Giddings & Lewis OPC Server

User Manual

Version 4.0

Giddings & Lewis
Controls, Measurement and Sensing

NOTE

Progress is an on going commitment at Giddings & Lewis. We continually strive to offer the most advanced products in the industry; therefore, information in this document is subject to change without notice. The illustrations and specifications are not binding in detail. Giddings & Lewis shall not be liable for any technical or editorial omissions occurring in this document, nor for any consequential or incidental damages resulting from the use of this document.

DO NOT ATTEMPT to use any Giddings & Lewis product until the use of such product is completely understood. It is the responsibility of the user to make certain proper operation practices are understood. Giddings & Lewis products should be used only by qualified personnel and for the express purpose for which said products were designed.

Should information not covered in this document be required, contact the Customer Service Department, Giddings & Lewis, 660 South Military Road, P.O. Box 1658, Fond du Lac, WI 54936-1658. Giddings & Lewis can be reached by telephone at (920) 921–7100.

The Giddings & Lewis OPC Server was developed for Giddings & Lewis by DASTEC Corporation. It is based on the OPC Toolkit from FactorySoft.

This manual is not available in hardcopy.

M.1301.0175

Release 2002

Table of Contents . . . .

Giddings & Lewis OPC Server

1. INTRODUCTION

This is a User Manual for the Giddings & Lewis OPC Server. The Giddings & Lewis OPC Server is referred to simply as the Server throughout this manual. The Server communicates with Giddings & Lewis controllers over Ethernet and supports data exchange with Client's via Microsoft's Object Linking and Embedding (OLE) for Process Control (OPC).

The manual is organized to give an overview of OPC technology and the Server, details on the configuration environment, detailed procedurally steps involved in a simple configuration sequence and then information on quality information, error information and troubleshooting.

While an overview of OPC is presented, a complete description of OPC is beyond the scope of this manual. References are provided where additional details can be found. While an in-depth understanding of OPC is not required to use the Server with OPC Clients, a good understanding of OPC can help you optimize configurations and may prove invaluable when trying to get multiple Clients working with a single Server instance.

Terms used in the manual that are capitalized are defined terms. You will find the definitions for these terms in Section 18, "*Definitions*".

This page is left blank intentionally.

2. OVERVIEW

2.1 Object Linking and Embedding for Process Control (OPC)

OPC is an acronym for OLE for Process Control. OLE is an acronym for Object Linking and Embedding. OLE is a Microsoft technology. OLE is available on Microsoft Windows 95/98 and Microsoft Windows NT operating systems. OLE is based on Distributed Component Object Modeling or DCOM. DCOM allows Clients to make use of an OPC server on a remote computer via a network. DCOM is part of Windows NT and can be added to Windows 95/98.

OPC is a standard written by the OPC Foundation for use in the manufacturing environment. It defines the way OLE is used so OPC Clients can interact with OPC Servers.

This specification details the methods of exchanging data between data sources (servers) and data users (clients). The data exchange between servers and clients is possible between servers and clients residing on the same computer (local server) and between servers and clients residing on different computers over a network via Distribute Communication Object Modeling (DCOM) (remote servers) .

The OPC Server is implemented as an out of process server, i.e., an executable program, rather than an in process DLL. Only one Server should be run on any one computer at a time.

2.2 Giddings & Lewis OPC Server

The OPC Server reads and writes data to and from controllers via Ethernet. The controller must have an Ethernet TCP/IP communication module installed and the ladder must have the proper communication Function Blocks programmed. See Section 4, “Requirements” for system requirements. This OPC Server meets the OPC Data Server Specification Version 3.0. It also supports OPC Version 1.0.

The Server has a graphical user interface (GUI) configuration environment with an "Explorer" look and feel. The configuration environment allows the Server to be configured with information such as controller IP addresses and available global variables so that the Server can communicate with these devices on behalf of Clients.

Data exchanged between the Server and configured controllers are always with the global variables that were defined during the controller’s programming in PiCPro™. PiCPro™ generates a file that contains all of the variables declared as global. The Server reads these files in as part of the configuration process generating matching Tags in its configuration. Clients and the Server then reference tag data via these global variables assigned in PiCPro™.

The Server was developed by DASTEC Corporation using the OPC Server Toolkit from FactorySoft, Inc..

2.3 Overview Of Operation

2.3.1 Configuration

The Server must be configured to know about the controller(s) with which it will be asked to communicate. In the Server configuration environment, each controller is referred to as a Device. Configuration of Devices includes defining Communication Paths to access controller(s), controller address information and the Global Variables available in the controller(s) to be accessed. The Global Variables available for reading and writing in each controller are configured by importing a file that was generated by PiCPro™ when the program was downloaded to the controller. This file is called an OID File. It contains information on all of the variables in the controller that were defined as global. When the Server imports an OID File, it creates a Server Tag for each of the global variable points in the controller. These Tags can then be accessed by Clients.

2.3.2 Client Connections, Groups and Items

An OPC Client makes use of the Server by first connecting to it, either locally or remotely over a network. To connect to a Server, a Client specifies the node on which the Server is located, if the Server is not on the same computer as the Client (local), and the Server’s registered name, **G&L.OPCServer.4** or **G&L.OPCServer**. If the Server is not already running, the

operating system of the computer that will run the Server will start the program. When the Server starts, it automatically loads the last configuration it used. Clients cannot select a configuration for the Server.

Once connected, the Client makes requests of the Server to do things on its behalf. In order for a Client to get the Server to write or read data to a controller on its behalf, it first creates one or more Client Groups in the Server. When the Client asks the Server to create a Client Group on its behalf, it also request certain attributes be set for the Client Group such as a name, whether or not the Server should scan data for the Client Group and if so at what rate. The Client can change most of these attributes at anytime by making a request to the Server. Client Groups are maintained in the Server on behalf of the Client.

Once the Client has a Client Group established, it requests that the Server add one or more Items to the group. An Item is associated to a Tag that is available in the Server by name. Note that Items are not the same thing as the Server Tags themselves. There will only be one Tag in the Server representing each global variable available in each configured controller but there could be many Client Items referencing each Server Tag. Just like Client Groups, the Items are objects created and maintained in and by the Server on behalf of the Client.

A Client identifies Items (data) to be added to Client Groups it creates by name. Recall that each Global Variable in each Device configured in the Server is assigned a Tag in the Server's configuration for that Device. Part of the Tag is a Tag name. To fully identify a Tag that the Client wants added to a Client Group, it must provide the full name of the Tag. The full name of the Tag includes the name assigned to the Device that represents the controller, any configured Subgroup names and finally the Tag name itself. A Subgroup name is a Server configuration Subgroup name, not the Client Group name. An example name to fully identify a Tag : Device101.Station2.:PartCount, where, Device101 is the name assigned to the Device in the Server configuration, Station2 is a Subgroup defined in the Server, and :PartCount is the Tag name representing the global variable in the controller.

2.3.3 Client Group Operations and Requests

Once the Client has established one or more Client Groups in the Server and added Items (representing global variables in actual controllers, i.e., Tags), it can request that the Server do certain operations with that Client Group. These operations can be the following:

- Scan the Tags for each Item in the Client Group data on its behalf. This is the typical data exchange mechanism. It is the most efficient and the preferred method for getting data from the controllers to the Clients. The Server reads all of the data associated with the Tags referenced by the Client Group's Items and then compares the new Tag data values with the last value sent to the Client. (The last value sent is stored in Item object that the Server is maintaining on behalf of the Client). If the Tag data is different than the Item data, the Item data is updated and the new value is sent to the Client as a type of Advise along with the new data's Time Stamp and Quality Information. Note that the sending of analog data on change is subject to deadbanding checks if so configured by the Client. The rate at which the Server Scans the Client Group is configured by the Client.
- Read a list of Items on its behalf. In a read operation, the Client request that a list of data Items, all already belonging to the Client Group, be read. Data is provided back to the Client along with Item Error Information, data Quality Information and Time Stamps. There are two types of read operations and two ways for the read data to be returned to the Client.

  The two types of reads are a Cache Read and a Device Read. In a Cache Read, the Server simply returns the Tag data value from the last time the Tags data was actually read. Note that no actual read is done to retrieve data from the controllers so the age of the data depends on when the last time the Client or some other Client actually had the Tag value read. In a Device Read, the Server actually goes out over the network to read the data from the controllers.

  The two ways data can be returned to the Client are Synchronous and Asynchronous. The Client specifies the type when it makes the request. A Synchronous read returns the Item data as soon as the data is available (either from the cache or from a read from the controller). In an Asynchronous read, the data is returned sometime later after the read operation is completed or new data is retrieved from the cache.
- Refresh the Items in the Client Group on its behalf. The Client request that a list of data Items, already belonging to the Client Group, be re-read from the actual controllers and that all Item data be sent back to the Client via the Advise mechanism. Data is provided to the Client (regardless of whether it has changed or not) along with error information, data Quality Information and time stamps.

- Write a list of Item data on its behalf. The Client requests that a list of data Items, already belonging to the Client Group, and associated data be written to the controllers. Similar to read operations, there are two ways the write operation can be performed, Asynchronous or Synchronous. In the case of writes, only the Item Error Information of the write operation are returned to the Client. There are not Time Stamps or Quality Information returned.

2.4 Overview of OPC Clients

The amount of control that you will have in the configuring of Client Groups, adding of Items to Client Groups and of setting of Client Group and Item properties will depend upon the actual OPC Clients that you are using. For example, some Clients may not allow you a choice of names for Client Groups added to the Server, to set scan rates and/or set Client Group active flags.

When Client actions such as setting a scan rate or setting the active state of an Item are referred to in this manual, the discussion is about what is possible from an OPC Client and not about what is possible from every OPC Client available.

This page is left blank intentionally.

3. DEVICE PROTOCOL SUPPORTED

The Server communicates with configured controllers via Ethernet over a UDP/IP socket (datagrams over connectionless sockets) to configurable service/port numbers. Controllers are identified for communication purposes by their Internet Protocol (IP) addresses. Each controller must be programmed with this address. See the PiC, MMC or MMC for PC Hardware manuals and the Function/Function Block Reference Guide for details on how to properly prepare and program your controller to support Ethernet communications.

A proprietary protocol used within the UDP datagrams to read and write controller variables. The protocol makes use of indices to uniquely identify parameters within a controller's ladder program. In addition to the error detection features inherent in Ethernet, IP and UDP, this protocol implements a checksum for each read or write command and a memory checksum. The memory checksum is to ensure that the Server has the same controller ladder Global Variable information that the controller is running. If the memory check sum does not match between the Server and the controller, the controller will return an error code in response to read and write requests. This indicates that the controller's program has changed since the Server imported the OID File and that either a different ladder needs to be downloaded to the controller or the Server needs reconfiguration including re-importing a different OID File.

The Server supports multiple outstanding messages to multiple controllers at the same time. Responses are matched with the transmitted commands via sequence numbers.

This page is left blank intentionally.

4. REQUIREMENTS

4.1 Hardware Requirements

4.1.1 Network Hardware Requirements

Since the Server communicates with the controllers over Ethernet, an Ethernet network must be in place. Depending on the cabling, distances, inter connectivity requirements, etc., this system may include bridges, routers, hubs, etc.. The network itself should be fully tested and be known to operate before attaching the controllers and the Server computers. Contact your system administrator for assistance or consult instructional documentation and manuals to setting up the network. It is beyond the scope of this Users Manual to discuss networking topics in any detail.

Once the network is in place and the Server computers and controllers are attached, check connectivity using available network testing tools and programs such as *ping*.

4.1.2 Computer Hardware Requirements

The following minimum computer hardware items are required for the computer that will be running the Server:

1. Pentium 75 MHz processor
2. 32 Megabytes RAM
3. 10 Megabytes hard disk space
4. SVGA display adapter (required for configuration only)
5. Ethernet adapter with proper interface type to attach to the Ethernet network.

While these are considered minimums, actual requirements will vary greatly depending upon the operating system, operating system options installed and the Server's configuration and the operation it will be requested to perform on behalf of Clients. Faster CPUs and more memory will greatly enhance the performance of the Server.

4.1.3 Controller Hardware Requirements

In addition to the standard system components, the PiC900 and MMC require Ethernet TCP/IP module be installed and configured while the MMC for PC requires the MMC for PC Support Software be installed and functional.

4.2 Software Requirements

The following software requirements must be met in order to configure and/or use the Server.

4.2.1 Computer Software Requirements

1. Windows NT 4.0 Service Pack 3.0, Service Pack 4.0 or newer recommended, OR Windows 95/98 with DCOM installed. You can download DCOM from the Microsoft site (http://www.microsoft.com/). Note that frequently DCOM will have been added to Windows 95/98 systems as part of the installation of other Windows programs such as Internet Explorer. Version 1.2 or newer is required.
2. OPC components:
   actxprxy.dll, comcat.dll - (NT or '9x version) These files provide some system interfaces used to register and find OPC servers using categories. These files are usually on NT, but may not be on clean installs. If these files are missing, then the server browsing function may not work, and servers may not be able to register themselves fully. These files are provided by Microsoft. Note actxprxy.dll should be at least dated 10/15/96 and version 4.70.1215. Also note that comcat.dll version 4.71 works and version 5.0 does not work.
   opcproxy.dll, opccomn_ps.dll - These are the standard OPC proxy DLLs from the OPC Foundation. If the proxy dlls are not installed, OPC will fail as soon as a connection is made. These files are provided by the OPC Foundation.
   opcenum.exe - This program is used by OPC Clients to browse for registered OPC servers on a computer. If this file is not installed, Clients may not be able to get a list of available servers. These files are provided by the OPC Foundation.
   OPCDaAuto.dll - This provides the OPC Data Access 3.0 Automation interface. Without it, Automation Clients

such as VB or VBA (Excel or Visio, for example) cannot access OPC servers. This file can be downloaded from FactorySoft.

The files listed above should be installed in the system directory as shared files and updated by version number. They should be registered in the order listed. DLLs are registered as follows:
"regsvr32 /s dllname"
The command "opcenum –regserver" will register the EXE.

3. Microsoft Foundation Class DLLs: **mfc42.dll, msvcrt.dll** and **mfc042.dll.**

4.2.2 Controller Software Requirements

1. PiCPro™ Version 10.1 or better for PiCs, 10.2 or better for MMCs, or 12.0 for MMC for PC.
2. Firmware that supports Ethernet TCP/IP communications in the Controller.
3. Ladder compiler with the following libraries: opinter.lib, io.lib, opc_enet.lib.
4. Ladder compiled with the following Function Blocks: OPC_10 or OPC_ENET.

4.3 Recommended Reference Material

The follow documentation and sources will provide additional information that will help in not only understanding the Server, but OPC in general.

1. OPC Overview documents including OPCOVW.DOC. These are available from the OPC Foundation Web site (www.opcfoundation.org)
2. Application note, "Networking with OPC using DCOM" from PCSOFT INTERNATIONAL. This application note provides information on configuring the computer systems to support DCOM, including information on security. It is available from the FactorySoft web site (www.factorysoft.com).
3. User Manual(s) for the OPC Client(s) that will be connecting to the Server.

The following manuals should be considered required reference material for getting communications operating between the controllers and the Server.

1. PiCPro™ Software Manual.
2. PiCPro™ Function Block Reference Guide.

5. REGISTERING AND DE-REGISTERING THE SERVER

The Server must be registered with the operating system on which it is to operate. The registration process places specific entries in the computer's Window Registry. This is so when a Client, either local or remote, asks the operating system for a list of potential servers or to start the Server by name, the operating system knows about the Server including its location and security settings. The Server is self registering. The Server is automatically registered as part of the installation process and unregistered as part of the uninstall process. If it is desirable to unregister the server without uninstalling and at a later time register the server, the following method can be used. Launching the server with the command-line argument /RegServer or -RegServer (case-insensitive) will register it. To un-register the Server, launch with the command-line argument /UnregServer or -UnregServer.

Example for registering: GL_OPCServer /RegServer
Example for un-registering: GL_OPCServer /UnregServer

Note: To install Verison 3.0 of the OPC Server, all previous versions of the Giddings & Lewis OPC Server must first be unregistered.

This page is left blank intentionally.

6. SERVER GRAPHICAL USER INTERFACE

In order for Clients to make use of a Server, the Server must be configured. Configuration includes setting up parameters defining the Communication Path(s), defining and configuring Devices to represent controllers, and configuring Tag data points that represent the Global Variables in the controllers. This section describes the graphical user configuration environment of the Server. See Section 7, "*Server Configuration Procedure*" for the sequence of steps involved in actually configuring the Server.

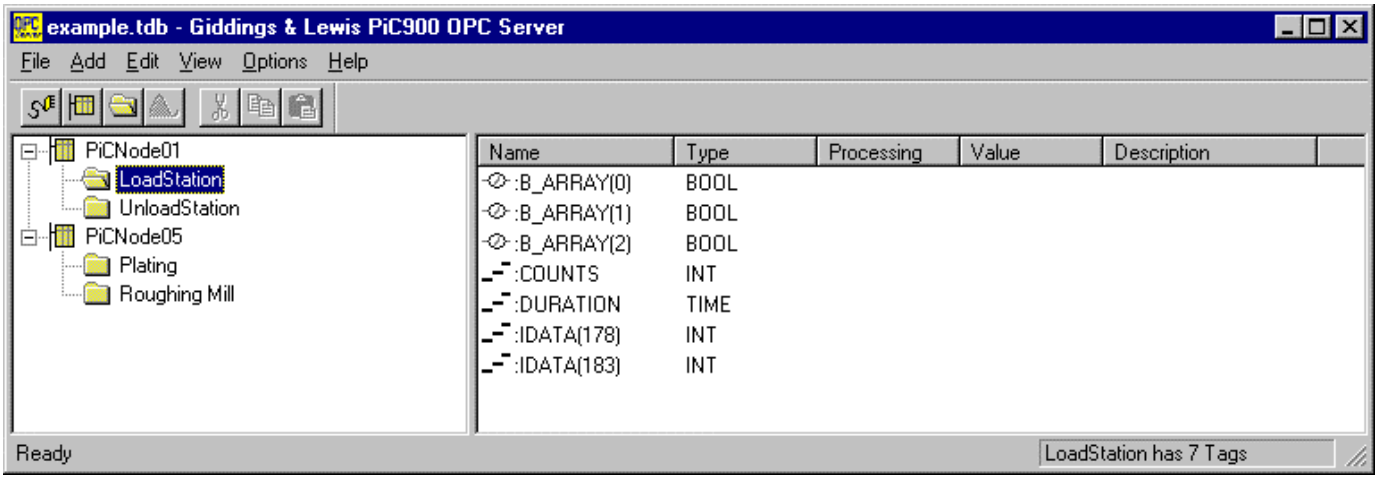
6.1 Overview

The Server has a graphical configuration environment similar to the Window's Explorer. There are two panes. The left pane is called the Device Pane. It contains a tree view of configured Devices and configured Subgroups. The right pane is the Tag Pane and it displays a listing of Tags that are in the current selection of the Device Pane, either a Device or a Subgroup. This is similar to the folder and file relationship and functionality of the Windows Explorer. The currently selected item in the Device Pane is referred to as the Selected Subgroup, regardless of whether it is a Device or an actual Subgroup.

A menu bar allows access to menu items to perform all necessary configuration operations and a tool bar has buttons to access frequently required operations.

A status bar at the bottom displays Server configuration status information.

Below is a view of the Server with a configuration defined for two Devices, each with two Subgroups. The Tags displayed in the Tag Pane are for the item that is highlighted in the Device Pane, Subgroup LoadStation for Device PiCNode01, or simply, PiCNode01.LoadStation.



6.1.1 Device Pane

The Device Pane contains a tree view of configured controller Devices and configured Subgroups. A Subgroup is a configuration grouping defined by the person configuring the Server. It has no relationship to the Client Groups created by the Clients using the Server. Both configured Devices and Subgroups can contain Tags.

Just like Windows Explorer, Subgroups can be defined within other Subgroups. Subgroups are depicted with folders icons, controller Devices with small controller icons. Both Devices and Subgroup icons can be "opened" and "closed". Opening a Device or Subgroup will show any Subgroups if any defined under it.

Only one Device or Subgroup may be highlighted (selected) at a time. As mentioned previously, the highlighted Device or Subgroup is referred to in this manual as the Selected Subgroup, even if the Selected Subgroup actually represents a Device. Right clicking on the Selected Subgroup will display its properties, either Device properties or Subgroup properties, for review and/or editing. In the example above, the Selected Subgroup is LoadStation under Device PiCNode01.

6.1.2 Tag Pane

The Tag Pane contains a detailed listing of the Tags in the Selected Subgroup. The Tags are displayed in columns as in an Explorer Detailed view type. The columns are:

Column Name	Description
Name	The Tag name is assigned when the Tags are generated for the Device when the Device's OID File is imported. The name is the same as the name used in PiCPro except that each Tag name is prefaced with a colon ":". A colon is required because PiCPro variable names can contain a dot "." Character and OPC Clients use dots as Subgroup specifiers when identifying Tags by name. Using a colon immediately before the Tag name allows unambiguous distinction between Tag names and Subgroup names. In front of the Tag name is a small icon representing the data type as BOOL, Analog, or Status.
Type	The data type as defined in the PiCPro program for the associated Global Variable.
Processing	Certain Tag types can be configured for scaling. If a Tag is configured for scaling, then this column entry will say **Custom**. If there is no scaling assigned to the Tag, the column entry will be blank.
Value	When the Server is in the Monitor Mode, this column entry will display the current value of the data read for the Tag. If the data can not be read for some reason, the column entry will say **Bad** to indicate that the quality is not good.
Description	An optional description for the Tag. When the Tags are generated during OID File import they initially have no descriptions. Descriptions can be added via the Tag properties dialog. Descriptions are for reference in the Server configuration only.

Clicking on the Name, Type or Description column headings will sort the respective column alphanumerically. Clicking a second time will resort the column in the reverse order.

Tags can be moved between Subgroups and Devices by first selecting one or more Tags and then dragging those selected using the left button to the new location. Note that Tags cannot be dragged to a different Device (or Subgroup in a different Device) and can not be moved if the Tag is currently in use. A Tag is in use if it is being monitored by the Server or if a Client has added a reference to it in one of its Client Groups.

Right clicking on a Tag displays a drop down menu of options that can be performed on the Tag. For this Server, only the Properties option is available. The Properties allows for review and/or editing of the selected Tag's properties. Note that the Device status Tags, :StatusCode and :StatusMsg, can not be move or edited.

6.1.3 Menu

The menu items allow access to all operations required to generate and edit the Server's configuration. Most menu items have sub menu items. The following menu items are provided:

Menu Item	Sub Item	Short Cut	Brief Description
File	New	Ctrl+N	Closes the currently open configuration and creates a new, blank Server configuration file.
File	Open	Ctrl+O	Brings up the Open file dialog box to allow the selection of a Tag Configuration File (files with extension tdb) . Selecting a file and clicking OK will close the current configuration and open the selected one.
File	Save	Ctrl+S	Saves the currently open Server configuration. If the configuration has not been saved before, the Save As file dialog box will be display to allow the entry of a file name and file path location.
File	Save As		Brings up the Save As file dialog box to allow the entry of a file name and file path location and upon clicking OK, the currently open Server configuration is saved to the file named.
File	1, 2, 3 and 4		1 through 4, if present, represent a list of the most recently edited Server configuration files. Clicking on one of the entries will close the current configuration and open the selected one.
File	Exit		Will stop the Server and Exit the Server application.
Add	New Comm Path	Ctrl+P	Display the Communication Path properties dialog to allow a new Communication Path to be defined.
Add	New Device	Ctrl+D	Display the Device properties dialog to allow a new Device to be defined.
Add	New Group	Ctrl+G	Display the Group properties dialog to allow a new Subgroup to be defined.
Add	New Tag	Ctrl+T	Not activated in this version of the Server
Edit	Cut/Copy/Paste		Not activated in this version of the Server
Edit	Delete	DEL	Deletes the currently selected item. Delete is not available if nothing is selected. If the item selected is either a Subgroup or a Device: Delete is not available if the Selected Subgroup is in use either by the Server while in Monitor Mode or by a Client, or if there are any Tags in the Selected Subgroup or in any Subgroup under the Selected Subgroup. If the item selected is a Tag : Not activated in this version of the Server.
Edit	Comm. Path		Pops up a Comm Path selection dialog box containing a list of the Communication Path names defined. Selecting a one from the list brings up the Communication Path properties dialog for that selection.
Edit	Properties	Alt+P	An appropriate properties dialog is opened depending on the item selected, either a Subgroup, a Device or a Tag properties dialog.
View	Monitor	Ctrl+M	Change whether the Server is in Monitor Mode or not.
View	Write	Ctrl+W	Write a value or values to a selected Tag or Tag(s).
View	Comm Errors**	Ctrl+E	Open and close a window to which communication error messages will be sent for viewing. These are debug type messages.

View	Status Bar	Alt+S	Toggles the display of the Status bar at the bottom of the Server application main window.
View	Comm Debug**		Open and close a window to which communication debug messages will be sent for viewing. These are debug type messages.
View	Server Debug**		Open and close a window to which OPC Server debug messages will be sent for viewing. These are debug type messages.
Options	Writes tab		Allows configurable, Server wide, write options to be set. See Section 6.7.1, "*Write* Options" for information on Server write options.
Options	Misc. tab		Allows configurable, Server wide, miscellaneous options to be set. See Section 6.7.2, "*Misc. Options*" for information on miscellaneous Server options.
Help	About		About brings up version and copyright information for the Server.

** The Comm Errors and Debug windows can display significant numbers of messages depending on the debug levels set if they are opened. These windows should not be opened and left open without supervision. The messages logged to these windows are not documented. They are intended for debugging only by support personnel. Having these windows open can significantly impact the performance of the Server and the computer, consuming both CPU time and memory.

6.1.4 Tool Bar

A set of tool bar buttons is provided to allow quick access to frequently used operations. The tool bar buttons perform the same functions as the equivalent menu item. The following table lists the tool bar buttons along with the menu equivalent. For a description, please refer to the corresponding menu item.

Button Symbol	Equivalent Menu Item	Brief Description - See the Equivalent Menu Item for complete description
	Add: New Comm.Path	Define a new communication path.
	Add: New Device	Define a new Device.
	Add: New Group	Define a new Subgroup.
	Add: New Tag	Define a new Tag
	Edit: Cut/Copy/Paste *	Cut/Copy and Paste Tags between Devices and/or Subgroup

*** Note:** The new Tag and Cut/Copy/Paste buttons are currently not activated in this version of the Server.

6.1.5 Status Bar

A status bar at the bottom of the Server window provides various status messages depending on the currently selected item and/or the current operation state of the Server.

6.2 Configuration Saving/Restoring

Server configurations can be saved and restored through File Menu operations. The files used to save Server configurations are called Tag Configuration Files. These files have the extension tdb.

The most recent file list on the file menu allows quick selection of recently edited files.

When the Server is started, either manually or by a Client invocation, the last file used by the Server is opened. This file serves as the Server's configuration until a new file is selected or created.

Note: The Server's configuration file cannot be changed while the Server has any Client connected to it or when it is in Monitor Mode.

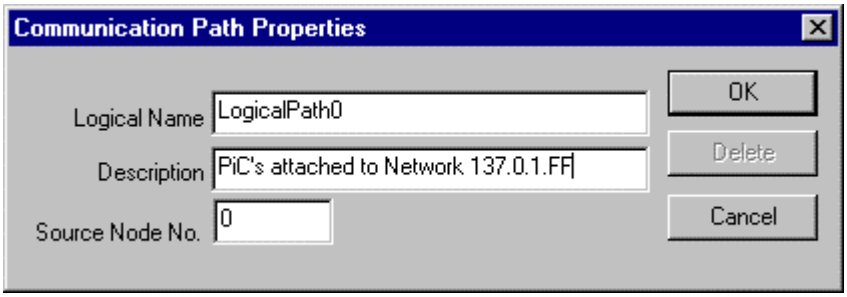
Note: If a configured Device is set to have Automatic OID File reloading, then the loaded configuration and possibly the Tag Configuration File itself may be modified with changed OID File import data. See Section 8.3, "Automatic OID File Import".

6.3 Defining and Editing Communication Paths

Defining Configuration Paths is the process of setting up one or more logical pathways that can be used to access controllers.

To define a new Communication Path, either click the **new Comm Path** button or on the Menu item **Add:New Comm Path**. To edit an existing Communication Path, click on the menu item Edit:Comm.Path and then select the Communication Path to be edited from the list presented in the selection dialog.

The figure below shows the Communication Path properties dialog that allows the properties to be defined and/or changed.



The following properties are part of configuring a Communication Path:

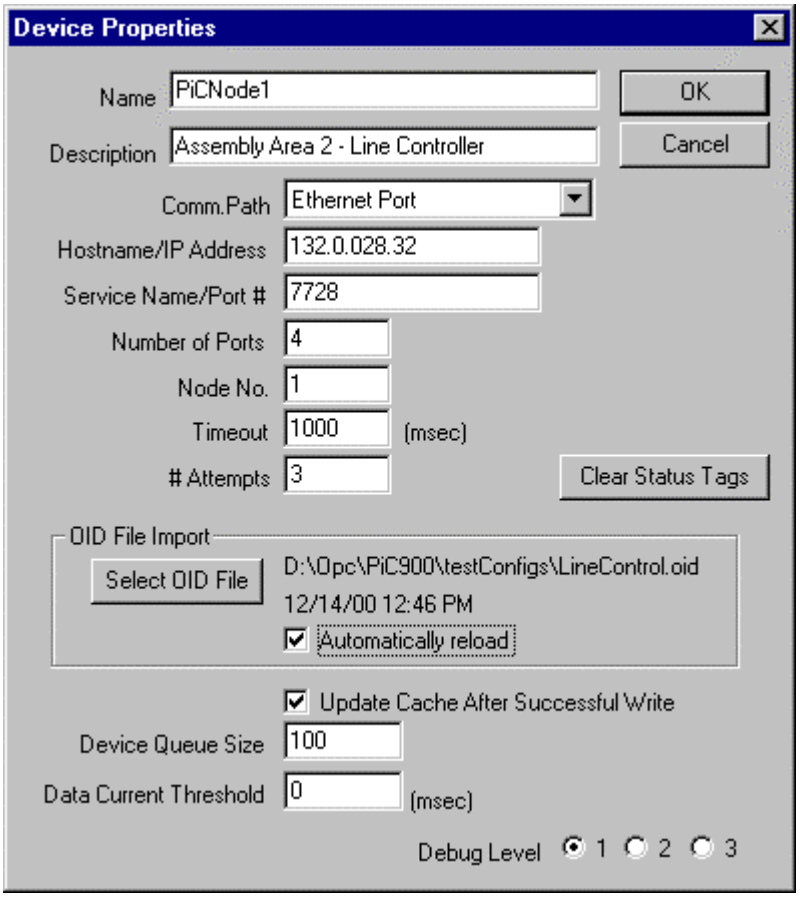
Item	Description
Logical Name	A character string you enter to represent the path being defined or edited. The name must be unique among all of the Communication Path names defined in the current configuration.
Description	An editable description for your reference in the Server configuration.
Source Node No.	The node number to be used as the source node number when communicating with all controllers defined on this Communication Path. All messages exchanged on the Ethernet network between controllers and the Server must have a source node number and a destination node number. The controller's actual node number is set via PiCPro. The Server is configured with the controller node numbers in the Device properties dialog. Valid Entries: 0 to 255, default 0

6.4 Defining and Editing Devices

The Server must be configured with Devices to represent the actual controllers with which it will be asked to communicate.

To define a new Devices, either click the **new Device** button or on the menu item **Add:New Device**. To edit an existing Device, select the Device and then either click on the menu item Edit:Properties or right click on the Device and click on Properties from the drop down menu.

The figure below shows the Device properties dialog that allows the properties to be defined and/or changed.



The following properties are part of configuring a Device:

Item	Description
Name	A character string you enter to identify the Device. The name selected must be unique to all other Device names assigned in the configuration.
Description	A description for your reference only.
Comm.Path	A Communication Path, previously defined, used to reach the Device.
Hostname/IP Address	The Hostname or IP Address of the Device. If a Hostname is entered, then the entered name must be a valid entry in the local host file or valid with a DNS. If you wish to use the IP address for the controller directly, enter the address in dotted decimal notation format. Note: On Windows 95/98, an entry must be made in the hosts file even if the dotted IP Address is entered.
Service Name/Port #	Either a UDP Service name listed in the Services file or the Port number of the UDP service offered by the controller for communications.

Number of Ports	The number of sequential, consecutive UDP port numbers to use when communicating with a single controller. When the number of ports specified is greater than 1, the Server will automatically send requests to numerically consecutive port numbers from the one entered or represented by, the “Service Name/Port #” entry on the Device Properties dialog. For example, if “4” is entered for the “Number of Ports” and a “Service Name/Port #” entry of “1234”, the Server will send the first command to port 1234, the second to 1235 and so on. Once the entire sequence of available ports has been used the Server will begin again at the first port. In the current example, the fifth command would use port 1234, the sixth 1235, and so on. If a command fails and needs to be retried, it is sent to the same port number on which it was originally sent. Valid Entries: 1 to 10, default 1
Node No.	The node number to be used as the destination node number when communicating with this controller. All controller messages exchanged on the Ethernet network must have a source node number and a destination node number. The source node used in the message is set in the Communication Path’s properties dialog. The actual controller node numbers are set in the controller via PiCPro. Valid entries: 0 to 255, default 0
Timeout	Time in milliseconds to wait for a response to a communication command. Valid Entries: 1 to 9999 msec., default 1000 (1 second)
# Attempts	Number of times the Server will issue a read or write request to try and get a successful data exchange completed before giving up and marking the requested as failed. Valid Entries: 1 to 5, default 3
OID File Import	This section shows and allows for the selection of the OID File used to define the Tags for this Device. To the right of the Select button is the name of the OID File currently imported for the Device and the date/time of the file. When the device dialog is first displayed, the information for last OID file imported for the device is display (or it is blank if no OID File has yet been imported). If a new OID File has been selected, then this information is for the newly selected file.
OID File Import:Select OID File	The Select OID File button allows the selection of an OID File to be imported. The file is not actually imported until the Device properties dialog is accepted and closed via the OK button. Importing an OID file will generate a Tag in the Server configuration for each global tag in the OID file. To the right of the Select button is the name of the OID File currently imported for the Device and the date/time of the file.

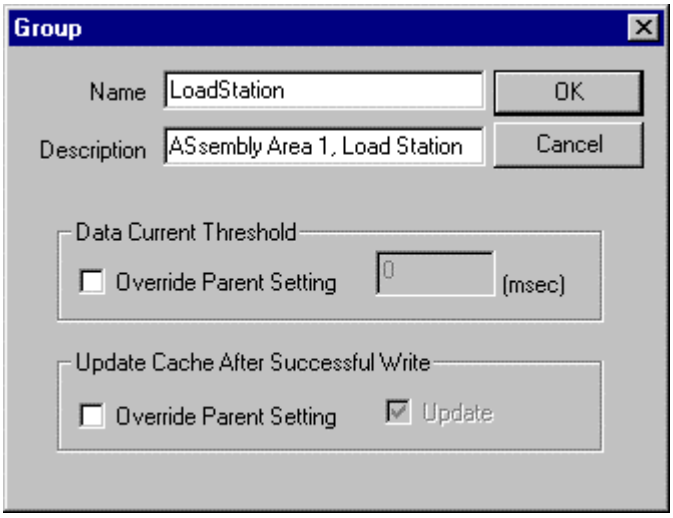
OID File Import: Automatically reload	The Automatically reload checkbox controls whether or not the selected OID File is automatically reloaded each time the Server's Tag Configuration file (tdb file) is loaded. If the box is checked then the configured OID File is loaded if it has a different timestamp from the currently loaded file, otherwise it is not loaded automatically. See Section 8.3, "Automatic OID File Import" for additional details on the automatic reloading of OID Files.
Update Cache After Successful Write	This sets the default Update Cache After Successful Write setting for all Subgroups and Tags belonging to this Device. See the Tag properties dialog for the meaning of this setting. Default is checked.
Device Queue Size	The number of read or write requests that can be queued waiting to be sent to this Device. If the Server is being requested to communicate with the Device faster than the Device or the Server can handle, than the number of requests on Device's request queue will start to increase. Once the queue fills, additional requests to read or write to the Device will generate errors back to the Client. Valid Entries: 10 to 32767, default 100
Data Current Threshold	This sets the default Data Current Threshold setting for all Subgroups and Tags belonging to this Device. See the Tag properties dialog for the meaning of this setting. Valid Entries: 0 to 60000 msec, default 0 (0 means always read)
Clear Status Tags	This button when clicked immediately clears the status tags for this Device. You do not need to press OK and Cancel does not cancel the operation.
Debug Level	Setting to select the amount of communication debug data is sent to the Comm Debug window. Level 1 shows only communication error messages. Levels 2 and 3 show the flow of communication requests. It is recommended that the Level remain at the default. Default is Level 1.

6.5 Defining and Editing Tags Subgroups

Defining Subgroups is optional. Subgroups are a way to organize the Tags created for a Device. To organize Tags, you create Subgroups and then move desired Tags from the Device to those created Subgroups. The way the Tags are organized is up to the you. Subgroups can defined within another Subgroup.

To define a new Subgroup, select the Device or Subgroup to which the new Subgroup is to be added and then either click the **new Group** button or on the menu item **Add:New Group**. To edit an existing Subgroup, select the Subgroup and then either click on the menu item Edit:Properties or right click on the Subgroup and click on Properties from the drop down menu.

The figure below shows the Group properties dialog that allows the properties to be defined and/or changed.



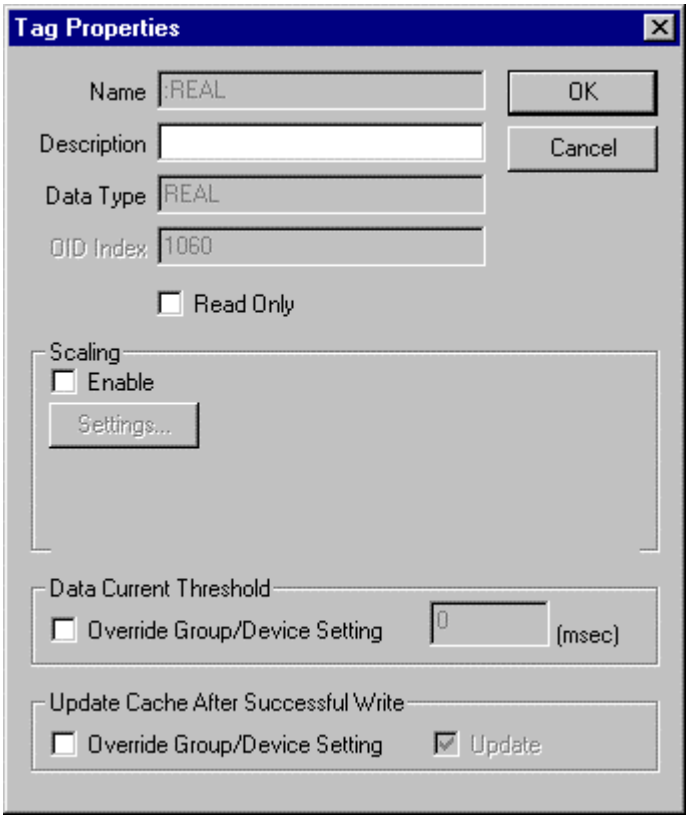
The following properties are part of configuring a Subgroup:

Item	Description
Name	A character string you enter to identify the Subgroup. The name must be unique with other Subgroups defined under the same Parent (Device or Subgroup).
Description	A description for your reference only.
Data Current Threshold	This sets the default Data Current Threshold setting for all Subgroups and Tags belonging to this Subgroup. The Subgroup defaults to the setting of the parent Device or Subgroup in which it belongs. Click on the Override button to enter a different value. See the Tag properties dialog for the meaning of this setting. Valid Entries: 0 to 60000 msec, default 0 (0 means always read)
Update Cache After Successful Write	This sets the default Update Cache After Successful Write setting for all Subgroups and Tags belonging to this Subgroup. The Subgroup defaults to the setting of the parent Device or Subgroup in which it belongs. Click on the Override button to enter a different value. See the Tag properties dialog for the meaning of this setting. Default is checked.

6.6 Defining and Editing Tags

Tags are created when an OID File is imported. The Tags are assigned non-editable Tag names when they are created. The Tag name is the same as the Global Variable name assigned in PiCPro except that it is prefaced with the ":" character. See Section 8, "*OID File Import and Tag Generation*" for information on the OID File import process.

The figure below shows the Tag properties dialog that allows the properties to be defined and/or changed. The dialog is accessed by selecting a Tag in the Tag pane and then either: clinking on the Edit:Properties, double clicking on the Tag, or right clicking on the Tag and then clicking Properties.



The following properties are part of the Tag properties:

Item	Description
Name	Tag name assigned to the Tag when the OID File was imported. It is the same as the global data point name as used PiCPro except that it is prefaced with the ":" character. It can not be edited.
Description	An editable description for your reference in the Server configuration.
Data Type	The PiCPro data type of the Global Variable imported. Can not be edited.
OID Index	OID File Index as imported from the OID file. Can not be edited.
Read Only	Tags are generated as both readable and writeable. Clicking this box will set the Tag as read only. Any attempt by a Client to write to the Tag will cause the Server to return Item Error Information indicating Bad Rights. Default: Unchecked
Scaling	This section allows scaling options to be set for the Tag. Clicking on Enable activates the Setting… button and sets up default scaling. If scaling is configured for the Tag, a message will also be displayed in this sections describing the scaling setting. See Section 7.4.1, "*Setting Tag Scaling*" for information on setting Scaling parameters. Default: Enable Unchecked
Data Current Threshold	The Tag's Data Current Threshold defaults to the setting of the Device or Subgroup in which it belongs. Click on the Override button to enter a different value. The Data Current Threshold is the time in milliseconds to consider the Tag's Server data cache value still up to date during data Scanning and Client

Refresh requests. A value of zero will always force data to be read. Example: If a tag was read last (and save in the cache) at 1:00:01.000PM and a Scan or read is requested at 1:00:05.00PM, the cached value would be used and returned if the Data Update Time Tolerance was 5000 (5 seconds) but if the Data Update Time Tolerance was 500 (0.5 seconds) the cache data would be considered "stale" and the Tag would be re-read.

Valid Entries: 0 to 60000 msec, default 0 (0 means always read)

Update Cache After Successful Write

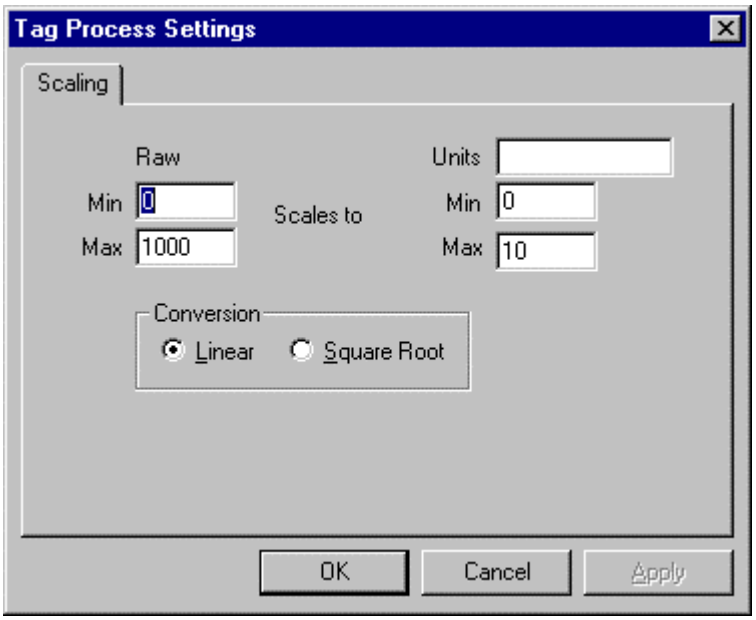
The Tag's Update Cache After Write setting defaults to the setting of the Device or Subgroup in which it belongs. Click on the Override button to change the setting for the Tag. When the Tag is written to successfully, the Tag's cache value will updated with the new value written if this box is checked. If the box is not check, then the Tag's cache value will not be updated until the next time the data point is actually read.

6.6.1 Tag Scaling

Scaling information can optionally be entered for each Tag via the Tag properties dialog. Data read from the Device for the Tag will first be scaled and then stored in the Server's cache and provided to Clients. Data received from Clients to be written is as well scaled before actually being written. Scaling is not allowed for Tags with the following controller Data Types: BOOL, STRING, DATE, TIME_OF_DAY, DATE_AND_TIME and TIME.

Note that scaling must be defined in order to support for Client deadband operations. See Section 9.3.3, "*Implement Deadband*" for information on deadband operations.

The figure below shows the Tag Process Setting dialog that allows the scaling to be defined. The dialog is accessed from the Tag properties dialog by checking the Enable Scaling checkbox and clicking on the Settings… button. The Scaling section on the Tag properties dialog is only available if the Tag type supports scaling.



The following properties are part of the Tag Process Scaling properties:

Item	Description
Raw Min and Max	The minimum and maximum values for the raw data.
Units	An editable description for your reference in the Server configuration. This description is appended to the data value when the Server is in Monitor Mode.
Units Min and Max	The minimum and maximum values for the data in the units entered.
Conversion	The type of data scaling to perform on the data, Linear or Square Root.

6.6.1.1 Linear

Linear scaling uses the following equation:

$$\text{Eng.Value} = \frac{(\text{RawValue} - \text{MinRawRange})}{(\text{MaxRawRange} - \text{MinRawRange})} * (\text{MaxEngRange} - \text{MinEngRange}) + \text{MinEngRange};$$

6.6.1.2 Square Root

Square Root scaling uses the following equation:

$$\text{Eng.Value} = \frac{(\text{Sqrt}(\text{RawValue}) - \text{MinRawRange}) * (\text{MaxEngRange} - \text{MinEngRange})}{\text{Sqrt}(\text{MaxRawRange} - \text{MinRawRange})} + \text{MinEngRange}$$

6.7 Options

Several configurable options can be set for the Server. To access these Server wide configuration settings, click the menu item Options. A tabbed dialog box will be displayed that organizes the various Server configuration options available.

6.7.1 Write Options

The write configuration options are accessed from the Write tab on the Options dialog. Currently the only option available is to set the Server to allow writes to the Device status tags. Normally each Device's Status Message Tag and Status Error Code Tag are read only and can only be cleared from the Device's properties dialog. By checking the option, Allow Write to Device Status Tags, the Server will mark all Device Status Tags with both read and write access permissions. This will allow individual Clients to write to these Tags. Note that if a Client had previously inquired about these Tag's properties, it will not know about the new access statuses until it queries again. Also note that Device Status Tags are not kept with individual values for each Client connected. If one Client clears or changes a Status Tag, all Clients will see the new value.

6.7.2 Misc. Options

Several miscellaneous Server configuration settings can be accessed from the Misc. tab on the Options dialog. The following table lists the configurable settings available:

Item	Description
Use Pre-built Read Commands	Normally the Server constructs each read command needed in order to perform an update scan as it is doing the scan. By checking the Use Pre-built Read Commands, the Server will build and re-build a set of the read commands necessary to retrieve the Tags configured for the Client's Group as the Client adds or removes Items from the Client Group. Depending on the protocol involved with a particular server, this can significantly help the performance of the server. With the Giddings & Lewis OPC Server, the effect is minimal. Experimentation is the only way to determine if using pre-built commands will help the Server's performance. Default: not checked, do not use pre-built commands.
Auto Save TBD on auto load of OID	If this option is set, then when a new Server Tag Configuration is loaded that results in the automatic reload of a Device's OID File, the Server will automatically save the new configuration (containing the new Tag information for the Devices that had automatic OID File reloads done). See Section 8.3, "Automatic OID File Import", and the Device Properties dialog.
OPC Server Debug Level	Sets the level of debug output messages sent to the Server Debug window when it is open. Level 1 shows Client Group level operations such as add an Item or create a Client Group. Level 2 shows additional details on how the Server processed those requests. Level 3 shows the details of every Item in the request and generates a very large amount of data to the window. It is highly recommended that the Level not be set to 3 unless it is know that the Client(s) are not making repeated requests and no Client Group is active. Default: Level 1

This page is left blank intentionally.

7. SERVER CONFIGURATION PROCEDURE

This section is intended to provide a step by step procedure for configuring the Server for operations. It is assumed that the Server had been installed and registered. It is also assumed that Section 6, "*Server Graphical User Interface*" on the Server's configuration environment has been reviewed. Please refer to that section for details on each of the configurable options available in the various dialogs presented during the configuration process.

7.1 Configuring Communication Paths

Communication Paths defined a pathway to configured Devices. In the case of this Server, the only real configurable option is the Server's Node number. This is the source node number used in the protocol messages to the Devices. Unless there is a good reason to assign different Source Node numbers, you should not need to define more than one Communication Path.

Click on the toolbar New Comm. Path button or on Add:New Comm.Path. The Communication Path properties dialog is displayed. Enter a Logical Name for the new path. This is the name by which the path you are defining will be known for future reference and editing. Enter an optional description if you wish. Change the Source Node number from the default of zero only if you have a good reason to do so. Click OK to accept the entries and create a new Communication Path definition.

7.2 Configuring Devices

You will need to define a Device in the Server configuration for each real controller that the Server is to communicate with. Click on the toolbar New Device button or on Add:New Device. The Device properties dialog is displayed with a default name assigned. Enter a name and description for the new Device. The Device name you enter will become part of the name for each Tag as referenced by Clients when they added Items to Client Groups they create.

Select one of the Communication Paths you previously defined from the drop down selection box and then enter the controller's IP and Service information along with its Node Number.

Click on the Select OID File button to select an OID File to import. Importing the OID file generates all of the Tags for the controller being defined. If the OID File is not available you can come back later to edit the Device and select the file at that time.

The other fields in the Device properties dialog are optional and normally do not need to be altered. Review each field's meaning and change as necessary.

Click the OK button. This will generate the new Device representing the desired controller. At least two predefined status Tags will be generated for the device to report communication errors. Additional Tags will be generated from the information in the OID File if it was specified. All of the new Tags will be listed in the Tag pane of the Server. Clients reference these Tags by DeviceName.TagName, where DeviceName is the name of the Device as you defined it and the TagNames are the Global Variable names as imported from the OID File (prefaced by the colon, ':' , symbol).

Define additional Devices following the same procedure if required.

7.3 Configuring Subgroups (Optional)

Initially all of the Tags imported for Device are located "under" the Device itself, i.e., by selecting the Device in the Device Pane, you see all of the Tags imported in one long list in the Tag Pane.

You can optionally define Subgroups in order to help organize the imported Tags. The way you organize the Tags is up to you. Subgroups can be defined directly "under" a Device or "under" other Subgroups.

Typically, Subgroups are set up to group Tags logically by function. Examples might be Subgroups for alarms or production data, Subgroups for machines, stations or lines, etc. By organizing the Tags into Subgroups, it allows users (Client) of the Server to locate Tags of interest easier rather than searching though one long list. This is particularly true if a Client supports the OPC browse capability.

An alternative to grouping Tags by function would be to group them by common performance requirement. Tag's each have their setting for Data Current Threshold and Update Cache After Successful Write default to the setting of the immediate Device or Subgroup to which they belong. If you had a set of Tags that all had very slowly changing values, you could place those under a common Subgroup set to have an appropriately set Data Current Threshold value. See Section 9, "*Optimizing Performance*" for information on optimizing communication throughputs.

To define a new Subgroup, select the Device by clicking on it and then click on the toolbar New Group button or on Add:New Group. The Group properties dialog is displayed with a default name assigned. Enter a name and description for the new Subgroup. The name you enter will become part of the name for each Tag as referenced by Clients when they add Items to Client Groups they create.

The other fields in the properties dialog are optional and normally do not need to be altered. Review each field's meaning and change as necessary.

Click the OK button. This will generate the new Subgroup and place a new folder icon under the Device. You can continue to define other Subgroups under this or other Devices and/or Subgroups by selecting the appropriate Device or Subgroup and repeating this procedure.

To move Tags into a new or different Subgroup, select the Device or Subgroup in the Device Pane that has the Tags you wish to place in the new Subgroup location. Select the Tags from the Tag Pane that are to be moved. Multiple Tags can be selected over a range by using the Shift key in conjunction with the left mouse click and/or multiple Tags can be selected by using the Control key in conjunction with the left mouse button. Once the chosen Tags are selected, press and hold down the left mouse button, drag the selected Tags to the new Subgroup and then release the mouse button.

Note that the new Subgroup and its Tags will remain as configured even if a new OID File is imported for the Device. See Section 8.2, "Re-importing OID File".

Note: Tags cannot be moved if a Client is referencing them or if the Server is in Monitor Mode. Tags cannot be moved from one Device to another Device. Device Status Tags cannot be moved from their initial location.

7.4 Editing Tags

Tags are initially created when a Device's OID file is imported. Each Tag created has certain properties assigned such as a name and a data type. While most of these properties are not editable, you can enter a description for each Tag, set the Tag to be read only, set the Tag to be scaled (see Section 7.4.1, "*Setting Tag Scaling*" below) and override Tag parent settings for Data Current Threshold and Update Cache After Successful Write.

Bring up the Tag properties dialog by double clicking on the Tag, right clicking on the Tag and choose Properties, or select the Tag and click on the menu item Edit:Properties. Enter your new settings for the Tag and then press OK.

Note that the Tags' settings will remain as configured even if a new OID File is imported for the Device. See Section 8.2, "Re-importing OID File".

7.4.1 Setting Tag Scaling

One of the optional settings configurable for Tags is the ability to setup scaling parameters. If the data type of the Tag supports Scaling, then the Enable Scaling checkbox will be enabled indicating that you can configure scaling. Check the checkbox. The Settings… button will become enabled and a message will be displayed with the default scaling parameters. Click on the Scaling button to bring up the Tag Process Setting dialog, Scaling tab. Enter the desired scaling parameters and click OK. The message in the Scaling section on the Tag properties dialog will reflect your settings. See Section 6.6.1, "*Tag Scaling*" for information on the scaling parameters and how scaling works.

7.5 Testing

To initially test your configuration, select one of the Subgroups or Devices you have defined and place the Server into Monitor Mode (see Section 7.5.1, "*Monitor Mode*" below). You should see the data for the Tags in the Tag Pane displayed. If the data is displayed as Bad, then there is some sort of problem. Click on the Device (if not already selected) and view the

information in the :StatusCode and :StatusMsg tags (while in Monitor Mode). These tags should contain error information. Look up the meanings of these errors, along with corrective actions, in Section 14, "*Device Status Error Codes and Messages*". See Section 17, "*Trouble Shooting*" (trouble shooting) for addition help if necessary.

Note that the Server does not support the capability to write data to the controller(s), only to read.

7.5.1 Monitor Mode

The Server can be put into a Monitor Mode for the purpose of checking communications. From the Menu, select View and Monitor, or press Control-M. In Monitor Mode the Tags in the currently selected Subgroup or Device will be read. The returned values are displayed in the Tag Pane. If there is an error reading a Tag(s), Bad is displayed in the value column for the Tag(s). The Tags are read continuously until the Server is taken out of the Monitor Mode. After concluding the reads of all Tag(s) in the currently selected Subgroup, the Server waits 300 milliseconds and then starts reading the values for the Tag(s) again.

The Server should not be left in Monitor Mode because it impacts the Server's performance and consumes CPU processor time. Most of the performance impact is from updating the Tag Pane value display.

7.5.2 Tag Write

The Server can be used to write a value to Tags for testing communications or for changing values in a device. Select one or more Tags in the Tag pane. From the Menu, select View and Write, or press Control-W. A pop up window will be displayed to allow entry of a value for each Tag to be entered. After pressing OK, the window closes and the value is written. If more than one Tag was selected, then the pop up window will be displayed for each after the previous value has been written.

Only Tags set to OPC writeable can be written to. If all selected Tags are not OPC writeable, then the tag write option is not available. If multiple Tags are selected, where some are writeable and some are not, then an error message will be displayed instead of the data input pop up for those Tags that are not OPC writeable. If the data entered cannot be converted to the data type of the selected Tag, then an error message is displayed and no attempt to write to that Tag is made.

If the entered Tag data is compatible with the Tag type and the Tag is OPC writeable, then the Server will try to write the value to the Device if the Tag is device writeable. The same rules for updating the Tag's cache value are followed as are for client writes. See the entry for "Update Cache After Successful Write" in Section **Error! Reference source not found.**, *"**Error! Reference source not found.**"*, for information on how a Tag's cache value is updated. If there is a problem performing the write operation, an error message will be displayed. Use the Comm Errors and Comm Debug windows to determine what the actual problem is.

If a value is entered for the pre-defined Device :StatusCode and :StatusMsg tag and/or a tag that is not device writeable, the Server will update the Tag's cache value only. No actual write occurs.

The data input pop up window contains three buttons. The OK button accepts the value entered, Cancel aborts the write operation to the current Tag (as indicated in the pop up) and Abort cancels the write operation to the current Tag in addition to any remaining Tags.

This page is left blank intentionally.

8. OID File Import and Tag Generation

For each Device defined, an OID File should be imported in order to define its Tags. Whenever a controller is downloaded with a new ladder program or a newly compiled ladder program with new Global Variables, the OID File must be re-imported.

8.1 Initial OID File Import

The first time an OID File is imported, all of the Tags generated are placed under the Device itself. If Subgroups are defined under the Device, the generated Tags can be manually moved from under the Device itself to the defined Subgroups.

The Tags as imported are all set to be Read/Write and have no description or scaling. By selecting the Tag properties, a description can be entered, the read/write setting changed and scaling set up.

8.2 Re-importing OID File

You will need to re-import an OID File for a Device if new Global Variables have been added to the PiCPro ladder program, compiled and downloaded for the Device. This is because whenever a new ladder program is compiled and downloaded to the controller that has new Global Variables in it, a new memory check sum is generated. This memory checksum is used by the Server and the controller to insure that the Server has been updated with the lasted Global Variables information. If an incorrect memory check sum is used by the Server when trying to communicate with a controller, the controller will return an error code and reject the request.

To re-import an OID File, it must be re-selected. From the Device's properties dialog, click on the OID Import button, select the OID file again and then clicking OK on the file selection dialog and OK on the Device dialog.

When a device has a new OID File imported, the import process attempts to maintain the configuration settings for the Device, specifically the Tags' descriptions and the Tags' locations in Subgroups. If a Tag in the old Device configuration that is not in the new OID File, then that old Tag is deleted from Device's configuration. Any new Tags in the OID that are not part of the current Device's configuration are added under the Device itself just as Tags are done when defined on the initial import.

If a Tag in the new OID File being imported has a different Data Type (and/or a different OID index), then the Tag is updated with the new configuration by the import process automatically. If a Tag is changed from one that had scaling enabled to one with a Data Type that does not support scaling, then scaling will be set to disabled.

8.3 Automatic OID File Import

The Server supports the capability of automatically reloading an imported OID File for a Device(s) whenever a saved Server Tag Configuration file is loaded. Each Device has this option configured through its Properties dialog via the "Automatically reload" checkbox in the OID File Import section.

If this box is checked, then when the Server loads in a Tag Configuration file (a tbd file), the Server will attempt to open and re-import the OID File specified for the Device if it is different from the currently loaded one. (The Server loads in a Tag Configuration file as a result of a user action initiated from the File menu option and/or when the Server first starts and loads the last Tag Configuration automatically). An OID File is considered different from a currently loaded one based solely on the files' date and time stamps. If the selected OID File is different then it is automatically re-imported following the same process and rules as described in Section 8.2, "Re-importing OID File".

The Server can be configured to automatically do a Tag Configuration save to store the new Tag information after any Device has had an OID File automatically reloaded. This is configured from the Options Menu under the Misc. tab by checking the "Auto Save TBD on auto reload of new OID" option. If this option is set and any Device has a new OID File imported, then the newly imported Tag information will be automatically saved in the currently opened Tag Configuration file (tbd file).

If any OID File is automatically imported during the load of a Tag Configuration file, the Server configuration is not considered changed. This means that if the "Auto Save TBD on auto reload of new OID" option is not set and the Server is

shutdown without doing a save, the newly imported Tag information will not be saved in the Tag Configuration file (tbd file). It also means that if no other changes have been made to the configuration, the Server will not prompt with the file-changed dialog on shutdown.

If a Device is marked for automatic OID File reload in a Tag Configuration file and the Server cannot find the specified OID File when the Tag Configuration file is being loaded, then the existing tags from the last save configuration will be used.

9. OPTIMIZING PERFORMANCE

Optimizing the Server performance may become necessary if Clients are requesting a large amount of data or if there are many Client making requests. There are several areas that you may wish to try to optimize. One is in the Server's throughput capability by trying to limit the amount of duplicate data the Server is requesting form Devices. If for instance, you have two Clients both with Client Groups containing the same Tag references and set for more or less the same Scan rates, the Server will be requesting those duplicate Tags twice. The other area for optimization is in the amount of data the Server returns to the Client(s) over COM/DCOM.

9.1 Data Current Threshold

Setting Data Current Thresholds help to keep the Server from asking for duplicate Tag data by defining a time limit within which the last data read for a Tag is considered as still "fresh". Recall that every controller Global Variable has a corresponding Tag in the Server configuration and with the Tag is a cache value that is updated on every successful read for the Tag, regardless of who or what caused the read. Having a Data Current Threshold helps eliminate duplicate Scan reads because when a Tag needs to be read, the Server first checks to see if the Tag's cache data is still fresh with respect to its Data Current Threshold and if so, skips the actual read.

A Data Current Threshold of zero (0) always forces the read, i.e., the cache data is always considered stale. Device Reads ignore the Data Current Threshold. By the OPC Specification, they always do an actual read operation to the Device.

Data Current Threshold settings works best when you are not using Pre-Built Commands. This is because a Pre-Built Command will be issued if even one of the Tags covered by the Pre-Built Command needs refreshing.

9.2 Pre Built Commands

Normally, as the default mode, the Server builds the read commands needed to perform update Scans on behalf of Clients each time they are needed. The Server also supports what are termed Pre-Built Commands. These are enabled from the Misc. tab of the Options menu item. See Section 9.2, "*Pre Built Commands*".

In the default mode of operation, read commands are built as they are needed. As the read command is being built each Item that is active in the Client's Group is evaluated to see if the Item is active and if the Tag cache value for the Item is "stale" with respect to the Tag's Data Current Threshold setting. If the Item is active and the data is "stale" the Server adds to the command under construction the information required in order to retrieve that Tag's data.

If the Server is configured to use Pre Built Commands, the read commands necessary to retrieve the Tags for the Items in the Client's Client Group are constructed as Tags are added and removed from the Client Group. When it is time for the Server to scan the Tags for the Items in the Client Group, the Server looks at each Tag to see if it needs to be read, based on Data Current Threshold setting. If any Tag in the command needs to be read, the Server issues the command, it does not have to build the command.

Determining which mode to use in order to get the best throughput performance out of the Server depends upon the Data Current Threshold settings for the various Tags in the various Client Groups and how frequently the Clients are adding Items to or removing Items from their Client Groups. For example, if Data Current Threshold settings are being used for some tags in the Client Group, and only one of the Tags needs to be updated, then the entire command will be issued with all the transfer time involved, where as if pre-built commands were not in use, a much smaller command would have been issued. On the other hand, if the majority of or all of the Tags needed to be read, then simply issuing the pre-built command would save the time of having to construct it.

Experimentation will most likely be the only way to determine the preferred method and in fact, with the protocol used between the controller(s) and this Server, there is likely to be little difference.

9.3 Client Configurations

How the Server performs is very much dependent upon how the Clients make use of it. The type of requests made also impacts DCOM network traffic in regard to the amount of data to be transferred to the Server in the way of request and the

amount of data transferred to the Clients in the way of response data. The ways Clients access data from the Server are very varied as is the amount of control you have in configuring those methods.

9.3.1 Preferred Operations

OPC in general assumes that Clients will be asking that the Server Scan Tags associated with Items in Client Groups and provide only data back when data has changed. Device Reads and Synchronous operation are to be generally considered "slow". You should, if possible, configure your Clients to only perform Cache Reads and/or to accept data on change from Client Group Scanning and to not do Device Reads. Client Group scan rates should be kept at the slowest possible rate and still met the Client application's requirements. Setting the rate any faster unduly burdens the Server and may take up communication bandwidth that could possibly be used by another Client. If there are Tags that a Client wants that can be grouped together on required update rate, than these should be place in separate Client Groups if possible, each with an appropriate update rate, rather than in one group set to the worst case rate.

9.3.2 Cache Reads versa Device Reads and Scanning

Both Cache Reads and Device Reads request the Client to pass arrays of information to the Server to specify which Items to read. Device reads perform actual reads from the Device ignoring the Data Current Threshold settings. All read operations return all data requested regardless of whether the data has changed or not. While Cache Reads are preferable to Device Reads since they eliminate the actual read operation, they are not as desirable as Client Group Scanning. Cache Reads do not ensure the data is up to date and hence may return stale data. Scanning on the other hand ensures that data is always kept up to date based on the Client Group's configured scan rate and only returns data that has changed. Further more, Client can specify deadband limits for their Client Groups thereby eliminating the frequent returning of "noisy" data.

9.3.3 Implement Deadband

The Server supports deadband processing on the data Scanned as defined in the OPC Specification. When a Client creates a Client Group, it can specify a deadband value. When the Server Scans values for Tags that are enabled for deadband process (described below), it will only return new Tag values to the Client when the newly read values have changed more than the deadband amount from the last value sent to the Client. The internal cache values for the Tags are updated independent of any deadband processing.

As defined by the OPC Specification, the deadband value is always specified as a percent of full scale. Therefore, In order to use deadband processing with a Tag you must specify what the Tag's full scale value is. This is done by setting up the scaling for the Tag. See Section 7.4.1, "*Setting Tag Scaling*" for configuring Tag scaling.

Deadband processing is not used for read operations, only for the Scan and Advise on change mechanism.

This page is left blank intentionally.

10. SUPPORTED CLIENT OPERATIONS

The Server supports the following Client operations:

- Server configuration space browsing.
- Synchronous and Asynchronous Read with both Device and Cache Read
- Synchronous and Asynchronous Write (Single and Block Writes)
- Group Definitions with Asynchronous Data Scan and On Change Update
- Data Refresh

This page is left blank intentionally.

11. DIALOG MESSAGES

The following messages may be displayed in pop up dialog boxes either during startup, configuration or shutdown. The messages are listed in alphabetically order along with explanations.

Dialog Message	Explanation (Internal ID)
A communication path already exists for the entered logical name.	Each Communication Path defined must have a unique name different from all other Communication Path names. Enter a different name. (IDS_APP_ERR_PATH_EXISTS)
Cannot change files while OPC Clients are connected to Server.	You cannot close and open a new Server configuration tag database file while Clients are connected to the Server. (IDS_APP_ERR_FILE_CHANGE_WITH_CONNECTIONS)
Cannot create device object! Check Hostname/IP Address and/or Service name/Port #.	The Server cannot create an internal object to represent the Device you are trying to create because the operating system cannot resolve the Device's IP Address and Service information. 1) Make sure the entries are valid. 2) If entering a hostname rather than a dot IP address, make sure that the name and address is listed in the operating system's host files. 3) If using dotted IP address under Windows 95, an entry still needs to be made in the hosts file. (IDS_APP_ERR_TCPUDPCOMM_CREATE_DEVICE)
Cannot create SEMAPHOR object!!!	The Server could not create an internal object to represent a Device because the operating system would not allow the creation of a semaphore object. Generally, this is due to a system resource problem. 1) Shutdown unnecessary applications. 2) Add memory or disk swap space. (IDS_APP_ERR_TCPUDPCOMM_CREATE_SEMP)
Cannot create socket, OS error EEEEE!!!!	The Server could not create an internal object to represent a Device because the operating system would not allow the creation of a communication socket. EEEEE is the error code from the operating system. 1) Shutdown unnecessary applications. 2) Look the error code EEEEE up if you have access to Microsoft function error codes. (IDS_APP_ERR_TCPUDPCOMM_CREATE_SOCKET)

Cannot create TCPUDPComm object!!!	The Server could not create an internal object to represent a Device because the operating system would not allow the creation of an object to represent TCP/UDP communications. This is probably due to a system resource problem or TCP not being installed on the computer. 1) Shutdown unnecessary applications. 2) Add memory or disk swap space. 3) Make sure TCP/IP has been properly installed and configured. (IDS_APP_ERR_TCPUDPCOMM_CREATE)
Cannot re-create device "*name*" for Host: "*hostname*", Service: "*servicename*"	The Server cannot re-create an internal object to represent the Device called *name* because the operating system cannot resolve the Device's IP Address and Service information. It may be possible that the Server configuration you are loading was created on a computer for which the information was valid. 1) Change the Device's Hostname/IP Address and Service Name/Port # entries in the Device properties dialogs if no longer valid. 2) If hostname is a name and not a dotted IP address, make sure that the name and address is listed in the operating system's host files. 3) If using dotted IP address under Windows 95, an entry still needs to be made in the hosts file. (IDS_APP_ERR_TCPUDPCOMM_RECREATE_DEVICE)
Creating OPC DA 1.0 Component Category failed. Creating OPC DA 2.0 Component Category failed. Creating OPC DA 3.0 Component Category failed.	These errors all have to do with installing, creating and registering the Server with the operating system. 1) Make sure that OPC Components have been installed. (IDS_APP_ERR_CREATE_OPC10_CAT), (IDS_APP_ERR_CREATE_OPC20_CAT), (IDS_APP_ERR_REG_OPC10_CAT), (IDS_APP_ERR_REG_OPC20_CAT)
Duplicate name.	The name you are assigning to the object you are creating, Device or Subgroup must be unique. This means that each Device must have a name different from all other Devices and that each Subgroup must have a name different from all other Subgroups under its Parent. (IDS_APP_ERR_DUPLICATE_NAME)
name has devices assigned to it. Can not delete.	You cannot delete a Communication Path that is being referenced by Devices. Change the Devices using this Communication Path to use a different Communication Path or delete those Devices before deleting the Communication Path itself. (IDS_APP_ERR_DELETE_PORT)

name is referenced (in use) or contains tags. Cannot delete.	If *name* is a Subgroup, you cannot delete a Subgroup that has Tags in it or has other Subgroups that have Tags in them. Move all Tags in the Subgroup and all Subgroups within the Subgroup to either the Device or to Subgroups not under *name*. Then you can delete it. If *name* is a Device, you cannot delete a Device that has Tags in it that are referenced by Items in a Client Group, i.e., that is being used by one or more Clients. Stop those Clients from referencing Tags in the Device. Then you can delete it. (IDS_APP_ERR_DELETE_GRP)
Ole Initialization failed.	Object Linking and Embedding initialization failed. Failure may be due to incorrect versions of the OLE system DLLs. 1) Make sure that the last OLE DLLs are installed by (re) installing the latest operating system service pack. (IDS_APP_ERR_OLE_INIT)
Registering server in OPC DA 1.0 category failed. Registering server in OPC DA 2.0 category failed. Registering server in OPC DA 3.0 category failed.	These errors all have to do with installing, creating and registering the Server with the operating system. 1) Make sure that OPC Components have been installed. (IDS_APP_ERR_CREATE_OPC10_CAT), (IDS_APP_ERR_CREATE_OPC20_CAT), (IDS_APP_ERR_REG_OPC10_CAT), (IDS_APP_ERR_REG_OPC20_CAT)

This page is left blank intentionally.

12. STATUS AND ERROR REPORTING INFORMATION

The Server provides several different methods for reporting errors and for helping to locate problems. This section describes where errors can be reported.

12.1 Tag Quality

As defined in the OPC Specification, data read and returned to Clients is always provided with Quality Information to indicate the quality of the data being returned. Depending on the Clients using the Server, they display or use this Quality Information in different ways. See Section 15, "*Server OPC Item and Tag Quality Values*" for information on data Quality Information and your Client documentation for information on how the returned Quality Information is used. Note that Quality Information only applies to read and Scanned data and does not give any indication as to the success or failure of write operations.

12.2 Client Read/Refresh/Write Item Error Information

As defined in the OPC Data Specification, the results of Client initiate read, refresh and write operations are returned the Client for each Client Item in the request. This is in addition to the success or failure Boolean return status for the request (and to the data Quality Information provided per Tag for read and refresh requests).

If during a Client requested read, refresh or write operation, a failure or error occurs, the entire request will be marked as failed. This means that some of the data in a read or refresh may not be valid and that some of the data in a write may not have been written. With completion of each request, the Client receives an array of Item Error Information, one error code for each Item in the request. Any Item that did not succeed will have a non zero error code returned.

The error codes that the Server can return are listed in Section 16, "*Server OPC Error Codes*". The Client can make a call to the Server to enunciate a message for each of these Client codes. Section 14.2, "*StatusMsg Error Messages*" also lists the messages that will be enunciated for each of the corresponding codes.

Consult your Client documentation for information on how the returned Item Error Information is used.

12.3 :StatusMsg and :StatusCode Tags

In additional to the information provided in the OPC Data Specification, each Device when defined has two Tags automatically created and configured. One Tag is a string type called :StatusMsg and the other Tag is an integer type called :StatusCode. The two Tags are always placed under the Device to which they belong. They cannot be edited or moved to other Subgroups or Devices. When any communication errors occur for the Device, these Tags will be updated with error information. :StatusMsg will contain a message to indicate the problem and the :StatusCode will contain an integer value code to indicate the problem.

See Section 14.2, "*StatusMsg Error Messages*" for a list of possible error messages that can appear in the :StatusMsg Tag. See Section 14.1, "*StatusCode Error Codes*" for a list of possible error codes that can appear in the :StatusCode Tag.

12.3.1 Server Access

To see the status values, select the Device and place the Server into Monitor Mode.

The Tags do not automatically clear. To clear them, go to the Device properties dialog and click on the Clear Status Tags button. :StatusMsg will be cleared and :StatusCode will be set to zero (0).

12.3.2 Client Access

The :StatusMsg and :StatusCode Tags can be read by Clients just as any other Tag and can be sent to Clients on change via a scan group on change just like any other Tag. The Tags are read only by default. The default can be changed from the Options menu to allow Clients the ability to write to them.

12.4 Error Logging

The Server supports three logging windows that, when opened, will log messages generated by the Server. It is recommend that you do not leave these windows open in a deployed system but only open and use them when trying to debug a problem or set up a new configuration. Leaving them open can impact the performance of the Server, particularly when some of the higher debug levels are turned on or there are many errors occurring.

The messages logged to these widows are not documented.

Depending upon the debug levels set and the number of Clients using the Server at the time, the messages being logged to these windows may seem quite confusing. To best make use of them, particularly when the higher level debug levels are set, try to limit the Clients accessing the Server to the one that is having the problem. Additional, try to limit the operations to only those that are having the problem. For example, if a certain Clients is not getting values returned from a Cache Read, stop all other Clients, stop the Clients with the problem from doing any write operations and disable all Client Group Scanning, then open the debug windows and trigger the read operation.

Each of the windows opens by clicking on the corresponding sub menu item under View. Clicking the sub item a second time closes the corresponding window. The windows' contents are cleared when closed and re-opened.

12.4.1 Comm Errors

This window is used to log communication errors. The messages are generally self-explanatory and relate the error condition to the Device name assigned in the Device properties dialog. There are no configurable debug levels associated with these messages.

12.4.2 Comm Debug

This window is used to log low level communication errors. The messages detail the sending and receiving of individual messages. The level of detail is controlled by the Debug Level set in each Device's properties dialog. See Section 6.4, "*Defining and Editing Devices*" for the meanings of each of the different possible debug levels. These messages are generally intended to assist technical support personnel.

12.4.3 Server Debug

This window is used to log high level communication requests and Client Group management actions. The messages relate error conditions and Client requests to Client Group names. Depending upon the level set, the messages detail Client requests to add Client Groups, add Items to Client Groups, do reads, writes, etc., as well as to the actual Items read or written and the Item Error Information codes for each. The level of detail is controlled by the Debug Level set on the Misc. tab of the menu item Options. See Section 6.7.2, "Misc. Options" for the meanings of each of the different possible debug levels.

13. DATA STORAGE/CONVERSIONS BETWEEN DEVICE AND VARIANT TYPES

Data types native in the controller are stored in the Server's cache to be read by Clients and data from Clients must be written to the controller via the Server. Clients and Servers exchange data of known data types only. This data types are a subset of Microsoft's VARAINT data types.

Below is a table listing the Data Types supported by PiCPro and the corresponding variant data types that the Server uses for them.

Controller Data Type	Size in bytes	Controller Data Type Description	Variant Data Type	Variant Data Type Description
BOOL	1	Boolean	VT_BOOL	Boolean
BYTE	1	Byte	VT_UI1	Unsigned Byte
WORD	2	Word	VT_I2	2-byte integer value
DWORD	4	Double Word	VT_I4	4-byte integer value
LWORD	8	Long Word	NOT SUPPORTED	No variant type can accommodate
SINT	1	Signed Short Integer	VT_UI1	Unsigned Byte
INT	2	Signed Integer	VT_I2	2-byte integer value
DINT	4	Signed Double Integer	VT_I4	4-byte integer value
LINT	8	Signed Long Integer	VT_R8	8-byte IEEE real value
USINT	1	Unsigned Short Integer	VT_UI1	Unsigned Byte
UINT	2	Unsigned Integer	VT_I2	2-byte integer value
UDINT	4	Unsigned Double Integer	VT_I4	4-byte integer value
ULINT	8	Unsigned Long Integer	VT_R8	8-byte IEEE real value
REAL	4	Floating Point (6-7 significant digits)	VT_R8	8-byte IEEE real value
LREAL	8	Floating Point (15-16 significant digits)	VT_R8	8-byte IEEE real value
STRING	1-64	Character String	VT_BSTR	Character String
DATE	2	Date	VT_DATE	Double-precision value denoting a date & time
TIME_OF_DAY	4	Time of Day	VT_DATE	Double-precision value denoting a date & time
DATE_AND_TIME	4	Date and Time	VT_DATE	Double-precision value denoting a date & time
TIME	4	Time Duration	VT_R8 (msecs)	8-byte IEEE real value

Data write request from Clients come to the Server with data in a VARIANT types for tags that may or may not be the same as the above table requires. The Server first attempts to convert this data to the VARIANT data types required for the tag as indicated in the above table. The conversion is done via the Microsoft VariantChange routine which will convert the data consistent with OLE rules. For each data that can not be converted to the target type the corresponding item in the write request will be marked in error and will not stop the remaining items in the request from being processed.

Note that there are no VARIANT data types available in OPC to explicitly support the data transfer of unsigned two byte, unsigned four byte and signed one byte quantities. The Server uses the VARIANT type that has the value range to hold the value range of the controller data type. It is up to Clients to coerce the data to the proper signed value.

This page is left blank intentionally.

14. DEVICE STATUS ERROR CODES AND MESSAGES

This section gives meanings and corrective actions for error codes and error messages that can be stored in the :StatusCode and :StatusMsg Tags respectively. See Section 12.3, "*:StatusMsg and :StatusCode Tags*" for information on these Tags themselves, how they are created, updated, cleared, etc.

14.1 StatusCode Error Codes

Error Code (decimal)	Meaning and Corrective Action
0	No error
1 - 16383	Error code returned from the controller. Look in the PiCPro Function Block Reference Guide to determine the meaning of the code and to determine what action if any should be taken. Look at the documentation for the Function Block OPC_ENET. Example: 1 means that the memory check sum between the OID File imported for the device and the current ladder program in the controller are not the same. This means that a new OID File exists and should be imported.
16490	No response was received from the device for an issued command. 1) Make sure that the IP or hostname information entered in the Device properties dialog is correct. Test by using the utility ping. Ping by the entry type used in the Device properties dialog, either hostname or IP. 2) Increase the time out value set for the Device in the Device properties dialog. 3) Make sure that the Source Node No. and Device Node No. are set correctly in the Communication Path and Device properties dialogs respectively.
32769	The Server encountered a problem that was not planned for. Please contact Technical Support.
32770	The Server could not queue a read/write command because the queue for the Device was full. Generally this means that Clients are requesting that the Server communicate with the Device faster than commands can actually be transmitted. 1) Slow down the request by reconfiguring Clients to set slower update rates and to read/write less frequently. 2) The queue size of each device is configurable from the Device properties dialog. Increase the queue size to help handle short periods of increased communications.
32771	The Server could not re-create some of the internal objects needed to communicate with the Device. It is possible and likely that a Device was defined with IP information that is no longer valid. 1) Check Hostname/IP Address and/or Service name/Port # is correct in the Device's properties dialog.
16384 - 16489, 16491 - 32768, 32772 - 36863	An unanticipated error condition was encountered in the Server. Please contact Technical Support.

14.2 StatusMsg Error Messages

The table below gives the possible messages that can be stored into the :StatusMsg Tag. These same strings can be returned to Clients using the Server's GetErrorString functionality on item error return codes. Each message is listed alphabetically along with the corresponding :StatusCode value. Look up this :StatusCode value for an explanation and corrective action.

String	Corresponding :StatusCode	Internal Code
Device returned error/status code D.	D	IDS_OPC_EC_DEVICE_ERR_CODE
Low level Server error.	32769	IDS_OPC_EC_SERVER_LOW_LEVEL
No response.	16490	IDS_OPC_EC_COMM_NO_RSP
The Server's transmit pending queue for device is full.	32770	IDS_OPC_EC_SERVER_QUEUE_FULL

Can't create device object! Check Hostname/IP Address and/or Service name/Port #.	32771	OPC_EC_SERVER_TCPUDPCOMM_CREATE_DEVICE
Unknown communication error code D.	16384 + D	IDS_OPC_EC_COMM_UNKNOWN
Unknown Server error code D.	32768 + D	IDS_OPC_EC_SERVER_UNKNOWN
Unknown Server error type/code EEEEEEEE.	No corresponding :StatusCode. The Server was asked to resolve the error code EEEEEEEE but this error code is not known to the Server.	IDS_OPC_EC_UNKNOWN

15. SERVER OPC ITEM AND TAG QUALITY VALUES

All data returned from the Server is marked with Quality Information. The various quality values that can be returned are defined in the OPC Specification. The specification defines three sets of quality: OPC_QUALITY_GOOD, OPC_QUALITY_UNCERTAIN and OPC_QUALITY_BAD. Each of these can be logically ORed with further information also defined by the specification.

In the Server, Tags are initially created with a quality of OPC_QUALITY_BAD. The following table shows the various quality classes and values that can be returned and possible conditions that might cause the quality to be set by the Server. See the OPC Specification for the numeric values of the OPC quality constants.

OPC Quality Class	**OPC Quality Code**	**Possible Reasons**
OPC_QUALITY_GOOD	none	Successful read or after a successful write with Update Cache After Successful Write option set.
OPC_QUALITY_BAD	OPC_QUALITY_LAST_KNOWN	A communication error has occurred but prior attempt had succeeded and that value is being returned as the last known good value.
OPC_QUALITY_BAD	OPC_QUALITY_COMM_FAILURE	The communications failed and there is no prior last value know to be accurate.
OPC_QUALITY_BAD	OPC_QUALITY_CONFIG_ERROR	There is some problem with the Server either due to a configuration problem such as a bad IP address or with a resource problem like unable to allocate memory.
OPC_QUALITY_BAD	OPC_QUALITY_OUT_OF_SERVICE	Client has asked to read / write a Item that is either inactive or the Client Group is inactive.

This page is left blank intentionally.

16. SERVER OPC ERROR CODES

The OPC Error codes listed in this section are Server specific codes that can be returned to Clients as result codes for individual Items in a read or write request. These are not all of the codes that can be returned, only those unique to this Server. Any error code documented in the OPC specification might be returned. For errors not listed in this section, refer to the specification. The Server specific codes listed are based on the OPC specification for error codes. All error codes are 32 bits, where bits 31 and 30 indicate the severity code, bit 29 is a customer code flag, bits 27-16 are the facility code and bits 0-15 are the code itself.

Note that the Server supports the GetErrorString functionality. For any error that the Server returns to a Clients, the Clients can have the Server return a descriptive string for the code. The descriptive strings for error codes that are specific for this Server are the same as those placed in the :StatusMsg Tag. See Section 14.2, "*StatusMsg Error Messages*" for these strings.

All the Server specific codes documented have the customer flag set and a facility code of four (4) indicating that the error codes are context i.e., OPC, specific. The table below shows all of the possible Server specific error codes group according to types. Each error code is given in both hexadecimal and decimal, along with the error's explanation.

Error Type: OPC_EC_SERVER : *Code Format: 0xE0048XXX*	*These types of errors are for Server errors unrelated to actual communication operations.*
Error Code Hexadecimal (decimal)	**Explanation (internal definition)**
0xE0048001 (3,758,391,297)	The Server encountered a problem that was not planned for. Please contact Technical Support. (OPC_EC_SERVER_LOW_LEVEL)
0xE0048002 (3,758,391,298)	The item could not be read or written because the Server could not queue the read/write command because the queue for the Device is full. Generally this means that Clients are requesting that the Server communicate with the Device faster than commands can actually be transmitted. The queue size of each device is configurable from the Device properties dialog. Increase the queue size to help handle short periods of increased communications. Slow down the request by reconfiguring Clients to set slower update rates and to read/write less frequently. (OPC_EC_SERVER_QUEUE_FULL)
Error Type: OPC_EC_COMM : *Code Format: 0xE0044XXX*	*These types of errors are related to problems communicating with devices.*
Error Code Hexadecimal (decimal)	**Explanation (internal definition)**
0xE004406A (3,758,375,018)	No response was received for the command issued to read or write the item. See the error code 16490 in Section 14.1, "*StatusCode Error Codes*". (OPC_EC_COMM_NO_RSP)
Error Type: OPC_EC_DEVICE : *Code Format: 0xE004XXXX , where XXXX = 0000 to 3FFF.*	*These types of errors are codes returned from devices during communications. XXXX is the code returned form the device. Look this code up in the device's documentation.*
Error Code Hexadecimal (decimal)	**Explanation (internal definition)**
Example: 0xE0040001 (3,758,358,529)	XXXX = 1. This means that the device returned an error code of 1. Look up this value, 1, in the device's documentation.

This page is left blank intentionally.

17. TROUBLE SHOOTING

This section gives problem solving assistance for various tasks and operations associated with the Server.

17.1 OPC/DCOM Server Operations

Operations such as installing, registering, un- registering

Problem: During Server registration, the Server displays messages such as “Can not …. DA 1.0” or “…DA 2.0..” etc..
Solution: Make sure that 2.0 OPC components are installed. See Section 4.2, “*Software Requirements*” for a list of 2.0 Components. If the Server is to be used locally, then these components may not be needed, particularly by many OPC 1.0 Clients. If they are required by the Client, or the Server is to be used remotely then these must be installed.

17.2 OPC/DCOM Client Operations

Operations such as connecting, launching data exchange in regards to DCOM settings.

Problem: A Client using version 2.0 Client can not find the Server.
Solution: Ensure that 2.0 OPC components are installed. See Section 4.2, “*Software Requirements*”. Make sure that the Server has been registered. See Section 5, “*Registering and De-registering the Server*”.

Problem: Sometimes Clients will try to open their “own” copy of the Server rather than share the same one.
Solutions: 1) If one of the Clients is running in a different context then the other (for example, one is running as a service while the other as a user process) then it is possible that a separate Server is being started for each context. Run the DCOM configuration utility dcomcnfg.exe. From the applications tab, double click on the name of the Server to bring up its DCOM properties. Select the Identity tag. Change the radio button selection from the default of “The launching user” to “The interactive user”.

Problem: Unable to make an OPC connection between a remote Client and the Server or connection works but no data returned.
Solution: 1) Not all of the OPC components are installed on both computers. The components must be installed on both the client’s computer as well as the Server’s computer. See Section 4.2, “*Software Requirements*”. 2) Windows NT Service Packs are mismatched or are not at least version 3.0. Install the latest Service Packs on all Client and Server computers. 3) Security is improperly configured, possibly including unidentified users. Configuring proper security is a very complex issue. The simple solutions is to have the same user log onto both Clients and Server computers so that all computers are using the same security access and rights or to disable DCOM altogether. Both these methods should be considered temporary because they may cause network security problems. Download and review the Application Note “Networking with OPC using DCOM” from FactorySoft’s web site for a detailed discussion of the issues involved.

17.3 Starting the Server

File not found, can not load, missing DLLs,

Problem: When the Server is started manually or via a Client connecting, the configuration is blank.
Solutions: 1) The Server cannot find the last file that was opened the last time the Server was exited. It will open a blank configuration file. Find out what happened to the configuration file. If it has been moved, renamed, or is in a path that is not accessible, correct the problem. Manually start the Server and reload the file, then exit. 2) The Server cannot load the configuration file because it is not compatible. The Server can load files created by earlier versions but not newer versions. Make sure the configuration file was created by the same or earlier version of the Server.

Problem: The Server will not start because of missing DLLs.
Solution: The Server makes use of standard Microsoft DLLs that should be part of all Windows 95/98 and NT systems. If you do not have these files, go to Microsoft's web site to download copies and place them in the operating system's system directory.

17.4 Stopping the Server

Server continues to communicate after Client stops, will not terminate,

Problem: After all Clients have terminated, the Server continues to run.
Solutions: 1) The Server should automatically terminate after the last Client stops using it. If a Client does not properly disconnecting from the Server, it is possible that the Server will continue to Scan the Tags for Client's Groups that it is still maintaining on behalf of the Client. You must manually bring up the Server form the operating system's task list and click on the Exit under the menu item File. A dialog will tell you that Clients are still connected and ask if you wish to exit anyway. Click yes. 2) If a change has been made to the Server configuration that has not been save to file, then as the Server is shutting down, a dialog will ask whether to save the changes or not. Someone must answer this dialog before the Server will exit.

17.5 Configuration Problems

Can't define something (device, group, tag), Can't import OID file, Can't delete, move something

Problem: Can't create device.
Solution: Make sure that the IP and Services are correct and valid. If entering the values by name, i.e., hostname and/or service name, make sure that the respective entries have been made in the operating system's hosts and services files. If under Windows 95/98, you must make an entry in the hosts file even if the you are entering the IP address in dot notation.

Problem: Can't delete a Subgroup.
Solution: Subgroups cannot be deleted if they contain any Tags or contain any other Subgroups that contain Tags. Move all Tags from all Subgroup in the Subgroup to be deleted to other locations outside of the Subgroup to be deleted. Although not necessary, you may want to manually delete each Subgroup under the primary one you are trying to remove just to insure they are all empty.

Problem: Can't delete a Device.
Solution: Device's cannot be deleted if they are being referenced by any Client Group. A Device is being referenced if the Client has added one or more Items for Tags that belong to the Device to one of its Client Groups. Devices can also not be deleted if Tags are being scanned by the Server in its Monitor Mode. Shutdown all Clients referencing the Device to be delete or have them remove the Tags within Client Groups that are on the Device. Take the Server out of Monitor Mode.

Problem: Can't move a Tag from one Device or Subgroup to another.
Solution: You cannot move Tags that are referenced in a Client Group or are being updated by the Server in Monitor Mode. Remove the Tag from Client Groups. Take the Server out of Monitor Mode. You cannot move a Device's :StatusCode and :StatusMsg Tags. You cannot move Tags from one Device to another Device.

17.6 Runtime Communication Problems

Problems with the communications once the Server is started and is configured.

Problem: The Server communicates for a while without error, however, after some time, all requests begin to time out. The Comm Error window shows no response messages and the Communication Debug window shows no response error messages and messages indicating that responses were received with no matching outstanding request.
Solution: This indicates that requests are being sent to the controller(s) faster than the controller(s) can process them. The controller supports a multiple outstanding request capability. This means that the Server can send multiple requests (both read and/or write) to the controller before receiving a single response. The controller queues the requests internally and processes them as fast as it can. If it starts falling behind in it's processing, eventually as it responds to each request it has queued, the Server will have already given up on getting a response. The rate at which data is being requested must be reduced. This means setting the Clients to use slower update rates and/or reducing the frequency of Device reads and writes. Set Tags to use a larger Data Current Threshold setting may help reduce the actual number of request sent as well.

Problem: Asynchronous requests take longer and longer to complete.
Solution: After a minimal check to ensure valid Items, Asynchronous requests are queued in the Server for operation. There is a separate queue per Client Group. Depending upon the amount of Server activity, how fast the controller(s) are responding and on how many Asynchronous requests Clients are making of the Server, the queue may start filling up. You will notice that values written take longer to actually arrive at the Device and that even when you stop Client write and read operations, they continue anyway. 1) Switch to using Synchronous requests. These run to completion before returning to the Client thereby ensuring that the Client can never have a growing Asynchronous queue. 2) Reduce the rate at which Clients are making Asynchronous requests. 3) Eliminate Asynchronous read requests in favor of Scan Advise methods. 4) If you have groups of Item data that need to be written, configure the Clients to write them all at once rather than individually. This creates one request versa many.

This page is left blank intentionally.

18. DEFINITIONS

Advise - Method by which the Server returns data back to the Client. The advise may be the result of a previous Asynchronous read request, a Refresh request or as a result of the Server scanning data for a Client Group and finding that some of that data needs to be returned to the Client because it has changed.

Asynchronous - Operation where the Client makes a request of the Server to do an operation but does not wait for the operation itself to complete. The actual results of the operation are returned later to the Client once the operation is complete. (See Synchronous).

Cache Read - A read request made by a Client that retrieves data from the Server's cache values for the Tags rather than actually requiring the Server to go to the Device to read the data for the Tags. (See Device Read) The data returned will only be as up-to-date as the last time the data was actually read. The time stamp and Quality Information for that last read data is also returned.

Call Back - See Advise.

Client - Any OPC Client that connects to the Server in order to read and/or write to configured Devices.

Client Group - A collection of Client Items put together at the request of a Client for the purpose of scanning, reading and/or writing values to one or more configured Devices. A Client Group has properties such as a name, a scan rate, an active flag, etc. A Client Group can contain Items that are associated with Tags belonging to different Devices. Client Groups are maintained by the Server on behalf of the Client. Note that a Client Group is not the same as a Server configured Subgroup.

Communication Path - A configuration object that represents a pathway to access Devices.

Device - A configuration object representing an actual *controller*. It is represented by a Device Icon in the Device Pane.

Device Pane - The left half of the graphical configuration environment of the Server that shows a tree view of configured Devices and Subgroups.

Device Read - A read request made by a Client that instructs the Server to actually go to the Device(s) to read the Tag values rather than having the Server go to its Tag cache. (See Cache Read). The data returned will be time stamped provided with the Quality Information for the read.

Global Variable - A variable defined in PiCPro for a ladder program that is marked as global. Variables marked as global are placed in the OID File when the ladder is compiled and downloaded to the controller.

Item - Items are added to Client Groups by Clients. Each Item associates with a Tag configured in the Server. There can be many Items referring to the same Tag, Items from the same Client Group, from different Client Groups from the same Client or from Client Groups form different Clients. Items have properties such as name, active state, last value, last time stamp, etc.. An Item's name is the same as the Tag's name to which it is associated and is in fact how the connection is originally made from the Item to the Tag when the Client first adds the Item in a Client Group.

Item Error Information - A part of the results of a request operation from the Client to the Server such as read, write or refresh, the Server returns an array of error codes to the Client, one array entry corresponding to each Item in the request. Each error indicates any error condition encountered while servicing the request for that Item. Note that the error information is in addition to the overall return status of the request as well as in addition to any Quality Information returned per Item (as is provided in a read or refresh for instance).

Monitor Mode - A mode the Server can be put into where is periodically scans the Tags in the Selected Subgroup. It is intended for testing purposes only.

OID File - File generated from PiCPro during the compile and download of a ladder program to a *controller*. The OID File contains information on all of the global variables in the ladder program and is read by the Server during the configuration process.

Parent - The immediate Device or Subgroup in which a Subgroup is defined.

Pre-Built Command - A setting for the Server that causes it to "pre-build" commands necessary

Quality Information - Information is provided with each piece of data provided by the Server so that Clients can determine if the values are good or not. The Quality Information values are defined by the OPC specifications.

Refresh - A type request that can be made by Clients to the Server. Force a callback for all active items in the group (whether they have changed or not). Inactive items are not included in the callback. This is like doing a Device Read for all Items in the Client Group.

Scanning - Process of reading all the Tags for active Items in the Client Group. Any Tags that have new data as a result of the read are returned to the Client.

Selected Subgroup - This is the currently highlighted Device or Subgroup in the Device Pane. The Tags shown in the Tag pane belong to this highlighted item and any properties changed or actions taken such as deleting in the Device pane will be for this highlighted item. To select a different Device or Subgroup as the Selected Subgroup, simply click on it.

Server - The Giddings &Lewis OPC Server.

Subgroup - A configuration grouping for a configured Device in the Server. It is represented by a folder icon in the Device Pane. Subgroups are optional. They generally are configured to allow Tags to be organized in a way that makes sense to users in order to facilitate locating tags or by some sort of performance criteria. Note that this is not the same as a Client Group which are configured by Clients and are totally independent of the Subgroups configured in the Server.

Synchronous - Operation where the Client makes a request of the Server to do an operation and then waits for the operation to run to completion. (See Asynchronous).

Tag - An object in the Server that represents a global variable in the controller. A Tag will “belong” to a configured Device and has properties such as a name, read/write access settings, scaling and description. Tags are created for a Device as part of configuring the Device and importing an OID File.

Tag Configuration File - A Server configuration file. These files have the filename extensions tdb.

Tag Pane - The right half of the graphical configuration environment of the Server that shows a list view of configured Tags.

Time Stamps - Date and Time information returned with all Tag data, along with quality information, to Clients as the result of read and scan operations.

Index . . . .

I

L

M

N

O

P

Q

R

S

T

U

V

W