.
Details	F10	Displays items as a list along with detailed information about the configuration of each item.
Dialog View	F11	Toggles the configuration window (right-hand pane).
Monitor View	F12	Displays OPC server data in a separate pane.
Sort By		Displays a list of options for sorting the columns in the right-hand pane of the screen. The options listed depend on the level within the view.
Show/Hide Columns		Displays a list of options that you can choose to show or hide in the view.
Select Language		Opens the Select Language dialog box. Choose the language you wish to use for your system and click OK . For navigation purposes, use the buttons and check boxes in the List section.
Global Refresh	F5	Refreshes the data for the entire Configurator screen.
Subtree Refresh	CTRL+F5	Refreshes only the data contained in the currently selected subtree.

Selecting Languages

The **Select Language** function on the **View** menu allows you to choose which language to use in your display. Choosing **Select Language** from the **View** menu opens the Select Language dialog box, shown in the figure below.

Note: A language resource .dll is required for language switching.



Select Language Dialog Box

Define the parameters listed in the table below. Then click **OK** to return to the work area.

Select Language Parameters

Parameter	Description
List	Lists available languages. Depending on which item you have selected, the view on the left will change. If English is checked, the languages will appear as their English name. If Localized is checked, the languages will appear with the native country in parentheses (for languages with several dialects only). When Native is checked, the languages are displayed the way they would be written in that language.
Installed Locales Only	If this is checked, local languages appear in the box.
Available Language Translations Only	Checking this box allows you to choose from available language translations only.

Go Menu

The **Go** menu commands are listed in the table below.

Go Menu Commands

Command	Shortcut Keys	Function
Back	CTRL+ALT+Left Arrow	Moves the cursor back to the previously selected item in the tree control.
Forward	CTRL+ALT+Right Arrow	Moves the cursor forward to the previously selected item in the tree control.
Up One Level		Moves the cursor up one level in the tree control.
Next Item	ALT+Down Arrow	Moves the cursor to the next visible item in the tree control.
Previous Item	ALT+Up Arrow	Moves the cursor to the previous visible item in the tree control.
Expand Item	ALT+Left Arrow	Expands an item that contains a submenu.
Collapse Item	ALT+Right Arrow	Collapses an item that contains a submenu.
Page Up	ALT+PgUp	Moves the cursor up to the first item in the tree.
Page Down	ALT+PgDown	Moves the cursor down to the last visible item in the tree.
Home	ALT+Home	Moves the cursor up to the first item in the tree.
End	ALT+End	Moves the cursor down to the last visible item in the tree.
Next Pane	F6	Moves the cursor to the next pane in the window.
Previous Pane	SHIFT+F6	Moves the cursor to the last pane used.

Tools Menu

The **Tools** menu commands are listed in the table below.

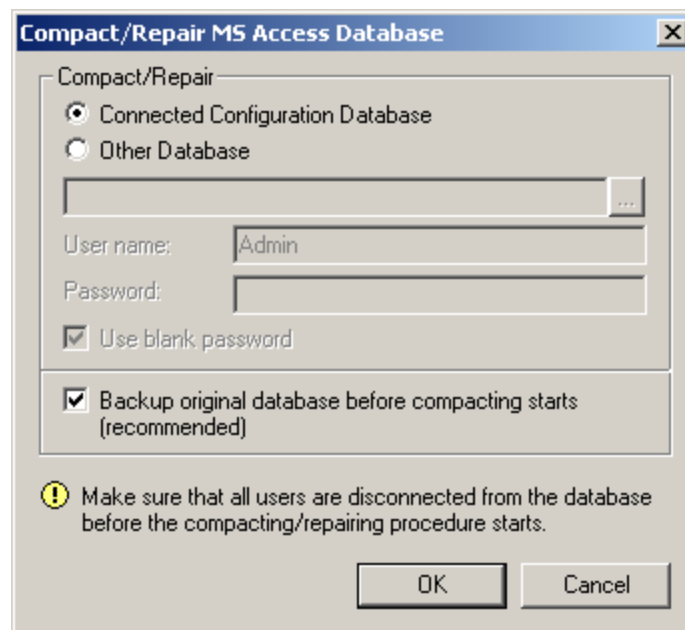
Tools Menu Commands

Command	Function
Options	Launches the Options dialog box.
Compact/Repair MS Access Database	Opens the Compact/Repair MS Access Database dialog box.

Compacting and Repairing MS Access Databases

You can compact Microsoft Access databases, which can be either configuration databases or historical databases, using the **Compact/Repair MS Access Database** dialog box, shown in the figure below. To open this dialog box, select **Compact/Repair MS Access Database** from the **Tools** menu. Microsoft Access-based databases are subject to database fragmentation over time, and the support for the database will compact the target database, reclaim unused space, and drastically improve database performance.

Note: It is critical that no users or client applications are connected to the database at the time of compacting and that, if the **Backup Original Database** option is selected, there is plenty of available hard disk space.



Compact/Repair MS Access Database Dialog Box

Help Menu

The **Help** menu commands are listed in the table below.

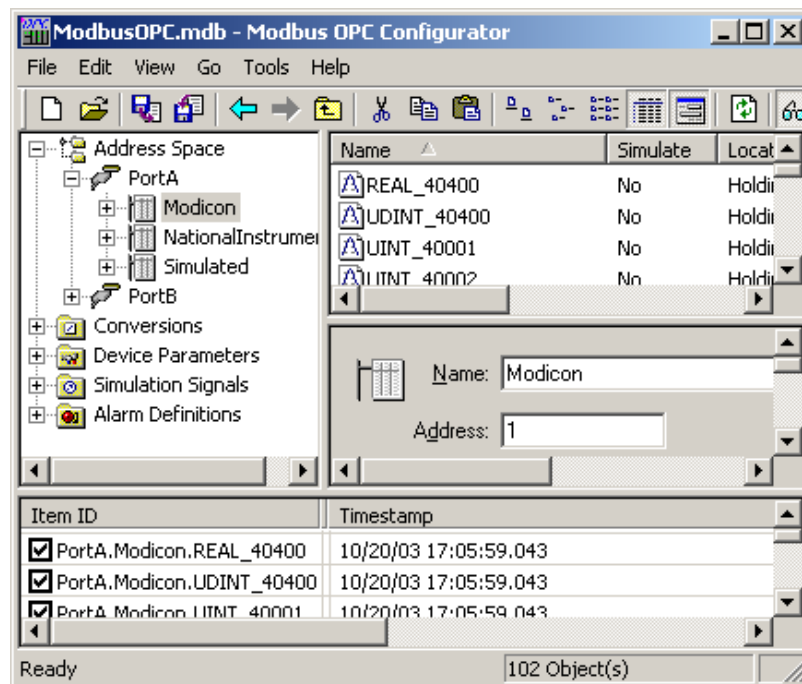
Help Menu Commands

Command	Shortcut Keys	Function
Help Topics	F1	Launches the online Help for the Configurator.
About Application		Launches the About Box , which contains information about the product version number and copyright.

Creating a New Configuration

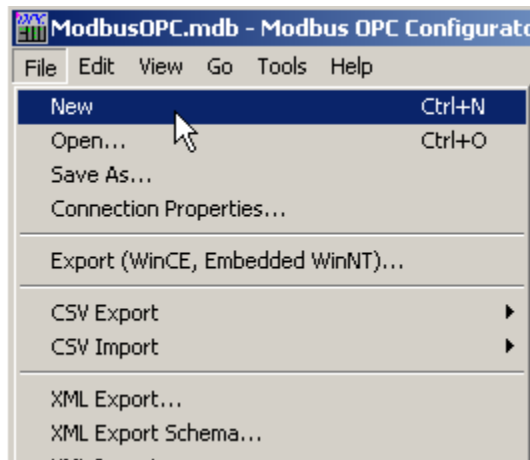
You must first create a database in which your configuration can be stored:

1. From the Windows **Start** menu, select **Programs > ICONICS Modbus OPC Server 3.1 > Modbus OPC Configurator**.
2. This opens the Configurator, as shown in the figure below. The screen consists of a split window with a tree control view in the left-hand pane and a configuration view in the right-hand pane. The Configurator provides a standard format for the configuration database, as well as a sample (default) configuration project. The Configurator also includes a toolbar and menus with many command functions.



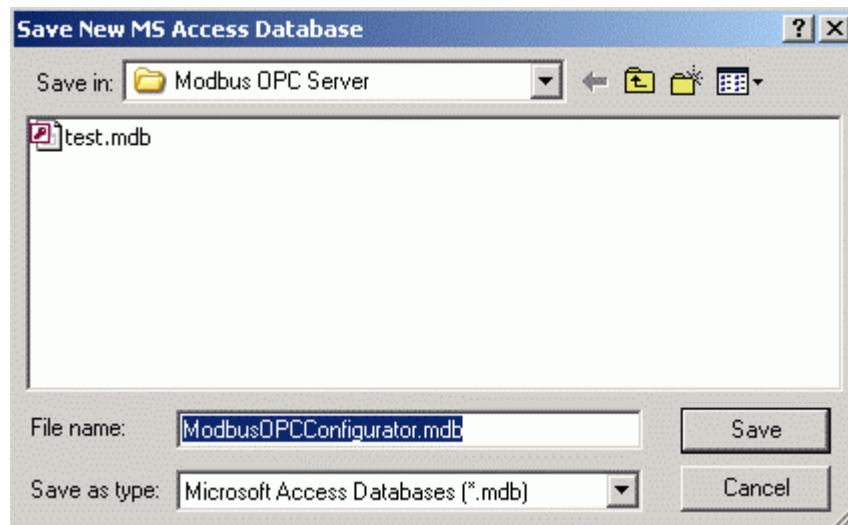
Configurator Screen

3. The Configurator uses Microsoft Access configuration databases. To create a new configuration database in the Configurator: Select **New** from the **File** menu, as shown in the figure below.



Creating a Configuration Database

4. In the **Save New Database** dialog box, select the database type from the **Save As Type** drop-down list, as shown in the figure below. Browse for the target directory, give the file a name, and then click the **Save** button.



Saving the New Configuration Database

Modbus-Specific Features

Modbus is a serial communication protocol allowing connection of 247 devices on a serial line. There is always one device (primary) controlling the communication. The rest consists of secondary devices. Every device is identified by its unique address. Its registers are read as Input (1 bit long) or Input Register (16 bits), or written to as Coil (1 bit) or Holding Register (16 bits). Registers of each type are addressed by using 16-bit numbers.

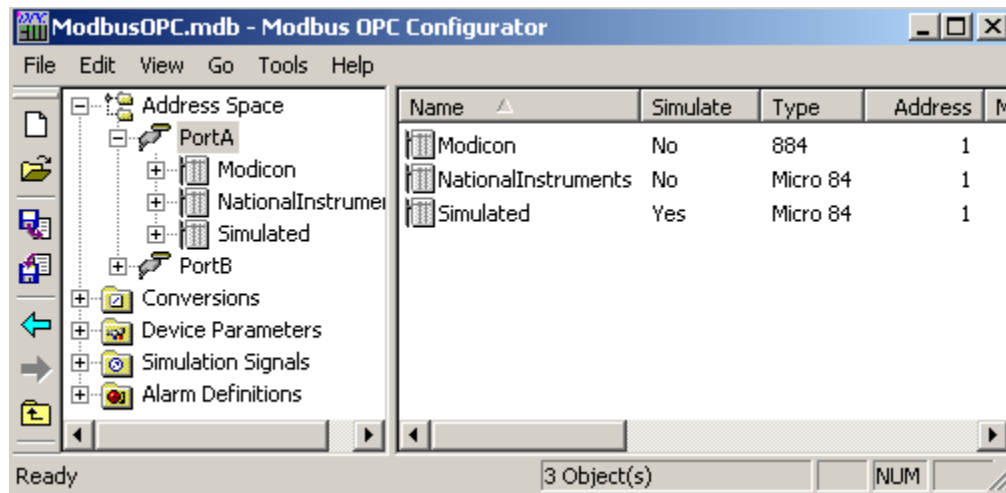
Address Space

The **Address Space** tree control in the Configurator, shown in the figure below, sets the properties and connection parameters for the following items:

- Ports
- Devices
- Folders
- Data Items

Data items can be organized hierarchically. It is similar to organizing directories and files on your computer's hard disk. The Modbus OPC Server offers several levels of hierarchy. The Configurator module uses the terms *Folder* and *Data Item*. A **folder** can contain additional folders and also data items. The **data items** are always the branches in the tree control hierarchy.

The hierarchical structure of the folders and data items helps to organize the devices and tags, as shown below.

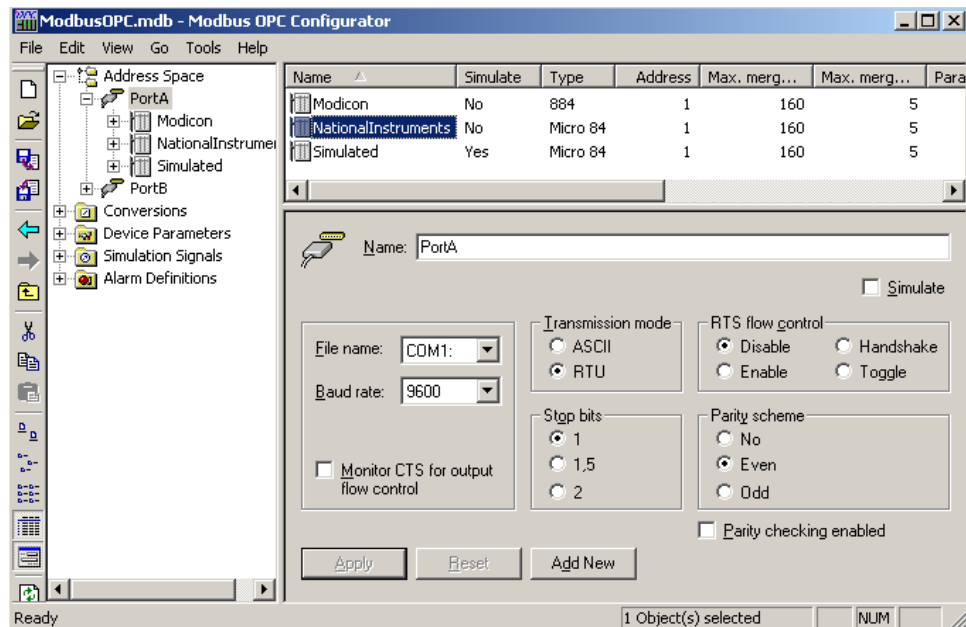


Address Space

Port

A **port** is equivalent to a physical serial port in the computer. To correctly configure a port means to set up a serial port as a **File name** (COM 1, COM2, COM3, or COM4), communication speed (**Baud rate**), and protocol characteristics: **Transmission mode** (ASCII or RTU), **RTS flow control**, **Stop bits** and **Parity scheme**.

When the **Parity checking enabled** check box is not checked, it will force the server to ignore the parity bits in the message.

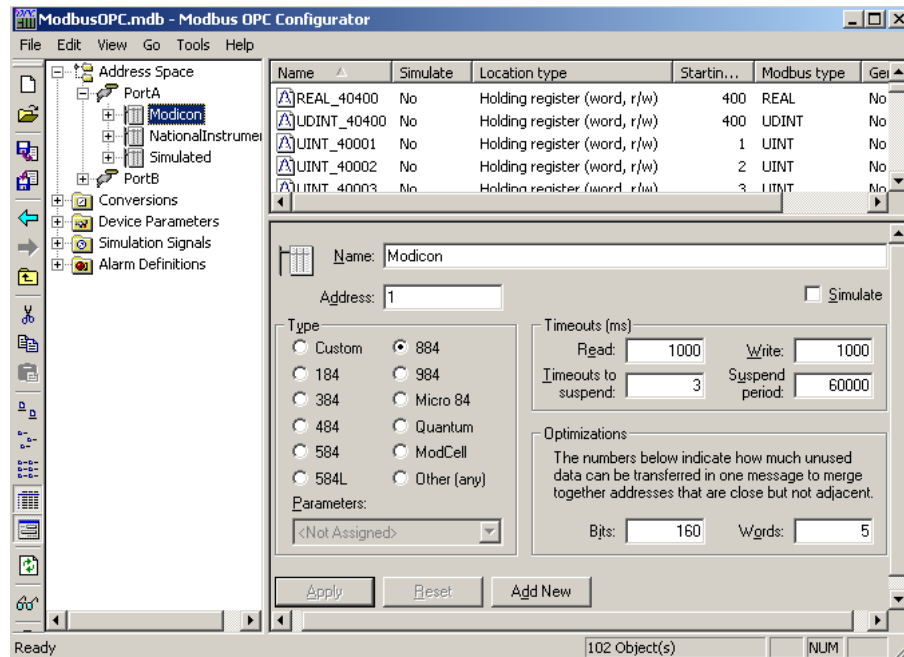


Configuring Ports

Devices

Every **device** is connected to particular port, so it logically creates the second level in the **Address Space** tree. Each device is represented by its symbolic name, and is uniquely identified by the Address value. It is impossible to have two devices with the same address connected to one port.

Setting up a device requires configuration of its unique address, type, timeouts and optimization parameters, as shown in the figure below.



Configuring Devices

Device Type

There is a group of six predefined standard device types enhanced with **Other (Any)** and **Custom** options. A device with the most limited parameters and the lowest performance is called the **Other (Any)** alternative.

If you have devices that are among the list of predefined Modicon types, use the **Custom** option and select one of the predefined device types from the **Parameters** drop-down list. For instruction on how to create new or edit predefined devices, see the "Device Parameters" section below.

Timeouts

Timeout parameters (separately for reading and writing data) specify the period of time (in milliseconds) the server will wait for a response from the device.

Optimizations

The server tries to optimize the communication with the devices by requesting as much data as possible in one message. Consecutive registers are merged together into one request for efficiency. The server also can read registers that are not really requested, if this allows it to join two blocks of requested registers. The numbers entered under **Optimizations** specify the maximum block length of adjacent unused data.

Folders

Folders can be used to group items logically. You can configure as many folders as required. Each folder can even have subfolders. The Modbus OPC Server supports up to three folder levels. The use of folders is not required.

If desirable, the configuration could just contain data items without any folders. But most likely this will only be useful if the application does not demand too many persistent tags.

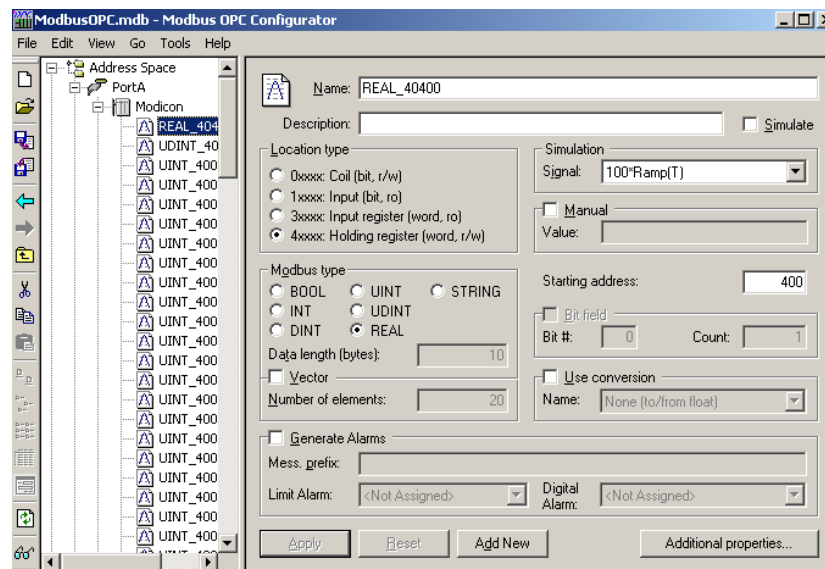
Data Items

A **Data Item** represents a register in the device or a range of registers. A symbolic name and description is associated with the data item. An OPC client can obtain the data item description.

The actual OPC item name (tag) is compounded from the **Address Space** root, the names of the folder and its subfolders, and the name of the data item. Data items can be located in any folder, even in the root of the address space.

Data items have the following properties, as shown in the figure below.

- **Name:** A logical name for the data item (Setpoint, Param001, ON_OFF, etc).
- **Description:** A descriptive comment for the data item.



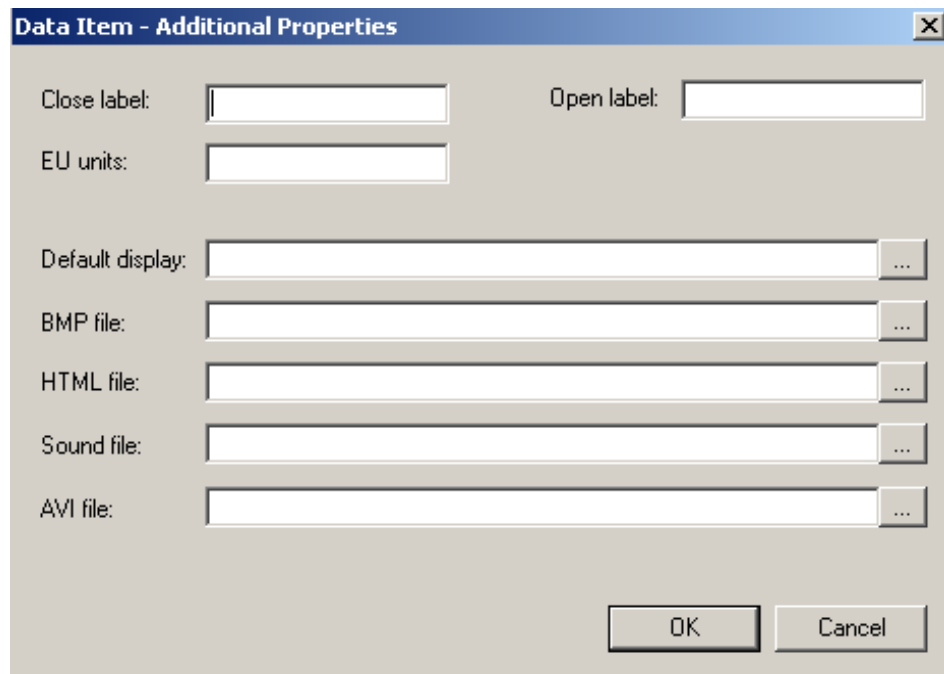
Address Space Logical Data Item

- **Location Type:** Location type is a type of a register in the device. Device registers are divided into **Coils**, **Inputs**, **Input Registers** and **Holding Registers**. The table below explains the name conventions used:

	<u>Read Only</u>	<u>Read / Write</u>
1 bit	Input	Coil
16 bit	Input Register	Holding Register

- **Modbus Type:** The location type (device data) will be understood as Modbus type (OPC data type). Modbus data type also depends on the Location type selected.
Example: Coil or Input (1 bit) device data type could be Modbus BOOL only.
When selecting Modbus String type, you must specify the data length (how many bytes will the String be represented by).
- **Simulation:** To test the client functionality, choose a **Simulation Signal** from the **Signal** drop-down list and check the **Simulate** check box. See the "Simulation Signals" section for information about creating simulation signals. All levels in the Address Space (port, device, folder, data item) support the process of simulation (**Simulate** check box). The parent list in the tree is superior; it has a higher priority when deciding to simulate the data item or not. In other words, a data item is simulated, if it itself has a simulation selected, or if any of its parents has the **Simulate** check box checked. (It may be simulated even if its **Simulate** check box stays unchecked)
- **Manual:** If checked, the data item will offer constant parameter value, because **Manual** setting is of the highest priority. The changes in the configuration take effect only when the server reloads the configuration (on startup).
- **Starting Address:** This value specifies the data item address (register number) in the device data space. With the UINT Modbus type, it is possible to extract bits from the register and use them as a Boolean or integer value (this functionality is read-only). You can specify a group of **Count** adjacent bits inside a word starting with **Bit #**. This way, it is possible to use a register for several separate data items.
- **Use Conversion:** To get the data value converted according to a prescribed form, choose one of the predefined or user-defined conversions. See the "Conversions" section for more details.
- **Generate Alarms:** Check the **Generate Alarms** check box to make the server generate alarms based on the data item value. The **Message Prefix** parameter is the text of the message for this data item; it will be followed by the text configured for a particular alarm type. The second part of the alarm message will contain the **Message Body** string (see **Alarm Definitions**). The server allows having any number of alarm definitions (templates) predefined. You can then combine one of them with the specific tags.

Clicking the **Additional properties** button opens the **Additional Properties** dialog box, shown below, which allows you to set a textual string for an Open/Close label, a Default display, a .bmp file, an HTML file, a sound file, and an .avi file. These properties are also accessible for OPC Data Access clients. For example, to see the HTML file name being presented in the OPC client, you should append **.HTMLFile** to the OPC Item name.



Data Item - Additional Properties

Close label: Open label:

EU units:

Default display: ...

BMP file: ...

HTML file: ...

Sound file: ...

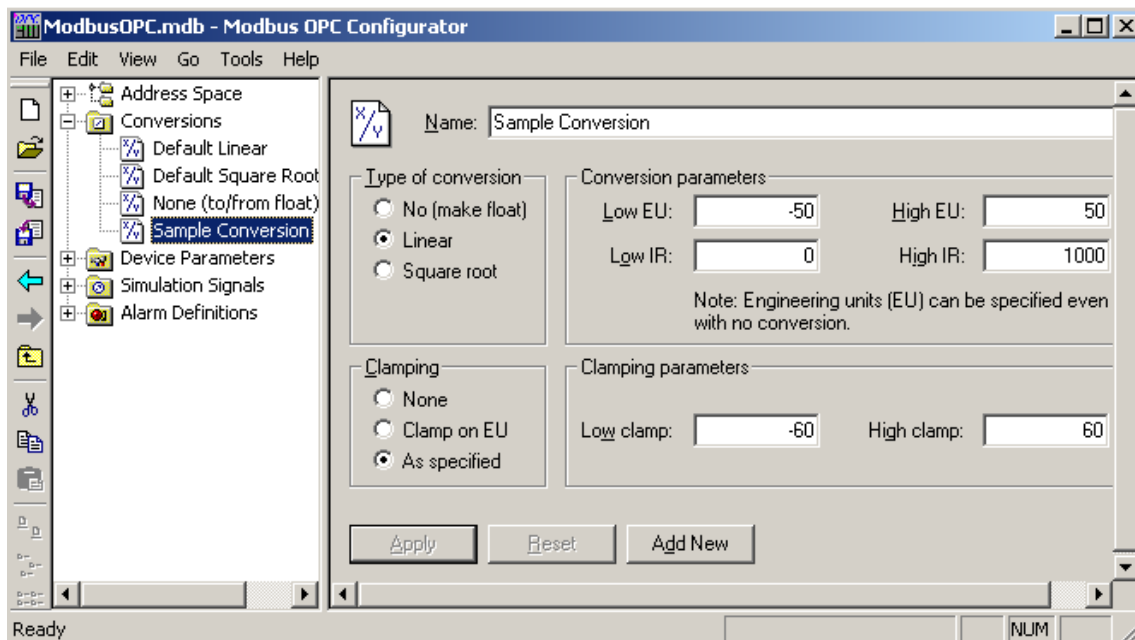
AVI file: ...

OK Cancel

Additional Properties Dialog Box

Conversions

You can tell the server to convert device data value simply by setting the **Conversion** properties, as shown in the figure below.



ModbusOPC.mdb - Modbus OPC Configurator

File Edit View Go Tools Help

Address Space
Conversions
 Default Linear
 Default Square Root
 None (to/from float)
 Sample Conversion
Device Parameters
Simulation Signals
Alarm Definitions

Name: Sample Conversion

Type of conversion
☐ No (make float)
☒ Linear
☐ Square root

Conversion parameters
 Low EU: -50 High EU: 50
 Low IR: 0 High IR: 1000
 Note: Engineering units (EU) can be specified even with no conversion.

Clamping
☐ None
☐ Clamp on EU
☒ As specified

Clamping parameters
 Low clamp: -60 High clamp: 60

Apply Reset Add New

Ready

Conversion Properties

Type of conversion

There are two types of units:

- **EU** engineering unit (client scale)
- **IR** instrument range (device scale)

No conversion converts the data into float data type, but does not change the value itself.

Linear or square root conversions keep a linear or square root relation between EU and IR.

Conversion parameters

Note that definition of range limits helps some client applications (e.g. GraphWorX, when creating slider) and makes sense even when no conversion is specified.

Clamping

If clamping is on, the data value will be limited to its High clamp/EU value, when it exceeds the upper limit, and similarly with the Low clamp parameter.

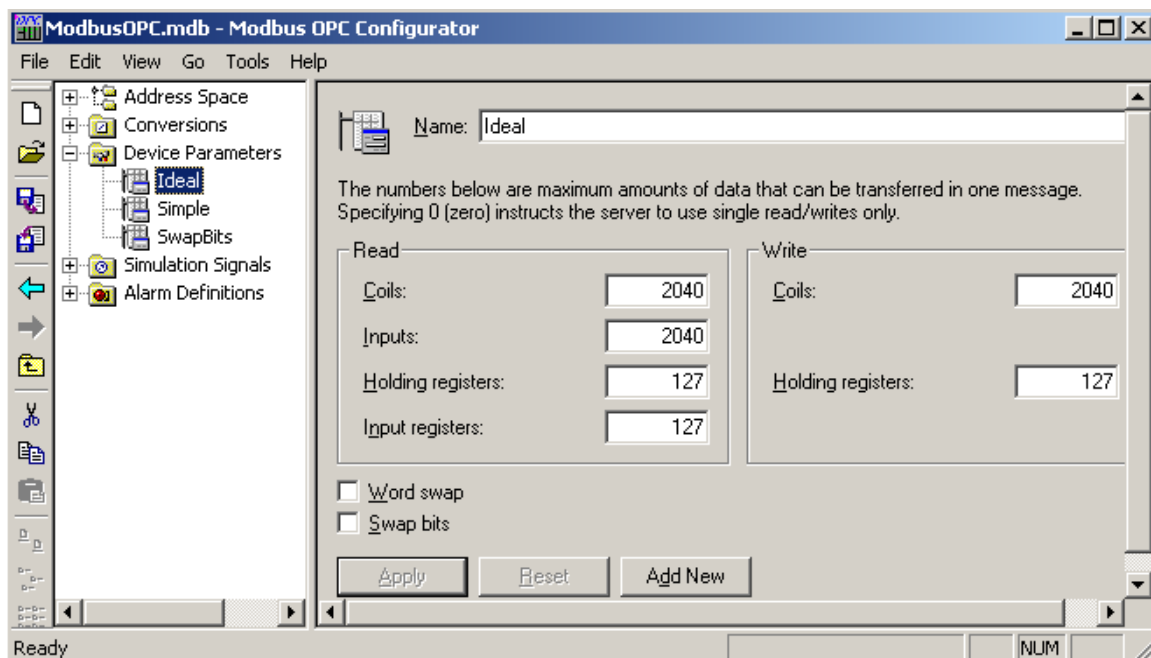
Device Parameters

The **Device Parameters** directory contains the list of custom device types. Device parameters influence the behavior and performance of the server for the device.

The meaning of numbers in the device parameters dialog, shown below, is the maximum amount of data that can be transferred in one message. Setting the value equal to zero forces the server to use single read/write messages only.

The server also allows you to configure the following additional properties for each device type:

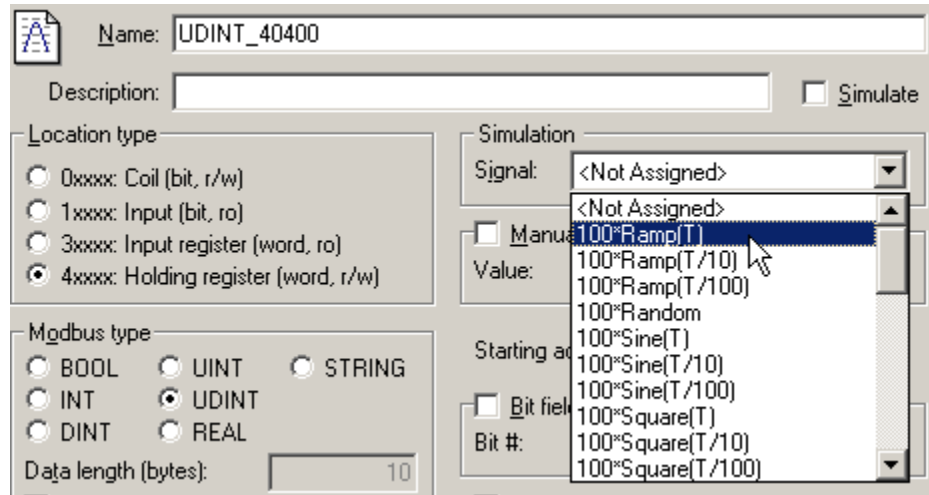
- **Word swap:** Swaps the first word with the second when reading/writing DINT, UDINT or float values.
- **Reverse bits:** Reverses the order of bits in word-sized values (least significant bit becomes the most significant bit).



Device Parameters

Simulation Signals

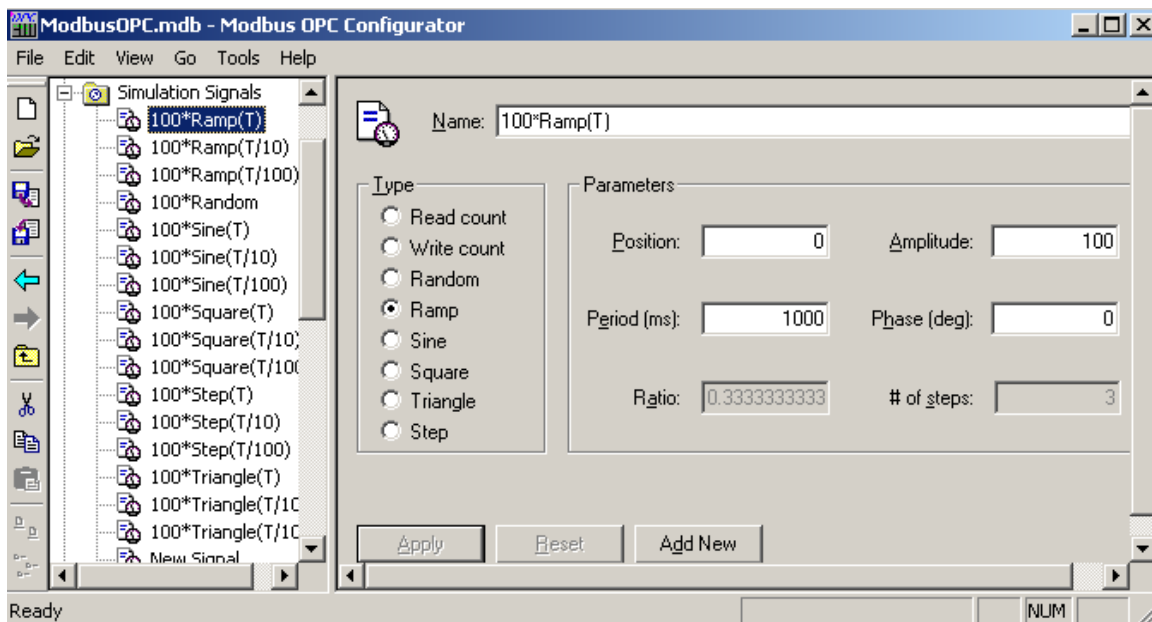
To test the client functionality, choose a **Simulation Signal** from the **Signal** drop-down list, as shown for the data item below. All levels in the **Address Space** folder support the process of simulation. The parent list in the tree control is superior; it has higher priority when deciding to simulate the data item or not.



Choosing a Simulation Signal

A large variety of simulation signals offered in the **Simulation Signals** folder on the tree control, as shown below. Each signal type has specific parameters, as shown below.

- **Read Count** is incremented by one each time the data item is read.
- **Write Count** increments when the data item is written.
- **Random** generates random a value within the **Amplitude** range starting with **Position**.
- **Ramp**, **Sine**, **Square**, **Triangle** and **Step** are periodical signals. Their time behavior is influenced by the **Period** and **Phase** parameters. **Period** specifies the signal frequency, while **Phase** moves the signal origin on the time axis.
- **Square** and **Triangle** signal types have one more parameter: **Ratio**. **Ratio** defines Triangle signal steepness, or Square signal H/L proportions.
- The **# of steps** parameter of the **Step** signal defines a number of steps signal amplitude will be divided into.



Simulation Signal Parameters

Alarm Definitions

Alarm definitions are divided into two alarm template types: **Digital** alarm definitions and **Limit** (analog) alarm definitions. Digital alarms can be defined for Logical data items, while Limit alarms can be defined for Numeric data types. Alarms cannot be defined for Textual data items. Default alarm definitions are provided in the "ModbusOPC.mdb" database.

Limit Alarms

To create a new limit alarm definition, right-click the **Alarm Definitions** folder on the tree control of the Configurator screen and select **New - Limit Alarm Definition** from the pop-up menu.

A limit alarm sets the values for four levels of alarms: **LoLo**, **Lo**, **Hi**, and **HiHi**. Limit alarm parameters can have subranges within the data item amplitude. Every subrange definition includes a **Message Body** that will be appended to the alarm message, the **Severity** of the alarm, and the **Req. Ack.** flag for alarm acknowledgement.

Deadband prevents the server from generating a huge amount of alarm messages and overloading the clients when the signal oscillates around one of the limits specified. The deadband value extends the limit zone. It results in sending only one alarm message even if the signal oscillates.

Name: Default Limit Alarm (0-10-90-100)

Update rate: 2000 ms Deadband: 1

Limit alarm definition

Limit	Value	Message Body	Severity	Req. Ack.
☒ HiHi	100	HiHi Level Alarm	850	☐
☒ Hi	90	Hi Level Alarm	500	☐
☐ Return to normal		Return to Normal		☐
☒ Lo	10	Lo Level Alarm	500	☐
☒ LoLo	0	LoLo Level Alarm	850	☐

Apply Reset Add New

Limit Alarm Definition

Digital Alarms

A digital alarm sets an alarm if the comparison between the **Alarm State Value** and the input state is TRUE. To create a new digital alarm definition, right-click the **Alarm Definitions** folder on the tree control of the Configurator screen and select **New - Digital Alarm Definition** from the pop-up menu. The default digital alarm definition is shown below.

The **Update Rate** parameter defines the frequency of checking the data item value, and possibly responding with sending the alarm message. You can request the alarm when the value equals TRUE or FALSE, define the **Message Body** that carries the information, and set the **Severity**, which is the alarm message importance. The severity value ranges from 0 to 1000. Checking **Return to normal** will generate a separate alarm message when the data item gains the opposite value. Checking the **Req. Ack.** check box tells the client that the alarm requires an acknowledgement. Then the alarm message can be not only sent but also acknowledged.

Name: Default Digital Alarm (on TRUE)

Update rate: 2000 ms

Digital alarm definition

☒ Enable

Value: True (1) Message Body: Digital Alarm Severity: 500 Req. Ack.: ☒

☐ Return to normal Return to Normal

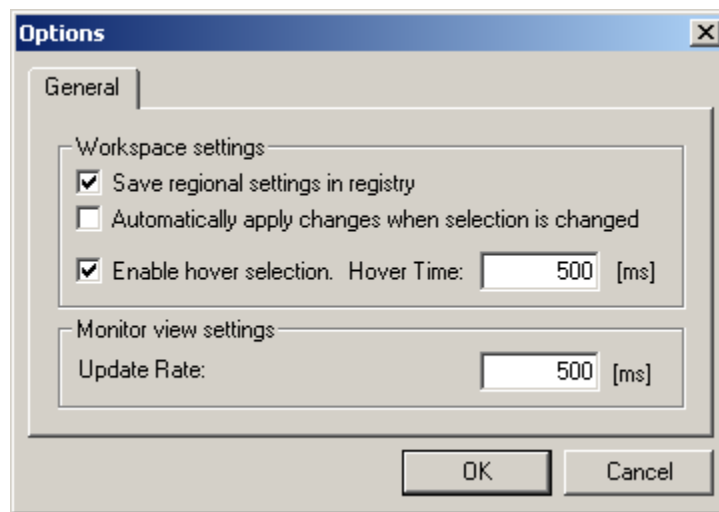
Apply Reset Add New

Digital Alarm Definition

Options

Selecting **Options** from the **Tools** menu opens the **Options** dialog box. The **General** tab of the **Options** dialog box, shown below, contains the following options:

- **Save regional settings in registry:** Checking this option allows you to save regional settings in the registry so that they are applied each time you start the configuration application. This applies to the language settings as well as time and date settings.
- **Automatically apply changes when selection is changed:** Checking this option allows changes to the configuration database to be saved each time you switch dialogs without clicking on the **Apply** button or being shown a message asking if you would like to apply changes.
- **Enable hover selection:** Checking this option allows you to highlight an item by moving the mouse pointer over that item and keeping it there for a specified amount of time (in milliseconds).
- **OPC Monitoring Update Rate:** This specifies the update frequency (in milliseconds) of the items in the Monitor View. The monitor scans the server and displays the tag values at the bottom of the Configurator screen.



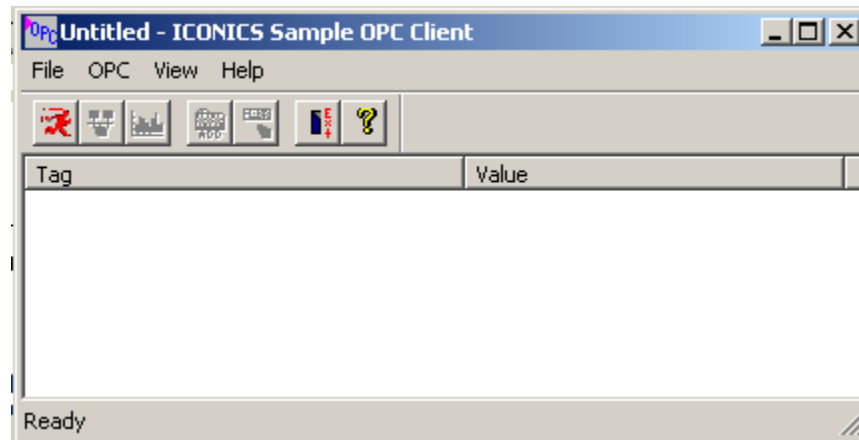
Options Dialog Box

OPC Server Runtime Module

The runtime module of the OPC Server gets started by the operating system when an OPC client tries to create an instance of the server. Upon startup, the server loads the currently active configuration created with the OPC Server Configurator. Runtime mode makes accessible items defined in the active configuration database. Also, some new items are created automatically (depending on the type).

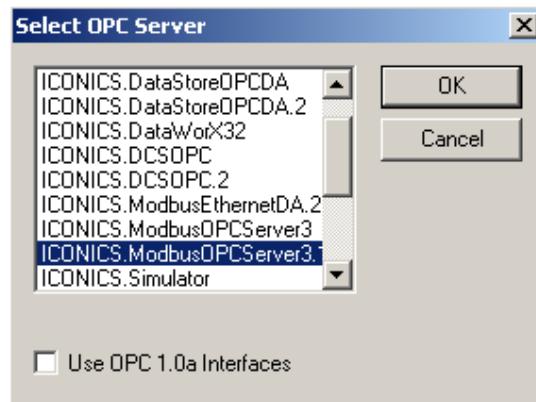
Sample OPC Client

The **Sample OPC Client** is used as a test case to debug servers and demonstrate some of the OPC capabilities. To open the Sample OPC Client from the Windows **Start** menu, select **Programs > ICONICS Modbus OPC Server 3.1 > Sample OPC Client**.



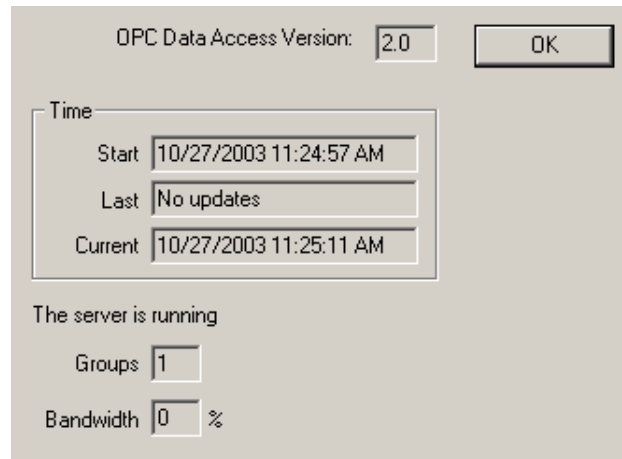
Sample OPC Client Screen

From the **OPC** menu, select **Connect**. The **Select OPC Server** dialog box appears showing all registered OPC servers, as shown below. Select **Modbus OPC Server** and then click **OK**. This connects the OPC client to the Modbus OPC Server.



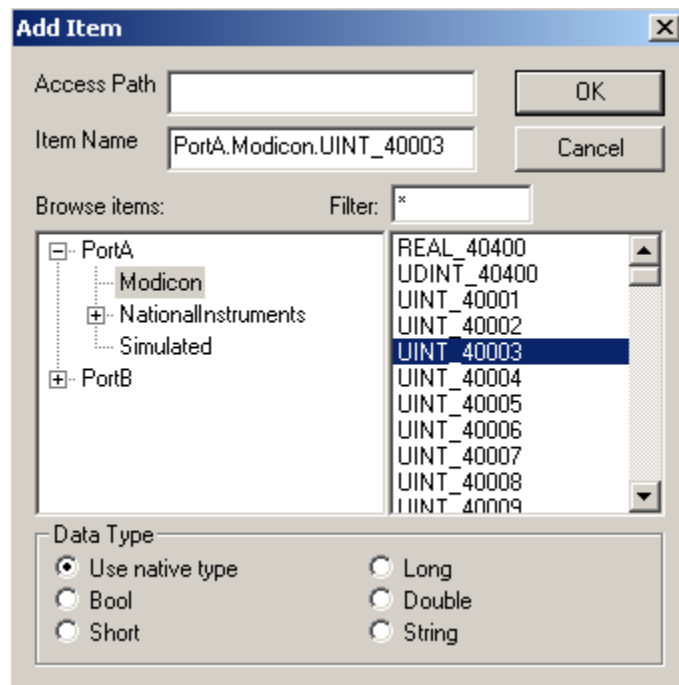
Select OPC Server Dialog Box

From the **OPC** menu, choose **Server Status** to view the OPC server status dialog box, shown in the figure below. The server status dialog box continuously updates the server start time, the time of last update, the current time, the number of groups, and the bandwidth for the selected server. Use this window to verify that the server is running and is connected to the OPC client. Click **OK** to close the server status dialog box.



Server Status Dialog Box

From the **OPC** menu, choose **Add Item**. The **Add Item** dialog box appears, as shown in the figure below. The **Add Item** dialog box displays the devices, data blocks, and data tags defined in the current server configuration. These are filtered by expression “*” and by data type. You can enter an access path, type, or select an item name, and choose the data type in which you want to view the requested tag. In the **Browse Items** list, select the device, group, and tag. The full name appears in the **Item Name** field. Click **OK**. The item appears as a **Tag** in the OPC Client window with updating data values. You can add more tags by repeating the steps described above.



Add Item Dialog Box

From the **OPC** menu, choose **Group Parameters**. This command displays the parameters of the group of tags.

Group Parameters for the Selected Server

The **Update Rate** determines how fast data are sent to the client (in milliseconds). The **Active** state (checked) initiates active communications between the client and the server. You can change the **Active** state or the **Update Rate**, for example, and click **Apply** to send changes to the server. If the group is not active, data are not sent to the client. You can choose to **Disconnect** from the **OPC** menu to restore all selections and to disconnect from the server.

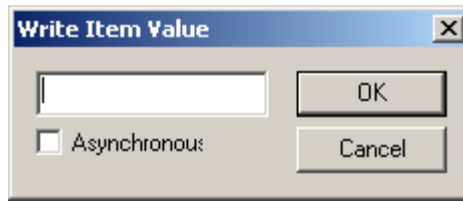
While a client is connected to a server, you cannot exit the server or open new configurations in the server. You can modify the existing server configuration while client(s) are connected; however, you cannot delete any device or tag in use. For example, you can take a device offline or change a tag's scaling values without shutting down a client that may be controlling many devices.

Start another instance of the OPC client and make the same server connection as those defined in the first client instance. You can add items, change the group update rate and active states, and view the server status in each instance of the OPC clients independently of the other instances. For example, the number of groups is "1" in each client because each one only has one group.

The only way that one client affects another is through data and data changes. Make sure both clients are showing the same tag.

To write a value to an item or tag, highlight a tag in the client and choose **Write Value to Item** from the **OPC** menu. This allows you to write a value for the data item, as shown in the figure below. Enter a numeric value and click **OK**. The client displays the new value at the update rate defined above. The client writes the value typed in the **Write Item Value** dialog box to the correct item in the Group object. The method of writing data is determined by the **Asynchronous** check box. When data are written synchronously, the call to the server blocks until the operation has completed, which may take a long time. When data are written asynchronously, the call to the server returns quickly and the operation is carried out on another thread. When the write has finished, the client's advise interface is notified with the results.

Note: A client defined with a very slow update rate does not receive the value immediately.



Writing a Value to a Data Item

How the Client Connects to and Gets Data From a Server

When OPC servers are installed on the system, they are registered so that clients can find them and users can view them in a list. To make a client/server connection, the user must first choose a server. The server name selected is used to get a class ID which, in turn, is used to create a COM object. If the user selects a server that is not currently running, the system starts the server and then creates the object in that server on behalf of the client.

Once the COM object is created, the client application has an IOPCServer interface, as defined in the OPC specification. This is the main interface to an OPC server, and this document refers to this as the Server interface. (The other interface is Group.) An OPC client always creates a group in the server and registers an advise interface with it to get asynchronous data notification.

When an item is added, the dialog displays either a hierarchy or a flat list of names, depending on the server. The names are queried from the Server object using the filter string and requested data type. When the user either types or selects a name, an item by that name is added to the Group object. This item is also read immediately to get an initial value.

Items in a group are scanned by the server; when their values change, the advise interface in the client is notified. A data structure containing the data for each item whose value or quality has changed (and only those items) is passed to the client's advise interface. The client unpacks and uses the data.

Server Status and Group Parameters

The Server status dialog periodically queries the Server object for status and displays the results. The **Group Parameters** dialog box queries the group state and displays the results. The **OK** button and **Apply** button in the **Group Parameters** dialog box write the parameters from the dialog to the Group object and read the parameters back. This primarily controls the scanning and updating of data. When a group is inactive, it does not send data notifications to the client and, typically there is no reason to scan the items when the group is inactive. The **Update Rate** in the **Group Parameters** dialog box specifies the rate at which data notifications should be sent back to the client, assuming that data have changed. This is also the rate at which items are scanned on behalf of this client.

Disconnecting the Client from the Server

You can disconnect from the selected server by choosing **Disconnect** from the OPC menu. The **Disconnect** command unregisters the advise interface, releases the Group object, and then releases the Server object. Releasing an object means that the client is no longer interested in the object, so the system can delete the object in the server.

Modbus Runtime Module

The Modbus OPC Server is the actual runtime module. The Modbus OPC Server is responsible for accessing the data on the OPC clients' requests. The OPC server has no user interface. It will be started automatically when the OPC client connects to the server.

The runtime module operates directly with the database. Read and write operations are performed directly with no intermediate cache level. It has the effect that every client works with accurate data. In normal situations, the data in the database will be accessed (read/write) by OPC clients. However, the data are stored in a standard database, and other software applications can manipulate its data as well through programming techniques such as Microsoft's ADO (ActiveX Data Objects) or OLE DB. The Modbus OPC Server refreshes (with a configurable delay) its tags in order to inform the OPC clients of changes through some external application.

One of the basic concepts of the Modbus OPC server is that monitored data are relatively stable in time. The runtime module resides in the ICONICS/Modbus OPC Server 3.0 directory.

Minimum Update Rate Settings

The server updates the tag values periodically. In normal cases, this setting is not important since all OPC clients access the data through the Modbus OPC Server and all clients will always see the most recent values. But, in some cases when there may be a third-party application writing to the database directly, you may want to change this setting.

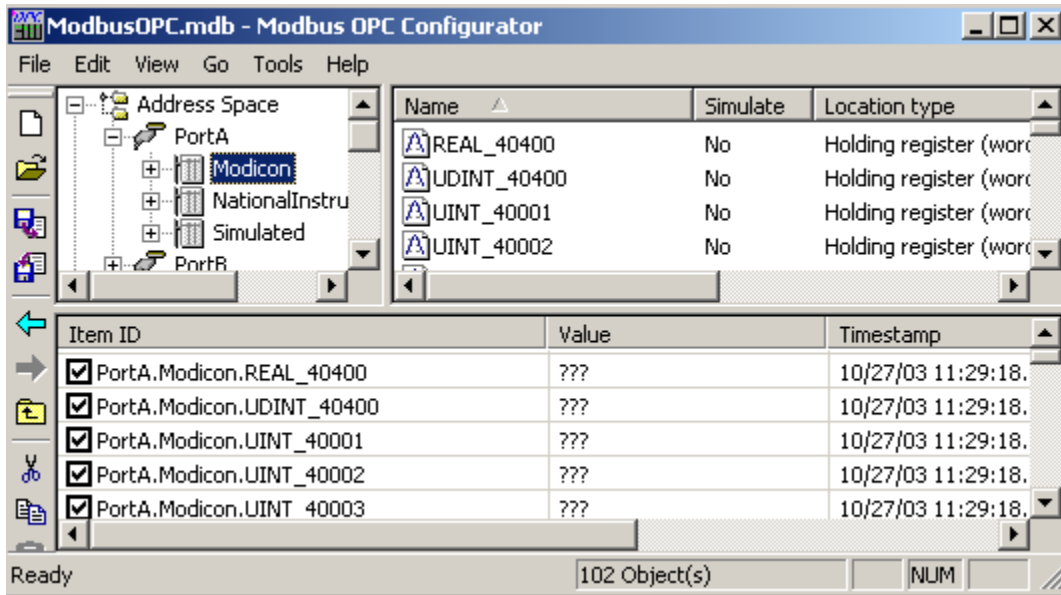
Active Configuration Database File Setting

The runtime module loads its configuration from the active configuration database. The Configurator sets this setting by the **Make Active** command on the **File** menu of the Modbus Configurator.

Monitor View

The Configurator includes a runtime monitor for viewing server data. To change to the monitor view, select **Monitor View** from the **View** menu. The runtime monitor appears in the bottom pane of the Configurator screen, as shown in the figure below. During runtime, the monitor scans the server and displays the tag values and other statistics such as date, time, and quality.

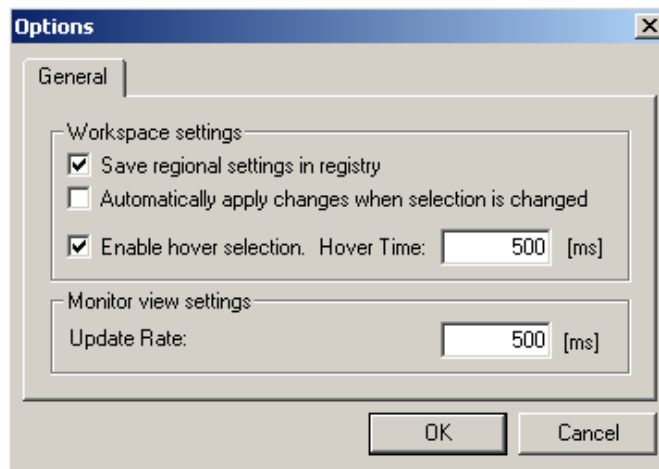
Monitoring is enabled for each item with a check mark next to it. To enable/disable monitoring for an item, you can click on the box to the left of the item. A check mark inside the box means the item is enabled for monitoring. If there is no check mark, then the item is disabled.



Runtime Monitor View

Setting the Monitor View Refresh Rate

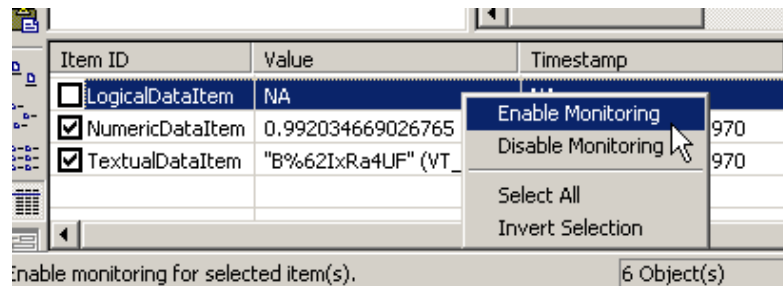
To set the **OPC Monitoring Update Rate** (in milliseconds), select **Options** from the **Tools** menu and click on the **General** tab of the **Options** dialog box, as shown in the figure below.



Options Dialog Box: General Tab

Enable Monitoring

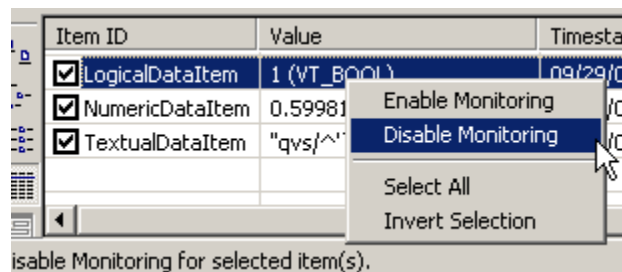
Monitoring is enabled for each item with a check mark next to it. To enable/disable monitoring for an item, you can click on the box to the left of the item. A check mark inside the box means the item is enabled for monitoring. If there is no check mark, then the item is disabled. To enable monitoring for an unchecked item in the monitor view, you can also right-click on the item and select **Enable Monitoring** from the pop-up menu, as shown in the figure below.



Enabling Monitoring for an Item

Disable Monitoring

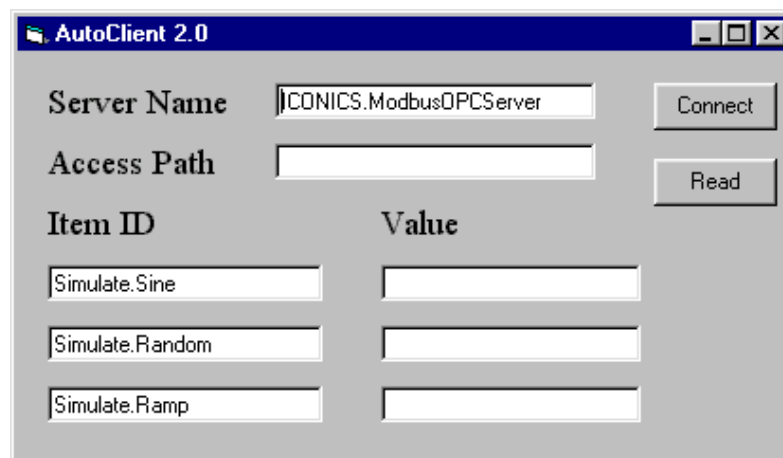
Monitoring is enabled for each item with a check mark next to it. To enable/disable monitoring for an item, you can click on the box to the left of the item. A check mark inside the box means the item is enabled for monitoring. If there is no check mark, then the item is disabled. To disable monitoring for a checked item in the monitor view, you can also right-click on the item and select **Disable Monitoring** from the pop-up menu, as shown in the figure below.



Disabling Monitoring for an Item

Running Sample OPC Automation Client

Run the Visual Basic project "Simple." This project has a form shown in the figure below.



Visual Basic Application Reading OPC Data

To connect to an OPC server and start reading data, follow the steps below:

1. Type the name of the server you need to connect to, along with the access path and item ID for up the three names.
2. Click **Connect**. The application reads the server name, creates the server's object, and connects to it.
3. Click **Read**, which will read the access path and item ID and starts reading data from the OPC server for all items requested, as shown in the figure below.
4. Toggle the **READ OFF/ON** button to stop and start reading from the server.
5. Click **Disconnect** to stop the communication and disconnect from the OPC server. Type another server and/or item names and click **Connect** to communicate with another source.

The screenshot shows the 'AutoClient 2.0' application window. It has a title bar with standard Windows window controls. The main area contains several input fields and buttons. At the top, there's a 'Server Name' field with the text 'ICONICS.ModbusOPCServer' and a 'Disconnect' button to its right. Below that is an 'Access Path' field, which is currently empty, and a 'READ OFF' button to its right. In the center, there's a table with two columns: 'Item ID' and 'Value'. The table contains three rows of data: 'Simulate.Sine' with value '31', 'Simulate.Random' with value '37', and 'Simulate.Ramp' with value '41'.

Item ID	Value
Simulate.Sine	31
Simulate.Random	37
Simulate.Ramp	41

Reading OPC Live Data

This project demonstrates how easy it is to connect to an OPC Custom interface server and read data from it. All the source codes and project's resources are available with ICONICS OPC ToolWorX.



Your comments and suggestions on the operation of this software are welcome. Please address them to:

ICONICS
100 Foxborough Blvd.
Foxborough, MA 02035
Phone: 508-543-8600
Fax: 508-543-1503
E-mail: support@iconics.com
Web: www.iconics.com
[ICONICS Technical Support](#)

© ICONICS, Inc. All rights reserved.

Introduction to OPCAdmin

OPCAdmin is a tool that you can use to connect to your running OPC server that was built with OPC ToolWorX, and see (possibly also change) all kinds of statistical, debugging, internal, runtime, and configuration parameters and counters.

OPCAdmin is installed as a part of the OPC ToolWorX installation into the Tools subdirectory.

Connecting OPC Server

To connect an OPC server with the OPCAdmin utility, the OPC server must support special custom COM interfaces. These interfaces are contained in OPC ToolWorX (from version 3.1). New OPC servers created with the OPC ToolWorX 3.1 (and later) will automatically contain this support (interfaces).

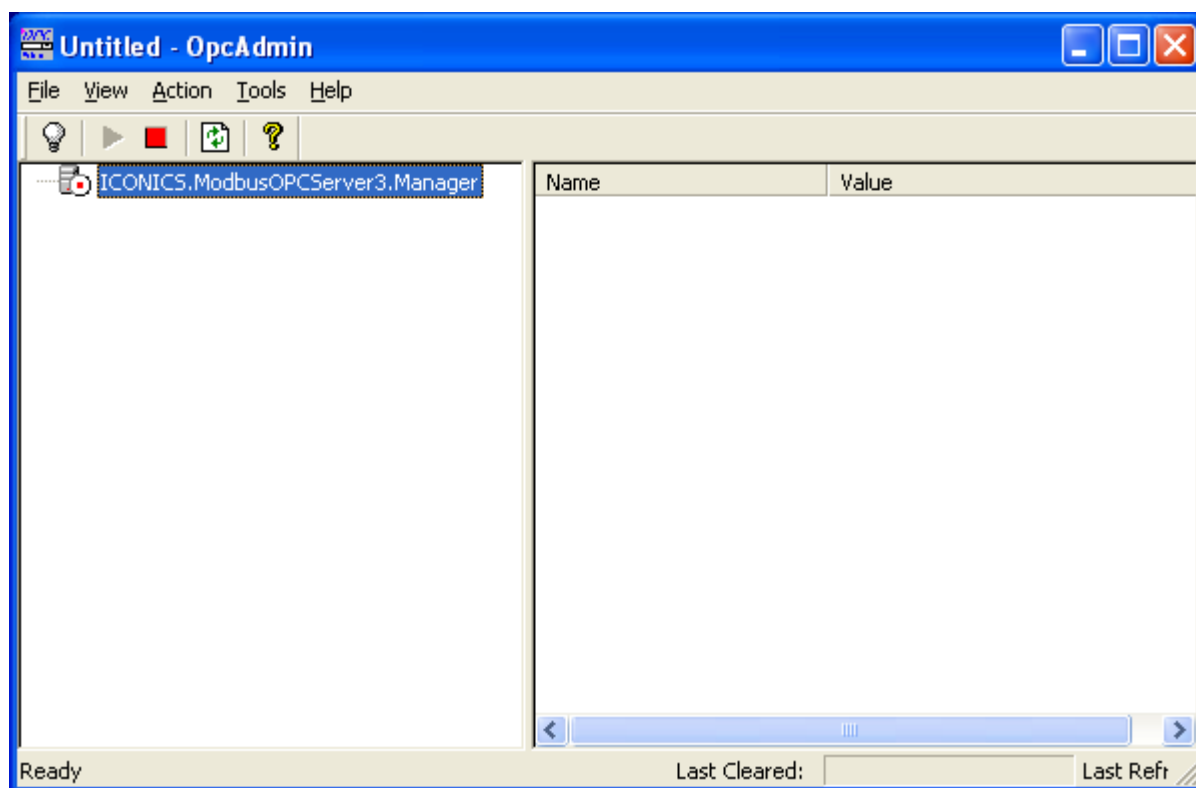
To connect to the OPC server, the ProgID of the “OPC Manager” COM creatable object must be specified as a program argument when the OPCAdmin is executed. The ProgID is a human-readable unique identifier for the COM component.

For instance, for the ICONICS Modbus OPC server the OPCAdmin should be launched:

```
OPCAdmin.exe ICONICS.ModbusOPCServer3.Manager
```

OPCAdmin User Interface

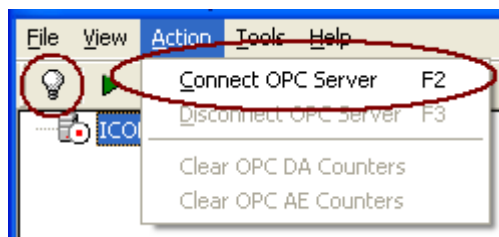
When you start the OPCAdmin with the OPC server, it will display the default application window, as shown in the figure below.



OPCAdmin User Interface

Connecting to OPC Server

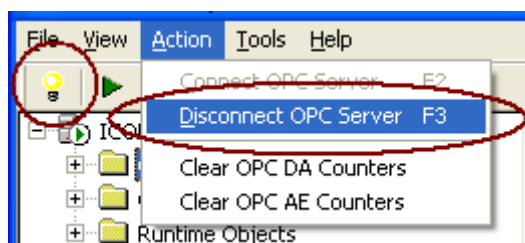
To start obtaining data from the server, the “Connect” operation to the server must be done. To do this, you can use the **Action → Connect OPC Server** command, press the F2 key, or click on the “bulb” icon, as shown in the figure below.



Connecting to OPC Server

Disconnecting OPC Server

To disconnect from the OPC server, you can use the **Action → Disconnect OPC Server** command, press the F3 key, or click on the “bulb” icon, as shown in the figure below.

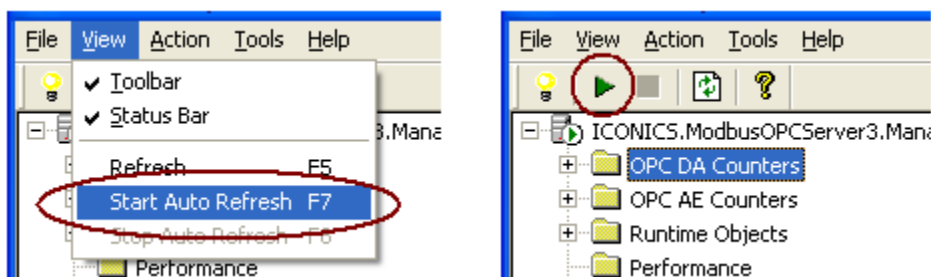


Disconnecting OPC Server

Auto Refresh and Manual Refresh

The OPCAdmin application can use two methods to get fresh data from the OPC server. It can get data automatically, with some period (Auto Refresh mode), or it can get data only when the Refresh (manual refresh) command is performed.

To switch the Auto Refresh mode on, use the **View → Start Auto Refresh** command, press the F7 key, or click on the “start” icon, as shown in the figure below.



Start Auto Refresh

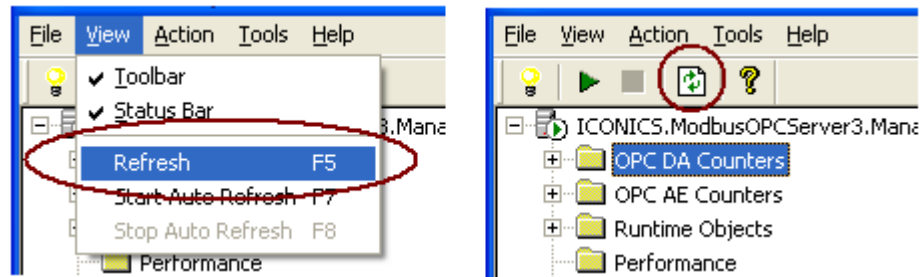
You can specify the period for automatic refresh under **Tools → Options → Refresh Rate**.

To switch the Auto Refresh mode off, use the **View → Stop Auto Refresh** command, press F8 key, or click on the “stop” icon, as shown in the figure below.



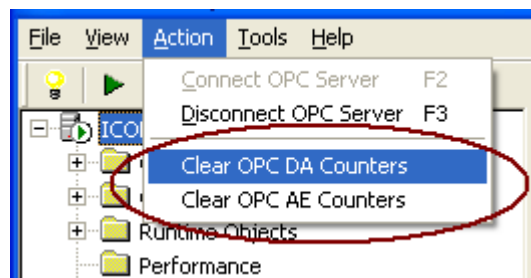
Stop Auto Refresh

To perform a manual refresh, use the **View → Refresh** command, press the F5 key, or click on the “refresh” icon, as shown in the figure below.

**Manual Refresh**

Clearing OPC DA and OPC AE Counters

You can tell the OPC server to zero its OPC DA or OPC AE counters. This can be very handy when debugging the server to see a number of certain OPC calls made during some period of time. To clear the counters, use the **Action → Clear OPC DA Counters** or **Action → Clear OPC AE Counters** command, as shown in the figure below.

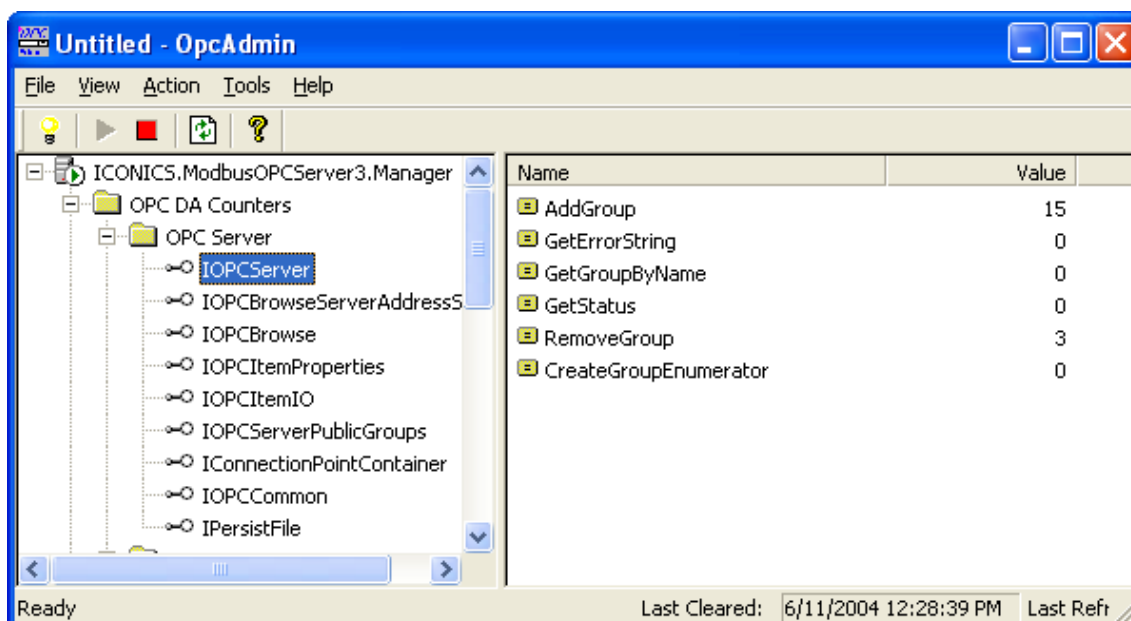
**Clearing OPC DA and OPC AE Counters**

OPCAdmin Information categories

This section describes the categories of data that the OPCAdmin retrieves from an OPC servers and displays for user

OPC DA Counters

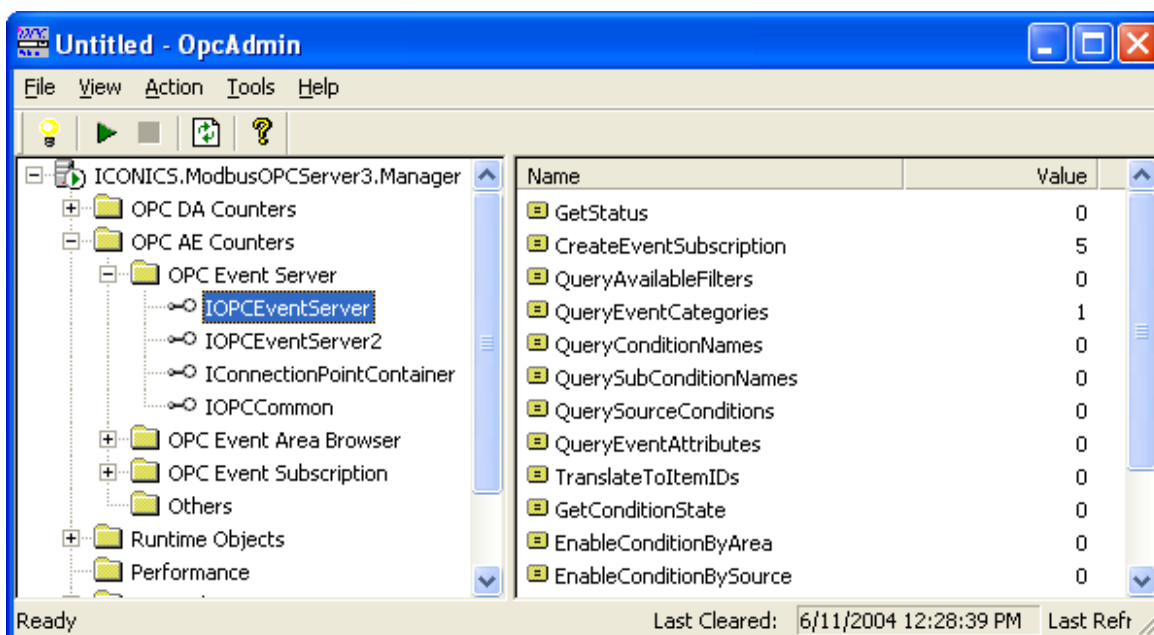
The OPC DA Counters tree displays counters for all OPC DA interfaces and methods, categorized into the objects (OPC Server, OPC Group) defined by OPC specifications for Data Access servers, as shown in the figure below. Information under these objects is further subdivided based on the DA Interfaces.



OPC DA Counters

OPC AE Counters

The OPC AE Counters tree displays counters for all OPC AE interfaces and methods, categorized into the objects (OPC Event Server, OPC Event Subscription, OPC Event Area Browser) defined by OPC specifications for Alarms and Events servers, as shown in the figure below. Information under these objects is further subdivided based on the AE Interfaces.

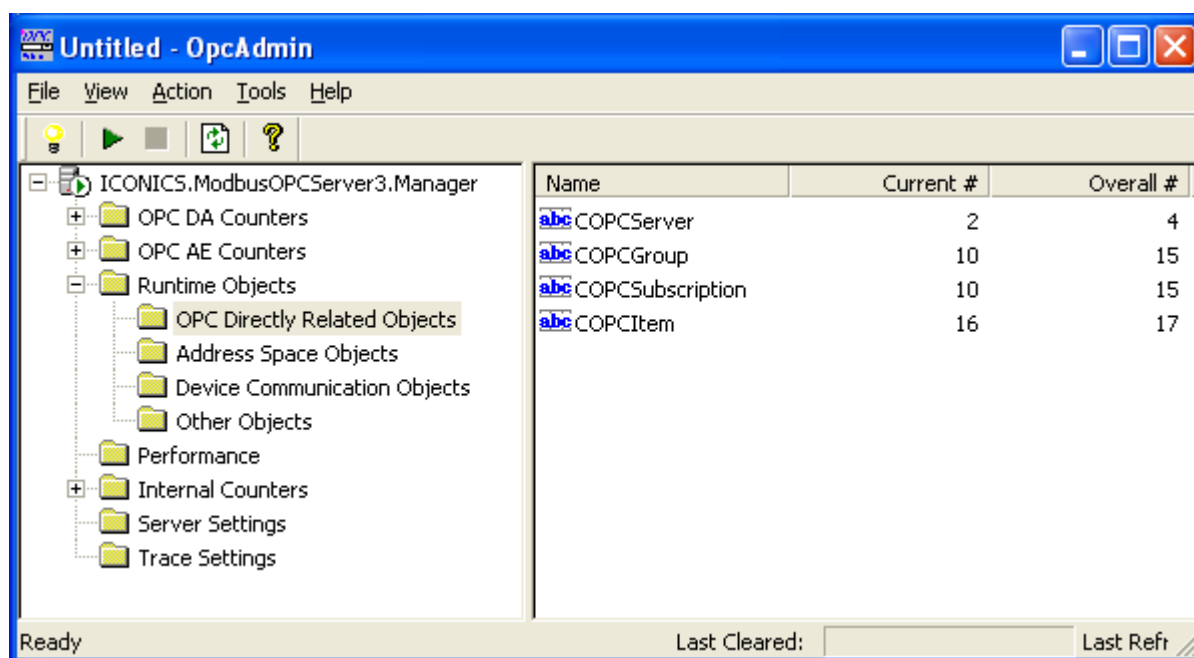


OPC AE Counters

Runtime Objects

The Runtime Objects tree displays numbers of objects of important internal toolkit's C++ classes created in the server, as shown in the figure below. It is subdivided into several categories. There are two values displayed for each item:

- Current #: number of objects of current class that are created and exist at this moment.
- Overall #: number of objects of current class that were created up to this moment, including the ones that were already deleted



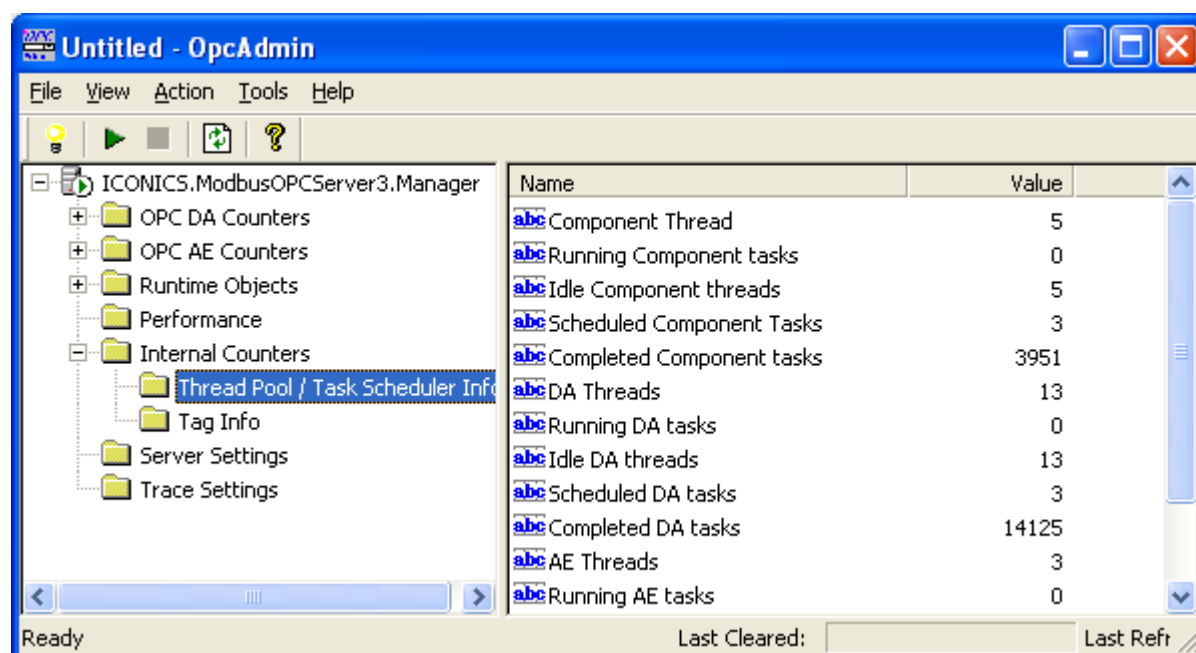
Runtime Objects

Performance

The Performance tree contains counters that are related to the performance of the server (not in version 3.1).

Internal Counters

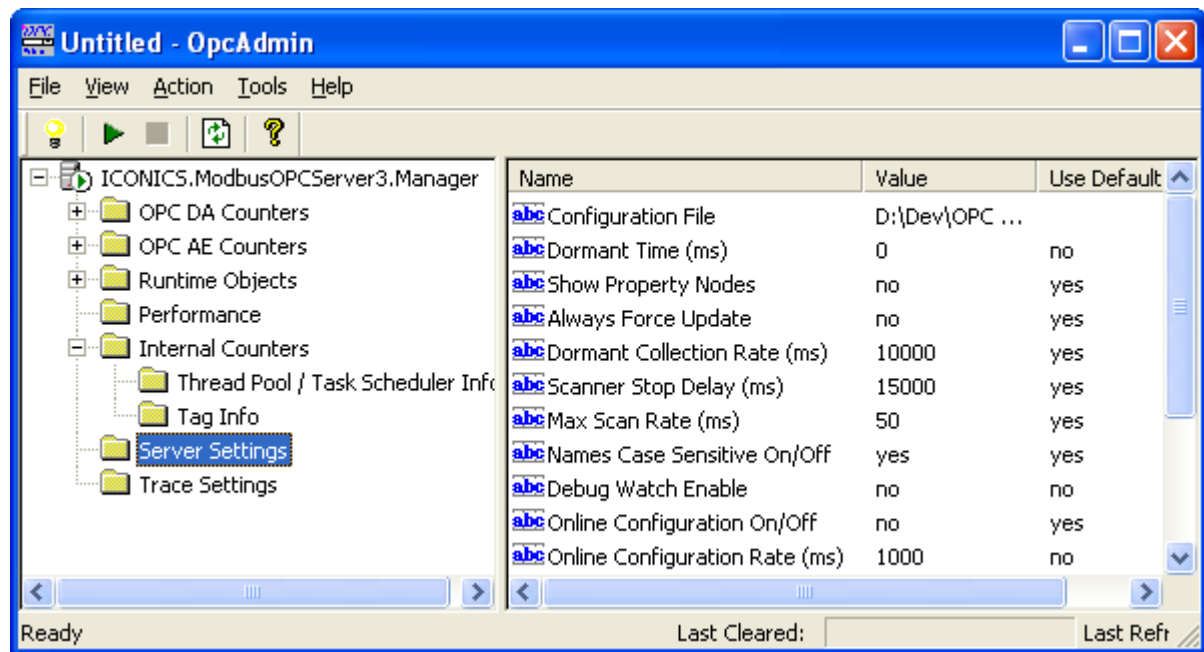
The Internal Counters tree contains other informative counters that the servers provide, and that do not fall into any of previous categories, as shown in the figure below. These are all highly internal counters, and can be used for debugging purposes.



Internal Counters

Server Settings

In the Server Settings tree you can find most of the internal setting for the server, as shown in the figure below. You can modify the values of each setting. There is also a visible notification whether the setting is set to its default. You can also tell any setting to use its default value. See more information on server settings in the OPC ToolWorX User's Guide, section 4.1.9.2 - Server's internal settings.



Server Settings

Trace Settings

In the Trace Settings tree you can find the settings for TraceWorX32 module that the OPC servers use for tracing into XML file. See more information on TraceWorX32 settings in OPC ToolWorX User's Guide, section 6.1.1 - TraceWorX32 logging module, or in separate Appendix describing the TraceWorX32 in more detail.