ROVISYS
Independent Automation Solutions

BridgeMaster ™
OPC Bridge Software

User Documentation

For

OPC Server to OPC Server Connectivity

1 Introduction

This document is intended to provide an overview of the BridgeMaster application. BridgeMaster is a software application that bridges data between any two or more OPC Servers. This interface supports bi-directional data flow and supervisory control between any two control systems. It enables operators to control digital and analog control loops implemented within DCS controllers or SCADA systems from other control system operator consoles such as the ABB Bailey INFI 90 Console. Likewise, operator stations may also be used for operator control of digital and analog control loops implemented in other control system controllers. These capabilities depend on the functionality supported by the OPC Server being utilized. The RoviSys OPC90 Server which is used to link with the ABB Bailey system supports all of these capabilities. BridgeMaster is fully compliant with Versions 3.0, 2.x and 1.a OPC Servers.

2 Functional Description

The BridgeMaster interface runs under any of the Microsoft Windows operating systems from XP and beyond.The two modes of operation associated with the interface are “Configure mode” and “Runtime mode”. In Configure mode a user can configure settings and create links (or tag mappings). In Runtime mode the interface uses the currently loaded configuration to communicate with the external OPC Server(s). BridgeMaster supports the following features:

- ✓ Support for OPC Data Access V3.0 compliant Servers, OPC Data Access V2.x compliant servers and OPC Data Access V1.a compliant servers.
- ✓ Dual server redundancy scheme supports redundant OPC Server connections implementing user-selectable fail-over mechanisms.
- ✓ Server Redundancy includes ability to define a heartbeat/watchdog tag per side.
- ✓ Ability to run as a Windows service for standalone, dedicated data transfer ability (without a user logged in).
- ✓ A BridgeMaster Control Panel application allows start and stop of the runtime service.
- ✓ Supports COM/DCOM
- ✓ Supports and implements OPC DA V3.x improved functionality such as:
    - ➢ Ability to transfer VQT (Value, Quality and Timestamp) on a group by group basis for all tags within a group.
    - ➢ Implements usage of Keep Alive mechanism for Redundancy.
    - ➢ Ability to set individual Item Deadband values on an item by item basis for noisy tags.
- ✓ User-Selectable two way data flow movement on a group by group basis
- ✓ Supports selectable User-Requested OPC Version to use on a group by group basis, defaulting to “Use Best Available Interface”. Each group provides feedback as to what OPC version is actually being used with the server.
- ✓ Supports ability to Bulk Add Items in order to speed up startup.
- ✓ Optimized data throughput via Bulk Tag Writes (VQT or ‘Value-Only’) during runtime.
- ✓ Allows selection of writes and reads to be made synchronously (polled) or asynchronously (on data change) on a group by group basis as well as selection of whether to transfer VQT or ‘Value-Only’ (if V3.x supported).
- ✓ OPC Server configuration dialog allows easy configuration of the OPC Server and node by browsing and selecting an existing OPC Server and node.

- ✓ Simple Tag Map definition dialog allows individual mapping of an ‘A’ side OPC server tag to a ‘B’ side OPC server tag. Tag Map definition includes defining attributes such as OPC Item names, description and data source. Optionally, scaling conversion using conversion tables can be defined to convert raw or discrete values from one system to the other.
- ✓ The Unlimited Version supports unlimited user-defined tag map definitions.
- ✓ Tag Map configurations are stored in a serialized compact binary file.
- ✓ Simple configuration with ‘Drag & Drop’ capabilities for mapping lots of tags in bulk quickly.
- ✓ Supports ability to easily Import/Export BridgeMaster Tag Map configurations to CSV files, which can be created and/or edited with Microsoft Excel or any text editor.
- ✓ Error and informational messages displayed in the main window in runtime along with the optional user-configurable specification of maintaining this data in daily log files.
- ✓ Runtime tag monitoring ability to view and change (read/write) data values in either OPC Server from main client window.
- ✓ Runtime ability to add or delete tag maps, change item names, and/or change group update rates, Item Dead Bands, Keep Alive rates, etc.
- ✓ Up to 100 connectors (OPC interfaces) can be defined.
- ✓ Ability to browse any local or remote OPC Server.
- ✓ Allows runtime view of OPC Server Status, performance statistics and visual representation of a changing heartbeat.
- ✓ Written using multi-threading and C++ for faster data transfer and higher performance.
- ✓ Written in UNICODE (for support of UNICODE text, such as Kanji OPC Item names)
- ✓ Tested with over 40,000 tags (20K Links).
- ✓ Provides shortcuts for commonly used utilities such as Ping, DCOM Configuration Utility, Services Manager and running an external OPC Client.
- ✓ Data Import: Provides capability to easily import configurations from other OPC Bridging software such as Kepware LinkMaster, Emerson’s OPC Mirror, Matrikon’s Data Manager, Iconic’s DataWorkX32 as well as from Honeywell Experion (PKS) OPC Integrator.
- ✓ Provides capability to auto-detect which of the above CSV (or TSV) file formats it is and directly import into BridgeMaster.
- ✓ Data Conversion: Updated support for additional scaling methods such as Linear, Square root, Gain/Offset, BCD-8, BCD-16, OPC Quality to Value, OPC Timestamp to Value and Array Element conversions.
- ✓ Range Limit Clamping: Added support to clamp values to the output range or to a specific user-defined range. Added support to set the limit bits if value is clamped.
- ✓ Expanded support for the Discrete scaling to include Long to Long, Long to String and String to Long conversions.
- ✓ Expanded support for automatic data conversion of all ARRAY types including VT_I8, VT_UI8, VT_I4, VT_UI4, VT_I2, VT_UI2, VT_I1, VT_UI1,VT_R8, VT_R4, Boolean (VT_BOOL), currency (VT_CY), strings (VT_BSTR), dates (VT_DATE) and VT_VARIANT.
- ✓ Validate Connector- Added a validate method to verify all connectors, groups, tag maps, OPC Items and all Data Conversion tables.
- ✓ Limit bits-Added ability to display the limit bits for each tag value in the tag view.

3 Getting Started

This section will provide the required information to setup and configure both the BridgeMaster software as well as the communications interface to an OPC Server.

3.1 Configuring the OPC Server(s)

Before BridgeMaster can work properly in runtime, each OPC Server must be installed and configured correctly. Follow the installation and setup procedure specified with the documentation from the OPC Server vendor. A TCP/IP connection must also exist between the BridgeMaster node and the remote node running the OPC server you wish BridgeMaster to connect with. Consult the section “Using DCOM OPC Servers” at the end of this document to setup and configure remote OPC servers.

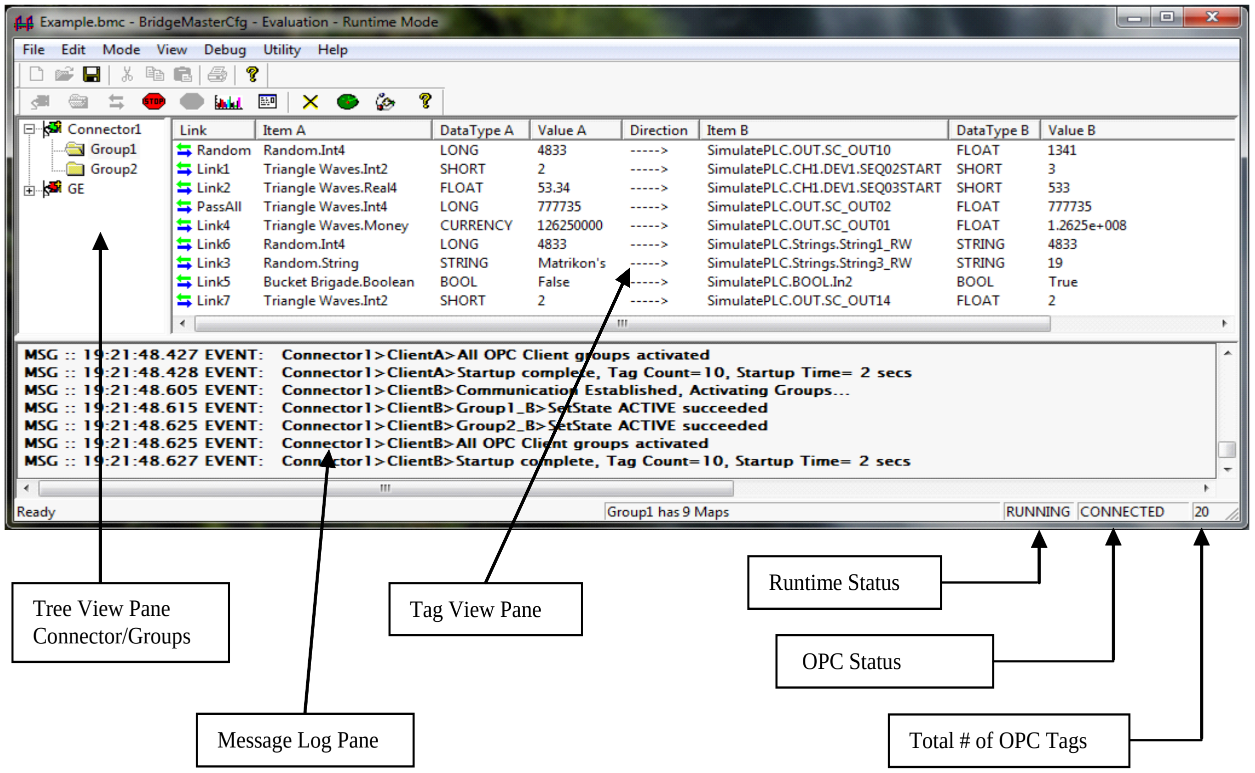
3.2 Configure Mode

In Configure mode the BridgeMaster Options along with the BridgeMaster configuration can be configured from the main window. BridgeMaster does not communicate with OPC Servers in Configure mode. Select *Mode* **|** *Configure* to change to Configure mode. Changing from runtime to Configure mode will disestablish all OPC tag items and groups and stop all communication with the OPC Servers. You can also click on the STOP button in runtime or press the keys <Shift+F5> to change back to Configure mode.

3.3 Runtime Mode

To change BridgeMaster to runtime select *Mode* **|** *Runtime*, click the GO button or press the F5 control key. While in Runtime mode the driver will act as an OPC Client to both OPC servers. On startup of Runtime mode, all OPC Groups and Items are established with the specified OPC Server(s). If one or more OPC servers cannot be connected to upon startup for the currently loaded configuration then an error will be logged to the log. Selection of *Configure* from the Mode menu item will also place the driver back into Configure mode.

When in Runtime mode BridgeMaster displays the following. Note: the currently selected Tag Map configuration is displayed in the title of the main window, for the example below this is displayed as <Example.bmc>.



Link	Item A	DataType A	Value A	Direction	Item B	DataType B	Value B
Random	Random.Int4	LONG	4833	----->	SimulatePLC.OUT.SC_OUT10	FLOAT	1341
Link1	Triangle Waves.Int2	SHORT	2	----->	SimulatePLC.CH1.DEV1.SEQ02START	SHORT	3
Link2	Triangle Waves.Real4	FLOAT	53.34	----->	SimulatePLC.CH1.DEV1.SEQ03START	SHORT	533
PassAll	Triangle Waves.Int4	LONG	777735	----->	SimulatePLC.OUT.SC_OUT02	FLOAT	777735
Link4	Triangle Waves.Money	CURRENCY	126250000	----->	SimulatePLC.OUT.SC_OUT01	FLOAT	1.2625e+008
Link6	Random.Int4	LONG	4833	----->	SimulatePLC.Strings.String1_RW	STRING	4833
Link3	Random.String	STRING	Matrikon's	----->	SimulatePLC.Strings.String3_RW	STRING	19
Link5	Bucket Brigade.Boolean	BOOL	False	----->	SimulatePLC.BOOL.In2	BOOL	True
Link7	Triangle Waves.Int2	SHORT	2	----->	SimulatePLC.OUT.SC_OUT14	FLOAT	2

```
MSG :: 19:21:48.427 EVENT:   Connector1>ClientA>All OPC Client groups activated
MSG :: 19:21:48.428 EVENT:   Connector1>ClientA>Startup complete, Tag Count=10, Startup Time= 2 secs
MSG :: 19:21:48.605 EVENT:   Connector1>ClientB>Communication Established, Activating Groups...
MSG :: 19:21:48.615 EVENT:   Connector1>ClientB>Group1_B>SetState ACTIVE succeeded
MSG :: 19:21:48.625 EVENT:   Connector1>ClientB>Group2_B>SetState ACTIVE succeeded
MSG :: 19:21:48.625 EVENT:   Connector1>ClientB>All OPC Client groups activated
MSG :: 19:21:48.627 EVENT:   Connector1>ClientB>Startup complete, Tag Count=10, Startup Time= 2 secs
```

3.4 BridgeMaster Licensing

BridgeMaster is a licensed product, and as such requires a software license to operate properly. Two modes of operation are associated with this licensing: Production mode and Demo Mode. Without a valid license present, BridgeMaster will run in Demo mode for up to 1 hour (use LicenseManager to add more demo time) and will stop communicating if in runtime, restarting BridgeMaster will restart demo operation. An unlimited number of tag maps are supported in Demo Mode. With a valid license, BridgeMaster will run in Production mode indefinitely. Two Production Mode licenses are applicable:

- **BridgeMaster Lite Version** – Supports up to 250 total tag maps (or links). Since each Link consists of 2 tags, the Lite version actually supports up to 500 tags. The tag maps are split equally amongst all connectors and all groups, for instance if you have 5 Connectors with 2 Groups each, then each connector is allowed to have up to 250/5= 50 tag maps each, and each of the 2 groups within each connector can have up to 50/2= 25 tag maps. The number of connectors is limited to 100.
- **BridgeMaster Unlimited Version** – Supports unlimited number of tag maps (or links) and tags. The number of connectors is still limited to 100.

The RoviSys License Manager program is used to install a BridgeMaster software license. If USB licensing is being used, License Manager can be used to report the state of the USB license. It can also be used to setup a demo license longer than the standard 1 hour. Installation of BridgeMaster includes the RoviSys License Manager program. It is easy to use and contains help to walk you through the simple licensing procedure. It is run from the Windows “Start | RoviSys LicenseManager” shortcut.

3.5 Running BridgeMaster as a Service

The BridgeMaster application actually contains three runtime programs:

- BridgeMaster Configurator (BMC)
- BridgeMaster Service (BMS)
- BridgeMaster Control Panel (BMCP)

The BMC and BMS programs contain the same runtime core code, however a single configuration can only be run at any given time in either one or the other, but not both programs. The user selects whether the interface is to run as a Windows System Service (via the BMS) or directly on the desktop application (via the BMC). To change the application to run as a service select the Options dialog, click the Service tab and check the *Automatically Startup as Service* checkbox. The default is to NOT run as a service, and therefore when the user changes to Runtime mode, the Configurator will run the project internally in the background (without the service being required). The BMC is used mostly for initial tag configuration of the project and the BMS is used to execute the configuration in runtime after everything is setup. However, as stated the BMC can also be used to run the project in runtime.

The important points to keep in mind in choosing which method to use are itemized below:

<u>When running BridgeMaster interface as a system service [BMS]</u>:

- After initially setting up and configuring a BridgeMaster system, this is the recommended and preferred method to use for final commissioning.
- Can be placed in Runtime mode or Configure mode from either the BMC or the BMCP.
- Service automatically starts up in Runtime mode using the last BMC Configuration.
- Service can be started or stopped from BridgeMaster Control Panel.
- User can logoff and interface will still run.
- PC can be rebooted without any user logging in and BMS will still startup.
- Messages can be viewed in the Log File (by running BMC and selecting (*View* **|** *Log File…*) from the BMC main menu.
- Daily Log files created from the Service have the format “BMS Log <date> <time>.log”
- When the user changes to Runtime mode from BMC, the BMC will startup the service (if BMS is not already running) and then place the BMS in Runtime mode with the current configuration of the BMC. The BMC configuration will need to be saved (if it was modified) before going to runtime.
- When the BMS is running and the BMC starts up, the BMC detects that the interface is running as a service and will connect to it and will automatically detect the mode of the BMS (Runtime or Configure). The BMC is always “ONLINE” with the service when the service is running; you can not do offline configuration on the same PC with a running BMS. Any configuration changes made in the BMC while the service is running will be made to the running BMS.
- When the user changes to Configure mode from BMC, the BMC will place the BMS in Configure mode, and BMS will still be running as a service, it will just not be running the interface and connecting anything.

- Keep in mind that when running as a service and connecting to remote servers, that the security credentials must be setup correctly for *BridgeMasterService* in order to access the remote server. This is accomplished by setting the BMS to run under a particular user account with access to the remote server node.
- Will only run for 1 hour if a Demo mode is used (and additional demo time hasn't been assigned), service will need to be restarted after that.
- Tag values can be monitored/read/written in the service from the BMC.
- Online changes to the configuration can be made in runtime from the BMC and will be sent to the BMS in runtime. This means for instance you can add/edit/delete tag maps and change update rates on the fly. However, when running as a service in Configure mode, any changes made within BMC will not be sent to the BMS until the user starts in Runtime mode again.
- In Runtime mode, the BMC will not be allowed to change (open or create a new) configuration during Runtime mode. The user must change to Configure mode in order to change configuration files.

<u>When running BridgeMaster interface from the Configurator [BMC]</u>:

- Can be placed in Runtime mode or Configure mode.
- User cannot logoff the PC otherwise interface will stop.
- After PC is rebooted a user must login (or be automatically logged in) in order for BMS to startup
- Messages can be viewed directly in the message log pane (as well as saved to a log file).
- Daily Log files created from Configurator have the format "BMC Log <date> <time>.log"
- Will only run for 1 hour if a Demo mode is used (and additional demo time hasn't been assigned), BMC will need to be restarted after that.
- Tag values can be directly monitored/read/written in the BMC.
- In Runtime mode, the BMC will not be allowed to change (open or create a new) configuration during Runtime mode. The user must change to Configure mode in order to change configuration files.
- BMC does not require the BridgeMaster Service in any way.

3.5.1 Remote Connections

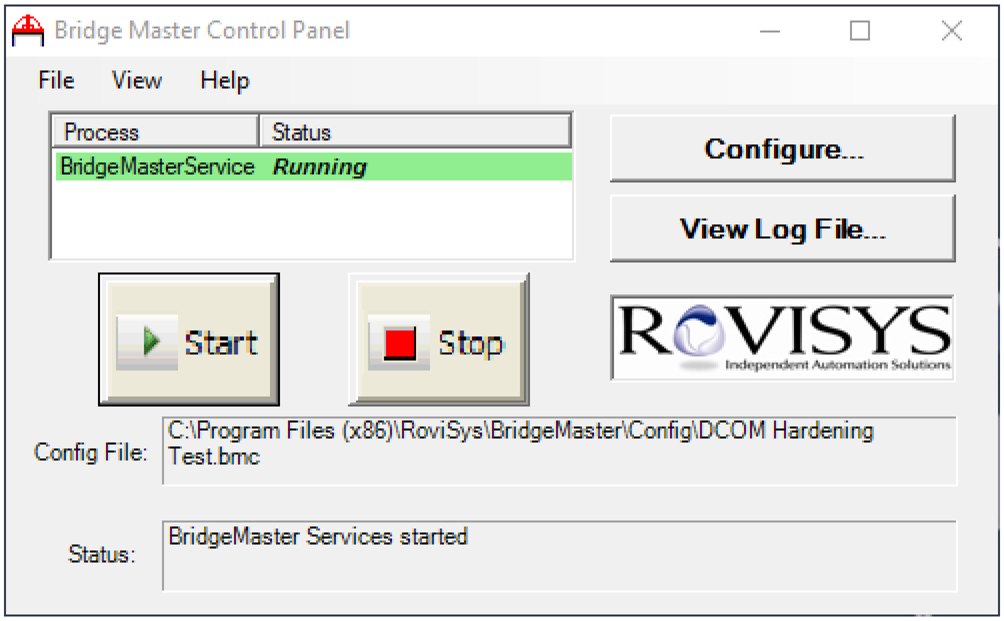
Also note when running BridgeMaster as a Service and one or more of the OPC Servers exist remotely, then the security needs to be setup correctly on the BridgeMaster service. This is configured by setting up the BridgeMaster service with a "Logged On User" which has permissions to the server on that node.

3.5.2 DeltaV Connections

If connecting to the OPC Server of the Emerson DeltaV system, it should be noted if BridgeMaster is running on the DeltaV Application Station and BridgeMaster is setup to run as a Service, which is recommended, then the BridgeMaster service should be setup to "Depend On DeltaV" so that it does not start up before the DeltaV system is ready, otherwise adverse states may occur with DeltaV, and DeltaV may not be able to startup the Historian.

BridgeMaster Control Panel

Launching the BridgeMaster service can be accomplished by using the utility program called the “BridgeMaster Control Panel”. This program can be used to startup the BridgeMaster service if it is not already running or to stop the service if it is running. It can also be used to start the BridgeMaster Configurator program as discussed above. To run the control panel, select (*Start* | *All programs*) and then browse to the RoviSys BridgeMaster section and select the “BridgeMaster Control Panel”. The main window looks like the following when the service is running:



Process	Status
BridgeMasterService	*Running*

Config File: C:\Program Files (x86)\RoviSys\BridgeMaster\Config\DCOM Hardening Test.bmc

Status: BridgeMaster Services started

This application when minimized will be placed in the system tray. A balloon popup will appear on any change of state in the service (stopped, started). A status window is provided to display whether an operation succeeded or not. Note: BridgeMaster must be setup to “*Automatically Startup as Service*” in order to use this application properly. If BridgeMaster is not setup to run as a service then the BMS will automatically shut down upon startup.

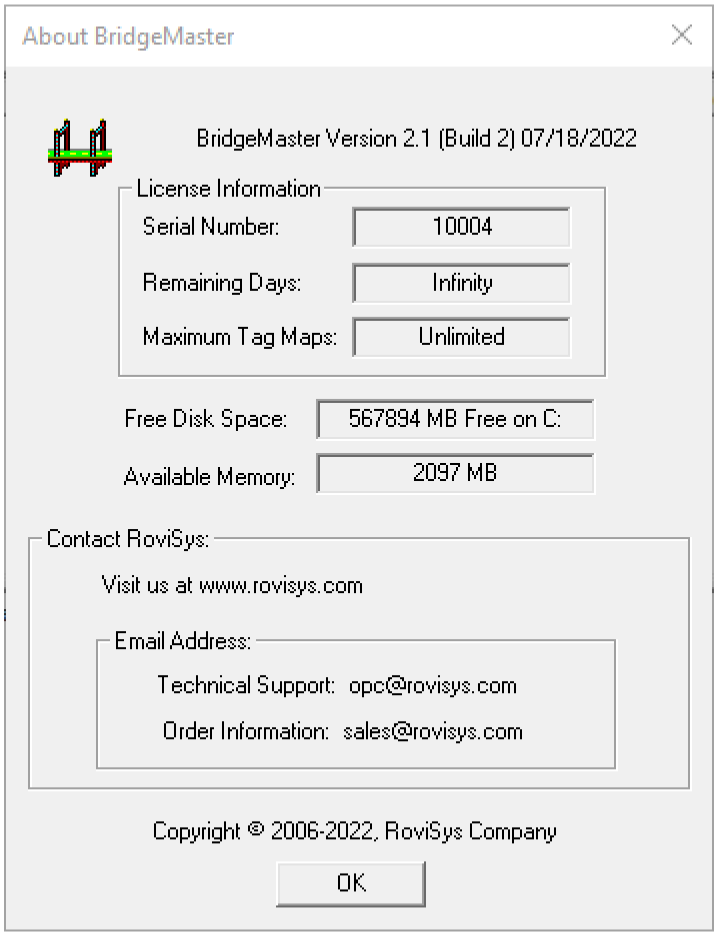
After clicking the start button, the Control Panel application will start the service. The following operations are allowed using the BridgeMaster Control Panel:

- Start- This button will start the BridgeMaster Service (BMS).
- Stop- This button will stop the BridgeMaster Service (BMS)
- Configure…- This button will start the BridgeMaster Configurator (BMC).
- View Log File…- This button will bring up the latest log file in notepad.

Note, if the BridgeMaster service is not installed. The status will show up as “Not Installed” in yellow.

3.6 BridgeMaster Version

The version of the BridgeMaster can be found by selected the *Help* menu in the main window. The version label below displays Major Version as 2, Minor Version 1 and the Build # is 9 built on July 18, 2021. Also displayed in the About dialog box is the License Information.



About BridgeMaster

BridgeMaster Version 2.1 (Build 2) 07/18/2022

License Information

Serial Number:	10004
Remaining Days:	Infinity
Maximum Tag Maps:	Unlimited

Free Disk Space:	567894 MB Free on C:
Available Memory:	2097 MB

Contact RoviSys:

Visit us at www.rovisys.com

Email Address:

Technical Support: opc@rovisys.com

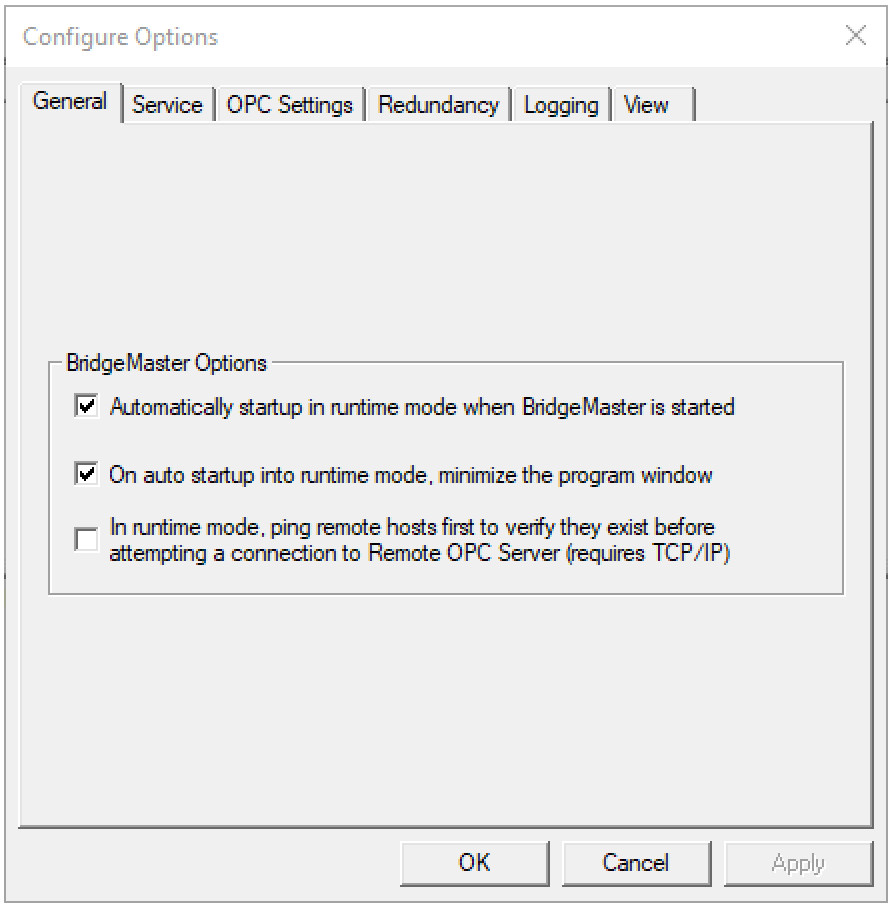
Order Information: sales@rovisys.com

Copyright © 2006-2022, RoviSys Company

3.7 Configuring BridgeMaster Options

To configure BridgeMaster Options, select *Configure Options…* from the *Utility* menu, or click on the button in the tool bar. The following displays the General tab of the Options dialog.

3.7.1 General Options Tab



BridgeMaster Options

[✓] Automatically startup in runtime mode when BridgeMaster is started

[✓] On auto startup into runtime mode, minimize the program window

[ ] In runtime mode, ping remote hosts first to verify they exist before attempting a connection to Remote OPC Server (requires TCP/IP)

A description of each option in the General tab is given below.

Automatically startup in runtime mode when BridgeMaster is started (checkbox)

Check this option to startup the program in runtime mode when running as BMC (the next time the BMC program is started). If unchecked then you must manually place BridgeMaster in runtime mode. Default is unchecked.

On Startup into runtime mode, minimize the program window (checkbox)

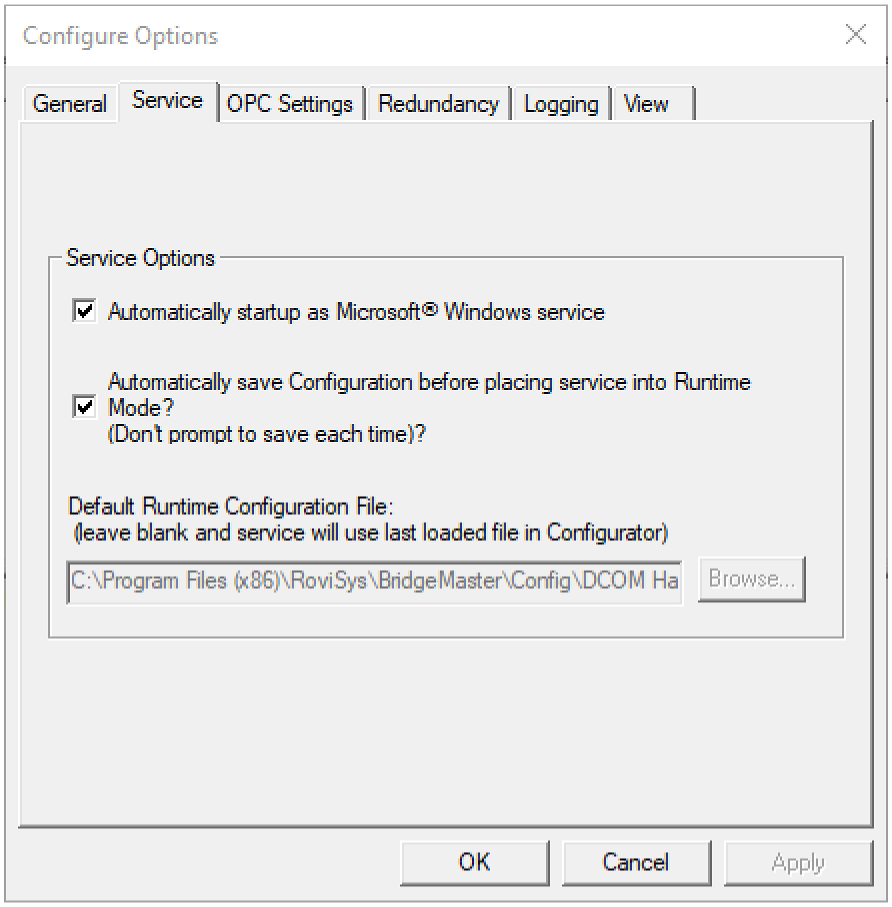
This option is used to control whether or not the BMC program starts up minimized or not. On startup of BridgeMaster if the "*Automatically startup in runtime mode…*" is checked above and this is checked then the program window will be minimized. Default is unchecked.

In runtime mode, ping remote hosts first to verify they exist before attempting a connection to Remote OPC Server (requires TCP/IP) checkbox:

This option should typically not be checked. If checked, BridgeMaster will first ping the remote host before attempting an OPC connection to it. Note, if this option is unchecked and a remote server becomes unavailable then BridgeMaster may hang trying to connect with it, depending on how DCOM is setup with the COM interface (OPC Server). Note, for local servers (running on the same PC as BridgeMaster) ping is not used and thus this option is irrelevant. Default is unchecked. This setting can be changed on the fly in runtime.

3.7.2 Service Tab

The next tab in the Options dialog box is the Service tab which provides general settings for running BridgeMaster as a Windows System Service (BMS) as shown below.



Service Options

☑ Automatically startup as Microsoft® Windows service

☑ Automatically save Configuration before placing service into Runtime Mode?
(Don't prompt to save each time)?

Default Runtime Configuration File:
(leave blank and service will use last loaded file in Configurator)

C:\Program Files (x86)\RoviSys\BridgeMaster\Config\DCOM Ha

The Service options are specified below:

Automatically startup as a Microsoft Windows Service:

This option is used to specify that BridgeMaster should run as a Windows System Service (as BMS) as opposed to running in the background within the BridgeMaster Configurator (BMC). See the *Running BridgeMaster as a Service* Section for more details.

When BridgeMaster is setup to run as a service:

- The BridgeMaster Service (BMS) will automatically startup running when the computer restarts without any users logged in.
- When BMS starts up, it will use the last loaded configuration of the BMC.

When BridgeMaster is running as a service:

- When you startup BridgeMaster Configurator (BMC), BMC will automatically connect to and be ONLINE with the BMS. BMC will then transparently display (take on) the state of the BMS: either running or stopped. Whatever is happening in the BMS will be reflected in the BMC.

- After BMC is connected to the BMS, anything you do in the BMC will be reflected within the BMS. For instance, if you click the STOP button, the BMS will be placed back into Configure mode. If you click START, the BMS will go to Runtime mode. If you add or delete a Link or Tag Map in runtime, this will also be reflected within the BMS. When you make any online tag or group edits (name change, update rate change, etc..) this will further be reflected in the running BMS.
- In Runtime mode, the BMC program can be exited and the BMS will still be running as a service.
- BMC acts like a Data Spy and/or surrogate application to the BMS that can not only peak into (but also control) the runtime operations of the BMS.

Automatically save Configuration before placing service into Runtime Mode?:

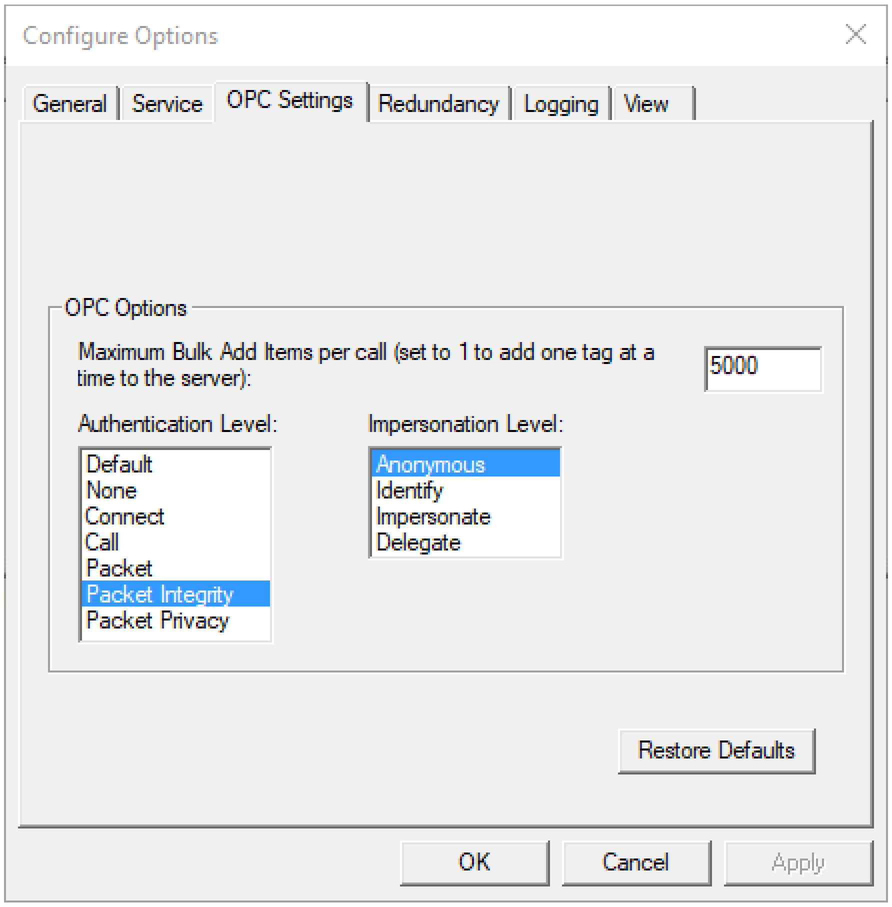
When running as a service, this option allows the user to go straight into Runtime mode without being prompted to save the configuration file (if it needs saving). Select this checkbox and BridgeMaster will automatically save the current configuration (if it was modified) before going to runtime. Even if you didn't directly change the configuration, BridgeMaster internally may have indirectly changed something. For instance, it does this when it reads the actual data type back for an Item ID and when it determines the actual OPC Version being used. When running as a service, BridgeMaster cannot be placed into runtime mode without first saving the configuration file (if it was modified). This is because during startup the BMS must run with the same configuration that BMC is using, otherwise a mismatch might occur.

Default Runtime Configuration File (leave BLANK and service will use last loaded file in Configurator):

This option is used to specify the runtime configuration file for the BridgeMaster Service (BMS). Currently, this is not user-selectable (since OFFLINE edits are not allowed at this time) and the BMS must always load and run the last configuration file of the BMC. Since the BMC acts as a surrogate to the BMS, it must always be running with the same configuration file. This option is used to verify that the BMS is running the same file that the BMC has loaded.

3.7.3 OPC Settings Tab

The next tab in the Options dialog box is the OPC Settings tab which allows setting OPC options as shown below.



OPC Options

Maximum Bulk Add Items per call (set to 1 to add one tag at a time to the server): 5000

Authentication Level:
- Default
- None
- Connect
- Call
- Packet
- Packet Integrity [✓]
- Packet Privacy

Impersonation Level:
- Anonymous [✓]
- Identify
- Impersonate
- Delegate

A description of each option in the OPC Settings tab is given below:

Maximum Bulk Add Items per call (set to 1 to add one tag at a time to the server):

This option is used during runtime startup when establishing items with each OPC Server. When adding OPC Items to the server, this option specifies how many items (tags) can be added in bulk at a time per call. BridgeMaster attempts to establish this many items in one call to the server. If it needs to make more calls then it does so until all items are established. Bulk adding tags to the server can speed up startup time with certain slow servers. As an extreme example, with trying to establish a 10,000 tag configuration with a remote APACS OPC Device Server, the startup time may take 6 hours if only 1 tag is established at a time, whereas it may only take 50 seconds if this setting is set to 5000. This option can be changed on the fly in runtime.

Authentication Level:

This option defines the DCOM authentication level BridgeMaster should use when connecting with OPC servers.

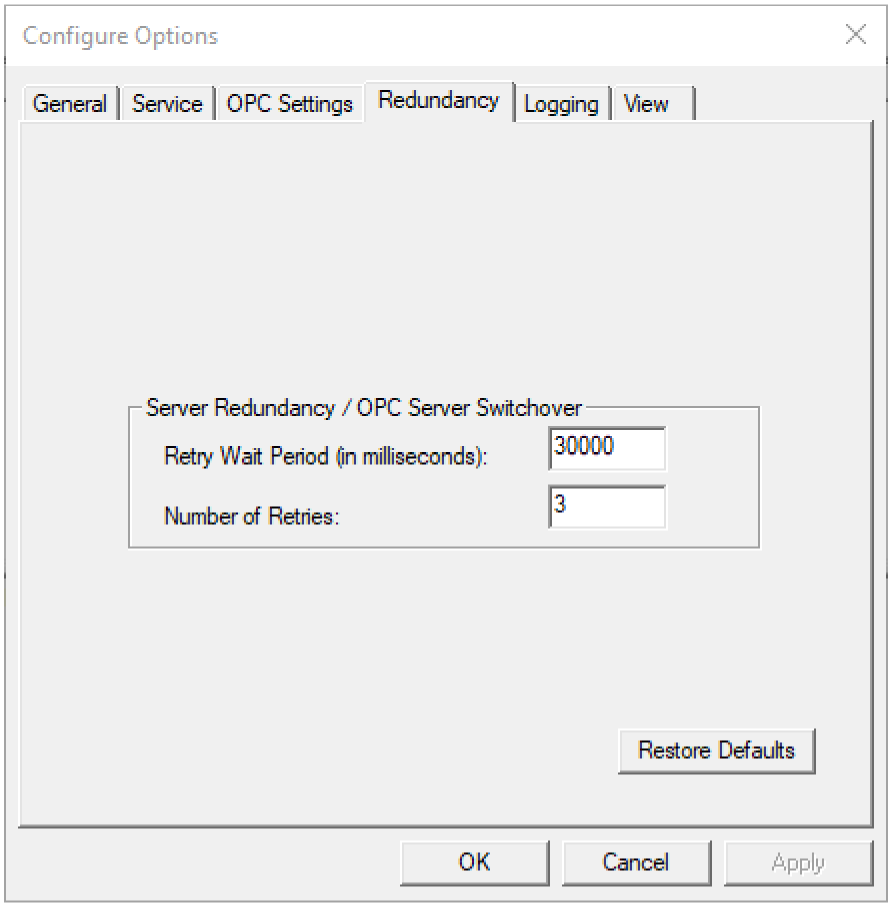
Impersonation Level:

This option defines the DCOM impersonation level BridgeMaster should use when connecting with OPC servers.

The *Restore Defaults* button will restore the default settings for the OPC Settings tab.

3.7.4 Redundancy Tab

The next tab in the Options dialog box is the Server Redundancy tab which provides general settings for the Server Redundancy options as shown below.



Server Redundancy / OPC Server Switchover

Retry Wait Period (in milliseconds): 30000

Number of Retries: 3

The Server Redundancy / OPC Server Switchover options are specified below:

Retry Wait Period (in milliseconds):

This option specifies the retry wait period before attempting another connection with an OPC Server that has stopped communicating. This is used in runtime mode during a failover. A connection attempt is made with the OPC Server at this interval (in milliseconds). This setting can be changed on the fly in runtime.

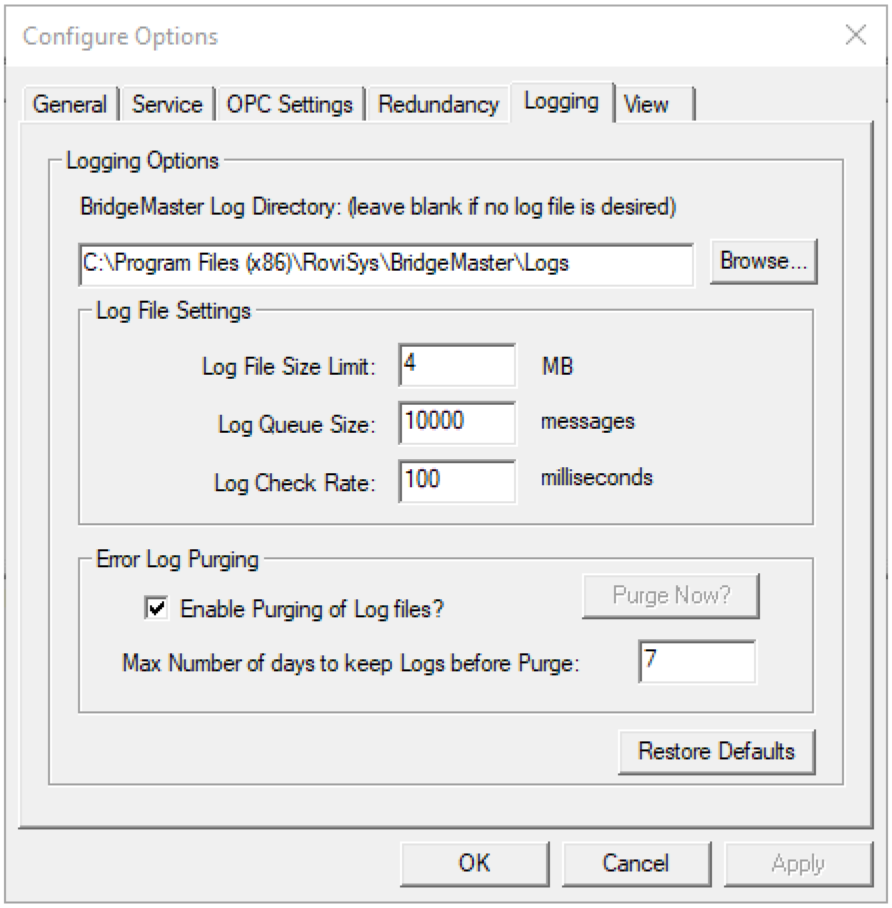
Number of Retries:

This option specifies the number of retries to attempt a connection with a redundant OPC Server after communication is lost with that server before switching over again. For example, if this is set to 3 and communication becomes lost with the primary server, then BridgeMaster will switchover to the secondary server (if defined) and attempt a connection 3 times to the secondary server before switching back to the primary. It will then attempt a connection 3 times with the primary and so on. This setting can be changed on the fly in runtime.

The *Restore Defaults* button will restore the default settings for the Redundancy tab.

3.7.5 Logging Tab

The next tab in the Options dialog box is the Logging tab which provides general settings for the Logging options as shown below.



Logging Options

BridgeMaster Log Directory: (leave blank if no log file is desired)

C:\Program Files (x86)\RoviSys\BridgeMaster\Logs

Log File Settings

Log File Size Limit: 4 MB

Log Queue Size: 10000 messages

Log Check Rate: 100 milliseconds

Error Log Purging

[✓] Enable Purging of Log files?

Max Number of days to keep Logs before Purge: 7

The Logging options are specified below:

BridgeMaster Error Log Directory:

This option is used to specify the destination path of the error log directory. If the directory is entered in this dialog box then all error and informational messages displayed in the message/log pane will also be logged to a daily log file. If this option is left blank then logging of messages to files will not occur, regardless of the show flags.

Log File Size Limit:

This option is used to specify the maximum log file size per file. If a message going to a log file would make the log file get bigger then this then a new file will be created with a BMC_Date_1.log, BMC_Date_2.log, etc.. convention

Log Queue Size:

This option is used to specify the size of the queue for the logged messages.

Log Check Rate:

This option is used to specify how often new messages are checked for in the queue.

Enable Purging of Log Files?:

This option is used to specify that log file be purged. If you want to keep all of your log files (and never want some of them deleted) then uncheck the checkbox.

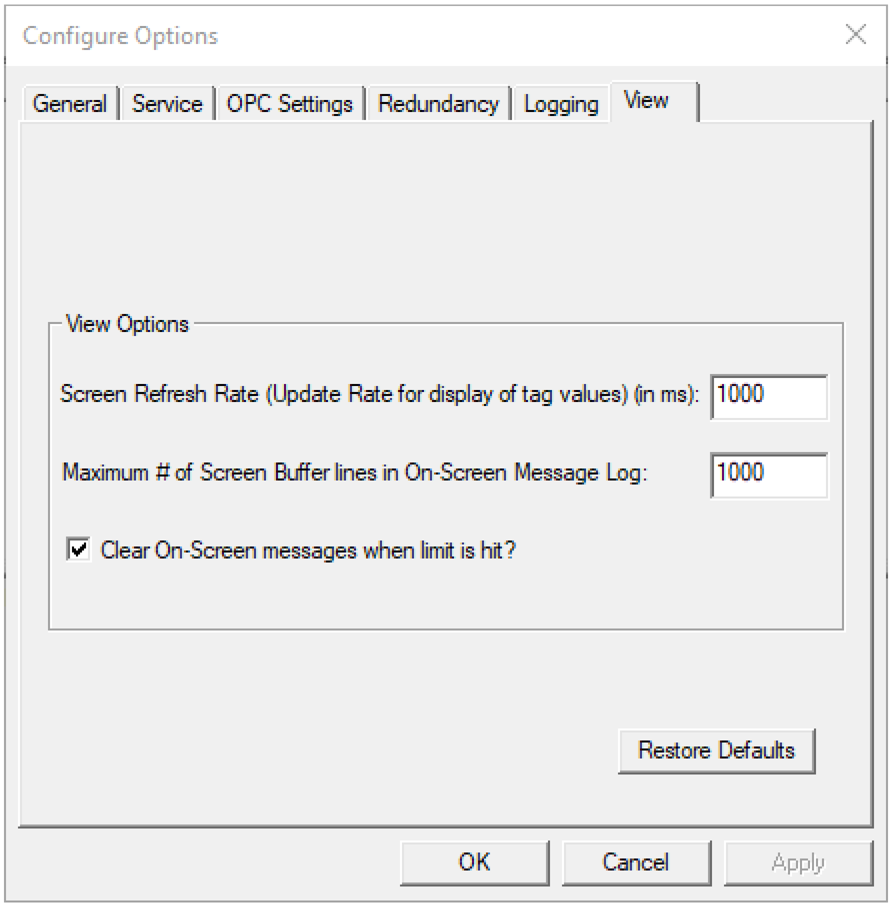
Maximum number of days to keep log files before Purge?:

This option is used to specify which log files will be automatically purged. Only files with the “BM*.log” will be purged from the Log Directory if they are older than this number of days. Checking for purging occur on application startup and at the end of the day.

The *Restore Defaults* button will restore the default settings for the Logging tab.

3.7.6 View Tab

The next tab in the Options dialog box is the View tab which provides general settings for the Viewing options as shown below.



Screen Refresh Rate (Update Rate for display of tag values) (in ms): 1000

Maximum # of Screen Buffer lines in On-Screen Message Log: 1000

[✓] Clear On-Screen messages when limit is hit?

The View options are specified below:

Screen Refresh Rate (Update Rate for display of tag values (in ms):
This option is used to specify how often in milliseconds tag values will be refreshed in the tag view pane.

Maximum # of Screen Buffer lines in On-Screen Message Log:
This option is used to specify the maximum lines displayed in the Message Log Pane. If the current number of lines is greater or equal to this the first 50% of these messages will be removed. This only concerns on-screen messages, whereas messages logged to the log file will not be removed. This option is used to minimize memory usage. The buffer can consume up to 1 MB of memory.

Clear On-Screen Messages when limit is hit?:

This option is used to specify that when the number of lines displayed in the Message Log Pane hits the limit defined above then the whole buffer (all on-screen messages) are cleared. If unchecked then only the first 50% will be removed.

The Restore defaults button will restore the default settings for the View tab.

4 BridgeMaster Tag Map Configuration

To interface to OPC items (also known as tags) from one OPC server to another a BridgeMaster Tag Map Configuration can be created. This configuration is saved as a binary file and is loaded upon startup of BridgeMaster. It contains the Links (or Tag Mappings) from one system to the other for which data is to be exchanged using the OPC protocol. A configuration within BridgeMaster is laid out as a hierarchy of user-defined objects. The top-level object is defined as a **Connector**, since it basically sets up and defines which OPC Servers are to be connected together for data exchange. The Connector object allows definition of a primary and if required secondary node hosting an optional redundant OPC Server. The next level down is called a **Group** and contains the actual tag maps. Each Group object can contain as many tag mappings as desired for the given license and is a way of collectively "grouping" similar tags together. The group is similar in concept to a folder directory on a file directory system, whereas the Connector object would be compared to the disk drive (i.e. C:\). Each group defines its own OPC Interface version to use, data update rates, refresh rates, dead band, time bias and data flow direction for all tag mappings contained or defined within it. In runtime, each defined BridgeMaster group object creates and encapsulates two (2) OPC Client Groups (those defined by OPC standards), one for each OPC Server specified in the Connector for either side. The next object is the actual **Link** Object (also known as a Tag Map). A tag map specifies the actual OPC Item IDs and Access Paths within each OPC Server, the data flow/ data master (the OPC Item originating the first value, if different from the default data direction specified in the group) and scaling options used for data value conversions. In runtime, the each Link object actually contains two (2) OPC Client Item objects (those defined by OPC standards), one for each server. Each OPC Item inherits properties specified from its parent Group and Connector objects.

4.1 Creating a New BridgeMaster Configuration

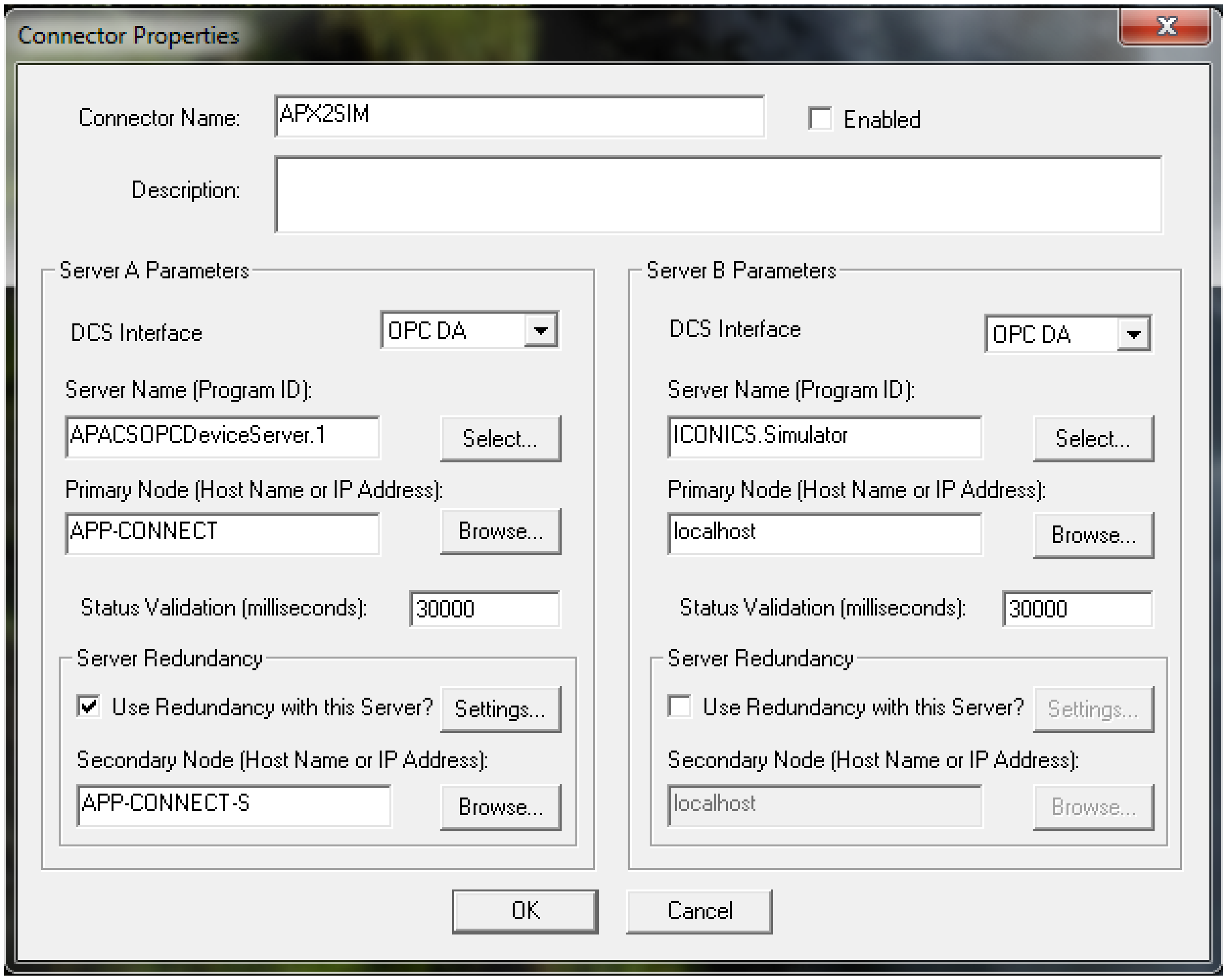
In Configure mode, a new BridgeMaster configuration can be created by simply selecting <*File* **|** *New…*> from the main menu. An empty connector called *Connector1* will be displayed in the tree view part of the main window. To save the configuration select <*File* **|** *Save*> and give it a meaningful name or select *File* | *Save As…* to copy or backup the existing configuration to another file. By default, all BridgeMaster Tag Map Configuration filenames have a ".bmc" file name extension.

4.2 Opening a BridgeMaster Configuration

In Configure mode, an existing BridgeMaster configuration can be opened by simply selecting <*File* | *Open…*>. The user-selected file will be loaded into the tree and tag view panes and can then be edited or placed into Runtime mode. To edit an object such as a Connector, Group or Tag Map, simply double-click on it or highlight the object and then select <*File* | *Properties…*> from the main menu. Note, BridgeMaster will not load or open a configuration in Runtime, the interface must be placed into Configure mode first. Some properties are not editable in runtime mode. However, BridgeMaster can save a configuration in runtime, since it allows tag maps or OPC Items to be created, deleted or changed on the fly in runtime.

4.3 Creating a New Connector

In Configure mode, a new connector can be created by selecting <*Edit* | *Add New Connector...*> from the main menu or pressing the keys <Ctrl+P>. The Connector may then be configured by double-clicking on it or by selecting it and then selecting <*Edit* | *Properties*…> from the main menu. Using the Connector Properties dialog box (shown below) you can define parameters for Server 'A' and Server 'B'. Server Redundancy is supported if the user selects the *Use Redundancy with this Server?* checkbox and fills in the remote node information in the respective Secondary Node edit box. Otherwise, the Secondary Node can be left blank or set to "localhost" if server redundancy is not used or required.



Connector Name: APX2SIM
Enabled: ☐
Description:

Server A Parameters
DCS Interface: OPC DA
Server Name (Program ID): APACSOPCDeviceServer.1
Primary Node (Host Name or IP Address): APP-CONNECT
Status Validation (milliseconds): 30000
Server Redundancy
☑ Use Redundancy with this Server?
Secondary Node (Host Name or IP Address): APP-CONNECT-S

Server B Parameters
DCS Interface: OPC DA
Server Name (Program ID): ICONICS.Simulator
Primary Node (Host Name or IP Address): localhost
Status Validation (milliseconds): 30000
Server Redundancy
☐ Use Redundancy with this Server?
Secondary Node (Host Name or IP Address): localhost

Using the Connector Properties dialog box, select a meaningful name for the Connector or leave it at the default. A text description box can be used to enter any desired comments about the interface or it can be left blank. First, browse for or enter the host name(s) on which the OPC Server(s) exist. Then select the OPC Server Name for the OPC Server 'A' side and the OPC Server 'B' side for the new connector. To browse for an OPC Server click the *Select*… button. To browse for a computer node select the *Browse*… button. If the interface is to use redundancy, then select the *Use Redundancy with this Server?* checkbox. If so, then select the desired secondary node and setup the redundancy settings by clicking the *Settings*… button. Note, remote servers do not have to be installed locally in order to connect with them remotely.

Note: if you are trying to browse remote servers, make sure you read section on OPCEnum. BridgeMaster uses the OPCEnum service on the remote machine to browse for remote OPC servers, therefore OPCEnum must be installed, running and accessible on the remote server. If OPCEnum cannot be run on the remote machine, BridgeMaster includes the ability to define OPC server program IDs and their class IDs in a text file called "BMLookupCLSID.txt". BridgeMaster will generate this file automatically as remote OPC Servers are browsed via OPCEnum. BridgeMaster users can use notepad to edit this file and add entries to the file pertaining to OPC Servers installed on a remote PC that is not running OPCEnum. Each line in the file defines one OPC Server. The format of each line is simply:
Program ID,{ClassID}

A comma separates the OPC server program ID and Class ID. The Class ID is enclosed within open and ending braces {}. The OPC server program ID and ClassID can usually be determined from the OPC server manufacturer. It can also be determined by searching the Windows registry. Searching for the name of the server executable will find the necessary information. Following is some example server definitions from an actual BridgeMaster "BMLookupCLSID.txt" file.

```
Ovation.OPC.4,{4525AD4E-17BD-11D5-8183-0050046FAB6D}
RoviSys.OPC90Server.1,{2508F89D-3490-11D3-9236-0040339FD356}
RoviSys.DLEOPCServerDA.3,{34B16516-ECFD-4F3A-8FCA-007E9B8D0ABE}
ICONICS.Simulator.1,{585DED25-936B-11D2-A8DC-00A024C111A3}
ICONICS.ModbusOPCServer3.1,{8F090662-9345-11D1-9BFC-00A024E7BC48}
SSNET.OPC.1,{937F9090-7888-4E55-8AEB-AD8E058044D5}
ICONICS.ModbusEthernetDA.2,{9BC87883-EEDA-11D3-9FDE-006067705B5A}
ToolWorX.Modbus.1,{9D8B4CA7-C7A3-11D3-A61B-00104B0D13C3}
ICONICSInc.IniFileRuntimeDA.2,{9E51F386-0712-11D4-B0B6-009027242C59}
ICONICS.SimulatorOPCDA.2,{A879768A-7387-11D4-B0D8-009027242C59}
National Instruments.DaqOpc.1,{9C491D07-4CD1-11D4-AE17-00105A1115C3}
Intellution.OPCiFIX.1,{3C5702A2-EB8E-11D4-83A4-00105A984CBD}
APACSOPCDeviceServer.1,{87782911-389B-11D4-BE59-005004773FEE}
OPC.DeltaV.1,{C3B72AB1-6B33-11D0-9007-0020AFB6CF9F}
DeltaV.DVSYSsvr.1,{FE199392-EBEA-11D4-8233-00C04F60F056}
Intellution.OPCEDA.3,{58C41832-474B-11D2-BE25-006097C8F3ED}
EEI.CSVReplaySimulator.1,{3FBB69A5-3E6B-11D4-8416-00008639559B}
HWHsc.OPCServer,{6031BF75-9CF2-11D1-A97B-00C04FC01389}
AE_PI_OPC,{B0496231-21FE-432D-A2AC-14B1D8ABF39E}
OSI.HDA.1,{1F7365EE-7CBB-4867-8CFC-54FD50FF67D6}
RSLinx OPC Server,{A05BB6D5-2F8A-11D1-9BB0-080009D01446}
RSLinx Remote OPC Server,{A05BB6D6-2F8A-11D1-9BB0-080009D01446}
OSI.DA.1,{B9662477-8C52-44C2-A16B-DE04DCBC45D9}
DSxPOpcSimulator.TSxOpcSimulator.1,{DC07E136-BC50-4108-9818-5A693D148ADA}
RSI.RSView32OPCTagServer,{E9F9ED00-7705-101B-9802-0000C07B665C}
Intellution.OPCEDA.1,{81F29A20-850C-11D0-B37C-00A024BC1942}
ToolWorX.ModbusDLL.1,{900CC3C1-E237-11D2-A594-00104B0D13C3}
DELTAV.SisOpcSvr.1,{771CE139-41B9-49D1-BE2A-B01993C9364B}
INAT TcpIpH1 OPC Server,{3DA28330-68CB-11D2-9C65-0021A0020009}
Intellution.SPServer.5,{68DEE241-F375-11D2-ADE0-006008B04755}
RoviSys.ModbusTCPServerDA.3,{485AD958-7D12-4381-B212-868F3261DB76}
Rovisys.ModbusRTUServerDA.3,{8BB17412-3861-42CF-800A-1A7ECFE40873}
ICONICS.ModbusOPCServer3.20,{8F090662-9345-11D1-9BFC-00A024E7BC48}
CIMPLICITY.HMI.OPCServer.1,{B01241E8-921B-11D2-B43F-204C4F4F5020}
```

A description of each Connector property is given below:

Connector Name
Select a meaningful name for the connector or leave it at the default.

Description
A text description box is provided which can be used to enter any desired comments about the interface. This can be left BLANK.

Enabled
The *Enabled* checkbox will enable or disable the connector and all of its children (groups and tag maps) in runtime from communicating with each other. Verify this checkbox is checked if you want the Connector to run in runtime mode, otherwise it will be disabled. A disabled connected will show up as Red in the tree view pane. This is useful when you have multiple connectors configured, one for each of several external servers, and at some point, one of the OPC servers or remote system is brought down (for maintenance, for example) and you don't want or need that connector interfacing with the downed OPC Server. In this case you would simply uncheck the *enabled* checkbox for that connector. Otherwise, if left enabled, it may take away processing power from the interfaces on the other connectors due to timeouts and/or DCOM problems. You can now enable and/or disable any and all connectors in runtime. Disabling a connector in runtime is like stopping the interface for that connector; the actual runtime interface thread will be stopped. Likewise, enabling a connector in runtime is like starting the interface for that connector or placing it into runtime mode; the actual runtime interface thread will be started.

Server Name (Program ID)
Enter the OPC Program ID of the OPC Server you wish to connect with. You can browse for servers on the Primary node by selecting the *Select*… button. A "Select OPC Server" dialog box containing all installed OPC Servers on the local (or remote) machine will be displayed (on the host specified in the Primary Node). To browse local OPC Servers on your local node, set Primary Node field to "localhost" or leave it blank.

Primary Node (Host Name or IP Address)
Enter the Host Name or IP Address of the Primary computer node for which the OPC Server you wish to interface to exists on. To browse for a computer node select the *Browse*… button.

Use Redundancy with this Server?
This checkbox indicates that redundancy will be enabled with this OPC server side. The following settings are provided to allow selection of when and under what conditions should BridgeMaster consider failing over to this secondary node. Note, the same OPC Server must exist with the same tag configuration (same Item IDs) that the primary is using in order for redundancy to be effective, although the OPC Server need not be the same version.

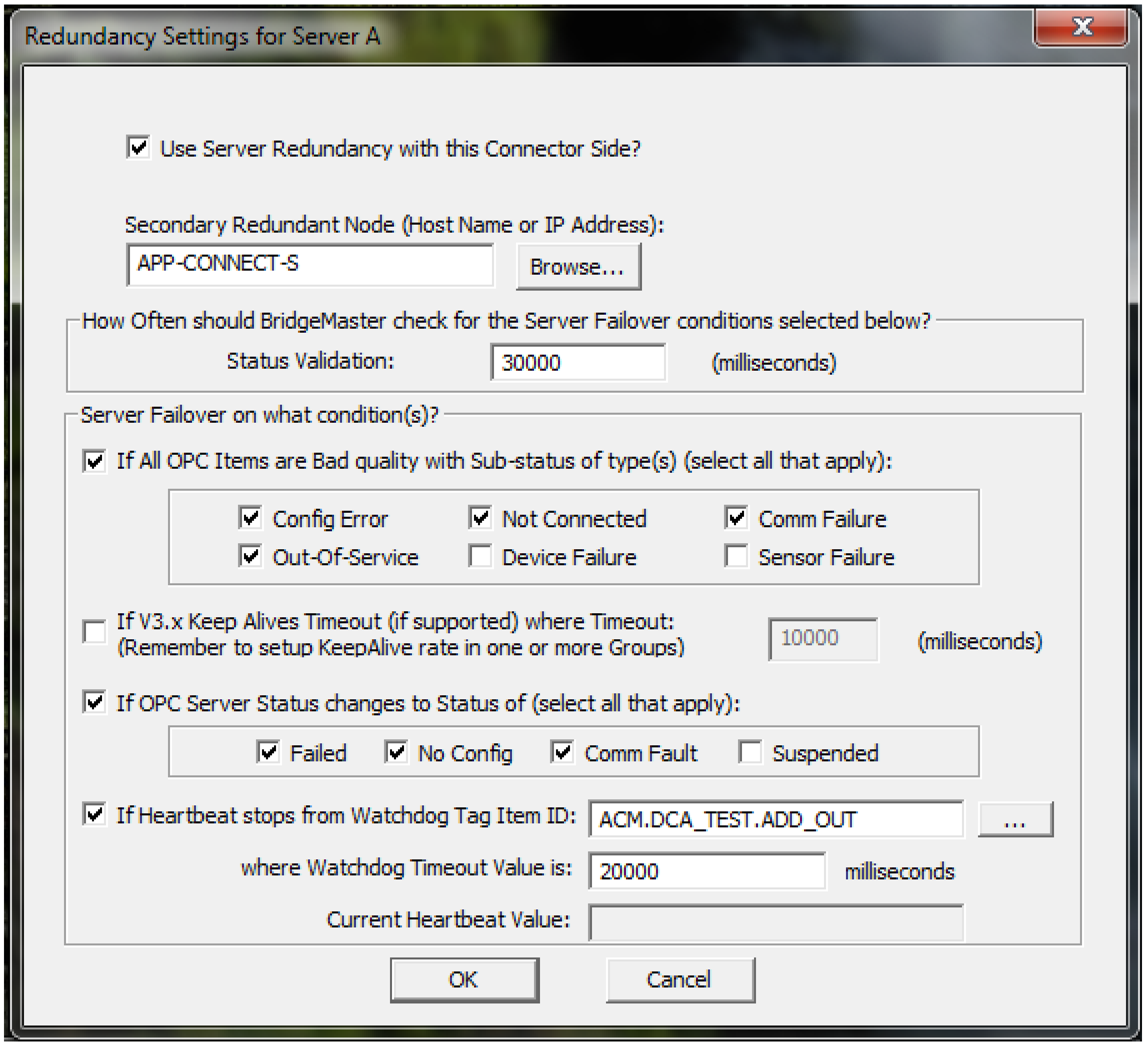
Secondary Node (Host Name or IP Address)
Enter the Host Name or IP Address of the secondary (redundant) computer node for which the OPC Server you wish to interface to exists on. To browse for a computer node select the *Browse*… button.

Status Validation
The *Status Validation* edit box is used to specify the interval in which status is to be checked or validated with the OPC Server. BridgeMaster will validate the status of the primary and secondary OPC servers at the rate specified by this dialog. If redundancy is configured then this

is used to determine if a fail over to the redundant OCP Server is required. It is specified in milliseconds with a default of 30000.

4.4 Redundancy Settings

BridgeMaster supports redundant connections on each server side of the interface; this is termed as "Server Redundancy". Since both sides of a given connector can be configured with two (2) redundant servers, this implies a dual redundancy scheme. This however is not to be confused with "System Redundancy" or BridgeMaster System Redundancy where you would have two (2) BridgeMaster nodes or two BridgeMaster systems that were redundant with one another. BridgeMaster system redundancy is targeted to be supported in the next version! If the primary node is down for any of the selected reasons, then BridgeMaster will automatically detect this and failover all tag and group connections to the server on the redundant node. For either server side of a given connector call up the Redundancy Settings dialog box from *Settings*… button in the Connector Properties dialog box. The following dialog will appear.


Redundancy Settings for Server A

[✓] Use Server Redundancy with this Connector Side?

Secondary Redundant Node (Host Name or IP Address): APP-CONNECT-S

How Often should BridgeMaster check for the Server Failover conditions selected below?
Status Validation: 30000 (milliseconds)

Server Failover on what condition(s)?

[✓] If All OPC Items are Bad quality with Sub-status of type(s) (select all that apply):
[✓] Config Error [✓] Not Connected [✓] Comm Failure
[✓] Out-Of-Service [ ] Device Failure [ ] Sensor Failure

[ ] If V3.x Keep Alives Timeout (if supported) where Timeout: 10000 (milliseconds)
(Remember to setup KeepAlive rate in one or more Groups)

[✓] If OPC Server Status changes to Status of (select all that apply):
[✓] Failed [✓] No Config [✓] Comm Fault [ ] Suspended

[✓] If Heartbeat stops from Watchdog Tag Item ID: ACM.DCA_TEST.ADD_OUT
where Watchdog Timeout Value is: 20000 milliseconds
Current Heartbeat Value:

The settings are described below:

Use Redundancy with this Connector Side?
This checkbox indicates that server redundancy will be enabled with this connector side. The following settings are provided to allow selection of when and under what conditions should BridgeMaster consider failing over to the server on the secondary node. Note, the same OPC Server must exist with the same tag configuration (same Item IDs) that the primary is using in order for redundancy to be effective, although the OPC Server need not be the same version.

Secondary Redundant Node (Host Name or IP Address)
Enter the Host Name or IP Address of the secondary computer node for which the redundant OPC Server you wish to interface to exists on. To browse for a computer node select the *Browse*… button.

Status Validation
The *Status Validation* edit box is used to specify the interval in which status is to be checked or validated with the OPC Server. When redundancy is enabled, this implies how often BridgeMaster will check the selected failover conditions. This is used by BridgeMaster to determine if a server fail over to the redundant OCP Server is required. It is specified in milliseconds with a default of 30000.

The Redundancy failover conditions are listed below. These conditions are logically 'OR-ed' together, meaning if any one **or** all of the selected failover conditions occur during the status validation then a failover to the server on secondary redundant node will occur. For each server side, Status Validation does not occur until startup completes. Specifically, this means that Status Validation does not occur until all tags of all groups for each respective server side have been added to the OPC Server and all (user-specified active) groups have been set active.

Notes:
1) Startup will complete regardless of whether or not tags were added successfully to the server.
2) When a group is set active this means data callbacks from the Server to BridgeMaster will occur.
3) User-specified active means you can specify a group is to be made Active or Inactive during runtime; the Inactive groups won't count.

If all OPC Items are Bad Quality with Sub-status of type(s) (select all that apply):
This checkbox will enable or disable failover when ALL Configured OPC Items are Bad quality of a certain type. Configured means the OPC Item was defined and has a non-blank tagname. For instance, some redundant OPC Servers (like ProcessLogix) will set all OPC Items to 'Out-of-Service' BAD quality when the node is unavailable or becomes the backup node and thus this can be used as a trigger for BridgeMaster to failover to the redundant server. These settings are logically 'OR-ed' together meaning if ALL configured items within the server side have BAD quality with any of the selected sub-status selections then failover will occur. In other words, if just one tag has GOOD or QUESTIONABLE quality, then failover will not occur due to this condition. Also, if one tag has BAD Quality with a sub-status that was not selected then failover will not occur due to this condition.

If V3.x Keep Alives Timeout (if supported) where Timeout:
This checkbox will enable or disable failover based on a Keep Alive timeout. If the OPC Server supports OPC Data Access v3.x Interfaces, then this setting is applicable. The Keep Alive

mechanism can be setup in each group to occur at a specified interval. However, only one group per server side is required to be setup with a Keep Alive. This mechanism is used to keep the connection open. When Keep Alives are enabled, the OPC Server sends an empty callback to BridgeMaster at the Keep Alive rate specified in each group if data has not changed in the group on the OPC server. If a data callback occurs with actual data, this also counts as a Keep Alive. The Keep Alive timeout period (specified in milliseconds) can be used for failover when the Keep Alives or callback interface fails for any one of the groups defined with an enabled Keep Alive rate (rate not equal to 0). It is recommended that you set this timeout value to a value of at least 2 times that of the shortest Keep Alive rate for any of the groups for each respective server side. Timeouts are checked at the Status Validation period, and one or more Keep Alives may have timed-out in between that period but a Keep Alive may have been received right before the validation check making the status OK. Once a group's Keep Alive was found to timeout for any group then a failover will occur (even if a Keep Alive arrives 0.001 seconds later). This is why it is important to not set the timeout period too low. Failing to set the Keep Alive timeout period high enough may cause premature failover. By default this option is left unchecked.

If OPC Server status changes to status of (Select all that apply):
This checkbox will enable or disable failover when the status of the OPC Server changes to any of the selected status definitions. For instance, if an OPC Server cannot communicate with its underlying communication system to access tag data then it should set its status to FAILED. If the Failed checkbox is selected and the server is FAILED then a failover will occur. On startup, under certain conditions a server may not have a tag configuration in which case the status will be NO CONFIG, in which case you may want to use the redundant node. A COMM FAULT status indicates that some (but not all) items are having difficulties communicating with the underlying hardware I/O, it is left up to the discretion of the user to use this server status as a failover condition or not. The same is true for the SUSPENDED status.

If Heartbeat stops from Watchdog tag of Item ID:
This setting will enable watchdog functionality and when enabled will failover when the watchdog tag stops changing value. The watchdog tag acts as a heartbeat (similar in function to the V3.x Keep Alives) and when it so to speak 'flat lines' and no heartbeat (value change) is received within the Watchdog timeout period then a timeout occurs which will cause a server failover to occur for that connector side.

Watchdog Timeout Value:
If the Watchdog functionality is enabled then this setting specifies how long to go before timing out when no change in value occurs. The watchdog timer starts after all tags are established and made active with a given OPC Server side, this is done in case a server takes a long time establishing tags with the server perhaps due to the quantity of tag maps defined. (Specified in milliseconds)

Current Heartbeat Value:
In runtime the current heartbeat value of the watchdog tag is displayed in this edit box. This is displayed in order to provide the user with affirmative visual feedback that the server is really alive and values are being received.

4.4.1 Failover

When using server redundancy, the whole point is to failover to the redundant server when you cannot communicate with the server on the primary node. This implies if there is a total failure to communicate with the primary node, then you will want to failover. As such, in addition to the user-selectable failover options listed in the previous section, BridgeMaster detects fatal conditions which imply total communication failure and will thus automatically failover on these conditions. For instance, if BridgeMaster detects that the server (or node) cannot be communicated with then a failover will occur.

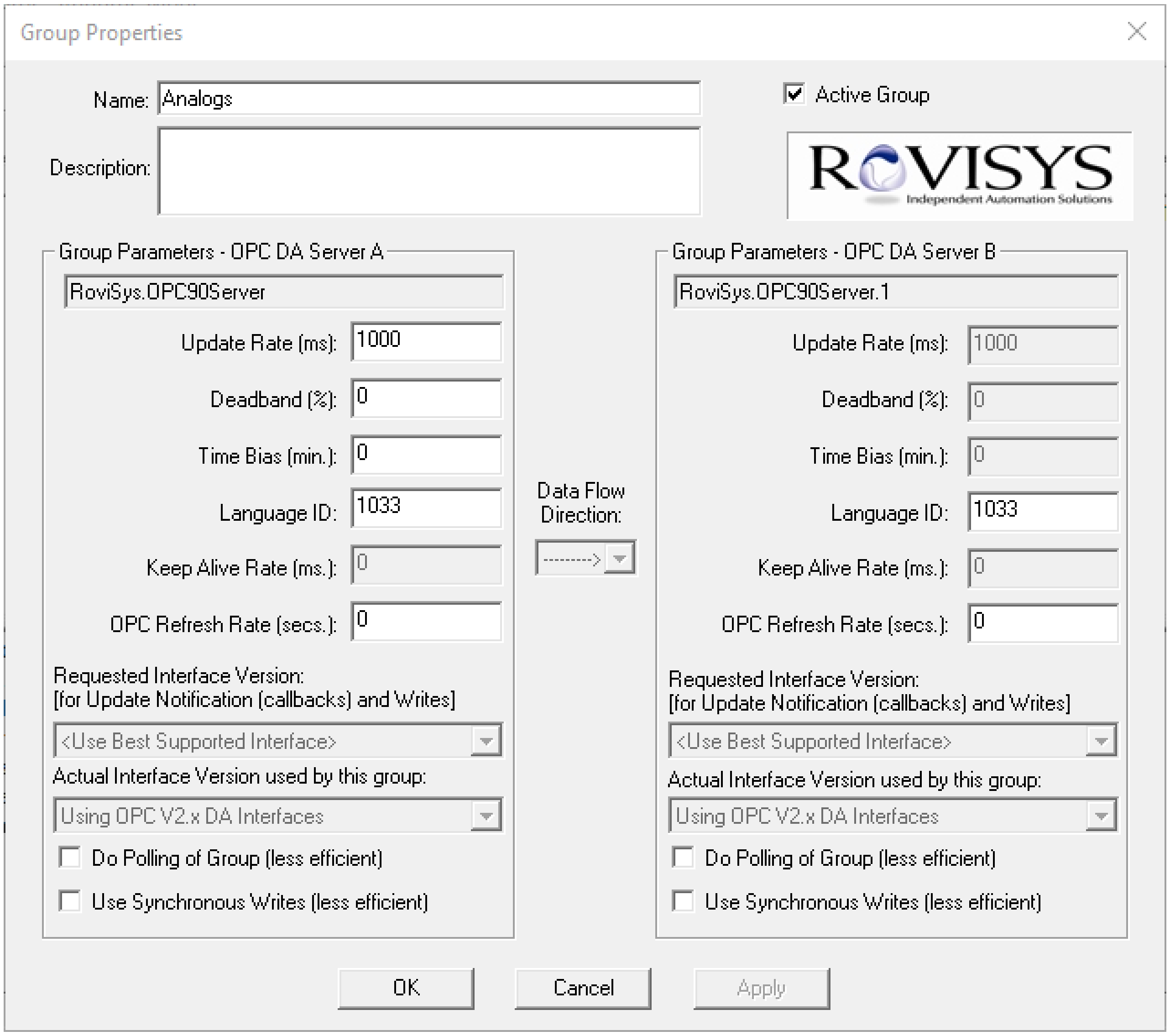
A Fatal condition exists in runtime after BridgeMaster receives 5 in row errors of any of these nasty errors of the variety (from OPC calls to the server):

- E_NOINTERFACE – No such interface supported. If we get just 1 of these then we failover.
- E_OUTOFMEMORY – Server ran out of memory. If we get just 1 of these then we failover.
- E_ACCESSDENIED – General access is denied error. If we get just 1 of these then we failover.
- RPC_E_NO_GOOD_SECURITY_PACJKAGES – No secuirty package installed. If we get just 1 of these then we failover.
- CO_E_REMOTE_COMMUNICATION_FAILURE – Remote communication to the computer providing the server failed. If we get just 1 of these then we failover.
- CO_E_SERVER_EXEC_FAILURE – Server executuon failed or the server crashed or is otherwise failed If we get just 1 of these then we failover.
- OPC_E_INVALIDCONFIGFILE – The Server's Config file is an invalid format. If we get just 1 of these then we failover.
- RPC_S_SERVER_UNAVAILABLE – Server is unavailable, this is the mother of all fatal communication errors, which provides indication that server crashed (or the process was terminated by the user or prematurely) If we get 5 of these in a row then we failover.
- RPC_S_UNKNOWN_AUTHN_SERVICE – The authentication service is unknown. If we get just 1 of these then we failover.
- RPC_S_SERVER_TOO_BUSY – The RPC Server is too busy to complete the operation. If we get 5 of these in a row then we failover.
- RPC_S_CALL_FAILED – The remote procedure call failed. This usually indicates that the Remote Server (EXE) was terminated. If we get 5 of these in a row then we failover.

Furthermore, an automatic failover will occur if the OPC Server supports the *IOPCShutdown* Interface and sends a shutdown request to BridgeMaster.

4.5 Creating a New Group

In Configure mode, a new group can be created by first selecting or highlighting the desired connector for which to add the new group to and then selecting <*Edit* | *Add New Group...*> from the main menu or pressing the keys <Ctrl+G> or right mouse clicking on the Connector and then selecting “Add Group”. The Group Properties dialog box (shown below) will be displayed for the new group and can then be edited.


Group Properties

Name: Analogs

Description:

☑ Active Group

Group Parameters - OPC DA Server A

RoviSys.OPC90Server

Update Rate (ms): 1000

Deadband (%): 0

Time Bias (min.): 0

Language ID: 1033

Keep Alive Rate (ms.): 0

OPC Refresh Rate (secs.): 0

Requested Interface Version:
[for Update Notification (callbacks) and Writes]

<Use Best Supported Interface>

Actual Interface Version used by this group:

Using OPC V2.x DA Interfaces

☐ Do Polling of Group (less efficient)

☐ Use Synchronous Writes (less efficient)

Data Flow Direction: -------->

Group Parameters - OPC DA Server B

RoviSys.OPC90Server.1

Update Rate (ms): 1000

Deadband (%): 0

Time Bias (min.): 0

Language ID: 1033

Keep Alive Rate (ms.): 0

OPC Refresh Rate (secs.): 0

Requested Interface Version:
[for Update Notification (callbacks) and Writes]

<Use Best Supported Interface>

Actual Interface Version used by this group:

Using OPC V2.x DA Interfaces

☐ Do Polling of Group (less efficient)

☐ Use Synchronous Writes (less efficient)

The Group Properties dialog box allows specification of the OPC Group parameters affecting all tag maps and OPC Items contained within each respective group. Define a descriptive name for the new group. A description for the group can also be entered (if desired).

The group parameters for the A and B sides are defined below:

Group Name
Select a meaningful name for the group or leave it at the default. This name must be unique amongst all groups in the connector.

Description
A text description box is provided which can be used to enter any desired comments about the group. This can be left BLANK.

Active Group
The *Active Group* checkbox specifies the group parameter's active flag and is passed to the OPC Server in runtime. This checkbox will enable or disable the group and all Links (Tag maps) contained within it in runtime from communicating with each other. Inactive groups will be displayed in the tree view pane as a pink folder. Data for the group and all tag maps contained within it can only be received or sent if the group is active within the server. Verify this checkbox is checked if you want the Group to run in runtime mode, otherwise it will be disabled. A disabled (Inactive) group will neither receive nor transmit values for any items defined within in it. Note: All tags within an Inactive group are still established in runtime with the server. You can set a group inactive in runtime to disable updates for all the group's items.

Data Flow Direction
This combo box sets the data flow direction between tags of the Server 'A' side to tags of the Server 'B' side for all tag maps within the group. Note this parameter cannot be changed in runtime. The direction can be one of the following:

→ Specifies that data is read from the items within the Server 'A' of this group and written or sent to their mappings in Server 'B' of this group.

←→ Specifies bi-direction data transfer where data flows in both directions and is read and written from the items within the Server 'A' and written or read from their mappings in Server 'B'. The Data Flow / Data Master defined within the Tag Map definition defines which server originates the value first upon startup of runtime mode. After startup, when a value changes on either side it is sent to the other side.

← Specifies that data is read from the items within Server 'B' of this group and written or sent to their mappings of Server 'A' of this group.

Update Rate (milliseconds)
This group parameter specifies the update rate (in milliseconds) for the OPC group. This rate is specified to the OPC Server when the OPC Group is created in runtime mode. It tells the OPC server how often we want to receive the data or how often it should check for changes in OPC Item values defined within tag maps within this group. Most OPC Servers are exception-report based, meaning they will send data when it changes. However, some servers will send data at this rate even if it does not change. This option can be changed in runtime mode.

Deadband (%)
This group parameter specifies the deadband (in percentage of full scale range) of an OPC Item. For analog points the EU Full Scale range exists within the OPC Server as EU_HIGH - EU_LOW. Most OPC Servers use this parameter to determine when to notify a client of a change in value. If the value is noisy, you may not want the OPC Server sending each value unless it changes by more than this percentage.

Time Bias

This group parameter specifies the Time bias (in minutes) from the local node to the remote node. This is passed to the OPC Server along with the deadband and Update Rate to tell the OPC Server to offset each timestamp for each OPC Item within this group by the specified amount. This is useful for checking or comparing timestamps on OPC Items, which may exist in different time zones. However, this is not a particularly useful parameter, aside from the fact that most OPC Servers do not utilize this parameter; BridgeMaster itself does not perform any timestamp Time zone conversion. It is a required parameter for creation of an OPC Group within an OPC Server and thus is left for user-specification for possible future implementation. Default is 0.

LCID

This group parameter specifies the Language ID of all OPC Items for the group. Most OPC Servers do not support this but if they do they would use this parameter to determine what language to use (for example for error strings or for String (VT_BSTR) data tags and labels (like ON/OFF and OPEN/CLOSE for digital points in English or Japanese). BridgeMaster supports the transfer of VT_BSTR (2 byte UNICODE character strings) in any language, however the displays and dialog boxes are currently always displayed in English. Default LCID is English (United States) = 1033. A value of 0 means no LCID. See the LCID Document for lookup values for any particular desired language.

Keep Alive Rate (milliseconds) (V3.0 only)

This group parameter specifies the Keep Alive rate (in milliseconds) for the group connection. This is used only by Version 3.x Servers. This tells the OPC Server how often to send an ‘Empty’ callback (with no data values) for this group in order to keep the connection alive. This is useful when redundancy is enabled. Callbacks with actual data which occur at the Update Rate also act as a Keep Alive so if the callbacks are occurring at this rate then the Server may not even need to ever send an empty Keep Alive packet. The Keep Alive rate must be at least twice as fast (shorter) then the Keep Alive Timeout value in order to not cause a premature failover. This can be changed on the fly in runtime. A warning will be given to the user before the dialog closes if the user is attempting to set it too low. Also if the Show Errors is enabled for the connector of the group and the Server does not support the rate specified, then an error message will be logged. Most V3.0 servers do not support Keep Alive rates faster than 1000 milliseconds (1 second). Set to 0=Disabled.

Note: When using redundancy only 1 group is actually required to enable the Keep Alive mechanism, the rest can be set to 0. As an extreme case, if you have 1000 groups each configured with the Keep Alive of 1000, then that will certainly be a lot of additional unneeded traffic on your network.

OPC Refresh Rate (seconds)

This group parameter specifies the data refresh rate (in seconds) that all values of all items of this group connection will be refreshed at. If enabled, this setting tells the OPC Server to refresh all items in the group at the specified rate, which in turn will cause the OPC Server to resend **all items values** (via callback) at that rate. BridgeMaster in turn, if the Data direction is set correctly, may cause a write to other server.

For example, if the group is setup as data direction A->B and we want to ensure that B is always getting the latest value of A and is basically locked into whatever value A is say within a 30 second period even if sometimes the value of B itself changes apart from A, which would cause a temporary mismatch between A and B, then we would specify a refresh rate for the A side of 30 seconds. There is no point in specifying a refresh rate for the B side in this case so we would set it to 0 since for this example we do not need be writing back to A. This means every 30 seconds we read all values from A side for this group and then write all values to B side (if needed) which will ensure the B side has the correct A values (Note: *We only write to B if A value or quality has changed and is different from B).* Note: for groups with a lot of tags say 5K+, setting the refresh rate too low (or refreshing too fast) is not recommended; if that is the case then maybe a 15 minute refresh might be best (or a value of 15*60=900). It is important to keep in mind that although this periodic refresh may ensure that the values on both sides always match up, at least periodically, the OPC Foundation has gone thru great lengths to provide mechanisms to reduce network traffic and provide efficient data transfer by means of data callbacks and advise connections which fly in the face of periodically refreshing data which temporarily defeats the purpose of these efficient exception reporting mechanisms built into the OPC protocol. Therefore it is advised to use this setting sparingly and only if required in order to ensure values are the same in both systems. The minimum refresh value is 10 seconds. Specifying a value of 0 will disable periodic refreshes of the group.

Requested Interface Version [for Update Notification (Callbacks) and Writes]

The *Requested Interface Version* drop down box allows the user the ability to explicitly select which OPC Interface version to use for the group. This option allows 'old-style' v1.0 interfaces to be used as well as the new v3.0 interface. This option is provided for OPC Servers that do not support say the v2.0 or v3.0 Interfaces. By default this option is set to "Use Best Supported Interface". The following options are selectable:

- Use Best Supported Interface- This setting means BridgeMaster will query the server during startup for the best available supported OPC Interface and attempt to use that.
- Use OPC V1.a DA Interfaces-only OPC V1.a DA Interfaces will be used for data transfer of all items of this group on this server side.
- Use OPC V2.x DA Interfaces-only OPC V2.x DA Interfaces will be used for data transfer of all items of this group on this server side.
- Use OPC V3.0 DA Interfaces (ValueOnly Writes)- OPC V3.0 DA Interfaces will be used for data transfer of all items of this group on this server side. However writes to this server side will be like V2.x; that is Value-Only.
- Use OPC V3.0 DA Interfaces (with VQT Writes) - OPC V3.0 DA Interfaces will be used for data transfer of all items on this server side of this group. Writes to this server side will implement the new V3.0 VQT Write functionality whereas data is transferred with Values, Quality and timestamp. If the server does not support VQT writes, then BridgeMaster will log an error and degrade the actual interface version to *Using OPC V3.0 DA Interfaces (No VQT)*

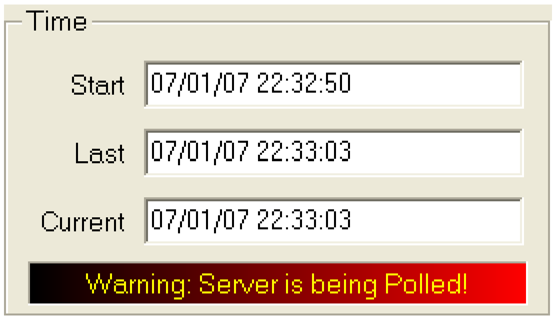
Actual Interface Version

The *Actual Interface Version* specifies to the user which OPC Interface version is actually being using for that group during runtime. If you request an interface that is not supported by the server, then this field will contain actual (lesser) interface version. The following options are selectable:

Do Polling with Server

The *Do Polling of Group* checkbox enables polling (synchronous read functions) of the OPC Server at regular intervals. If checked, then the Group's Update Rate specifies the polling interval. Polling the server implies that BridgeMaster will use the *IOPCSyncIO* interface (if either V1.a or v2.x interfaces were requested), and *IOPCSyncIO2* (if V3.0 interfaces was requested) to read all items from cache at the poll interval. This option is not as efficient since if v2.0 or v3.0 Interfaces are working correctly then there should be no need to read the data synchronously. This option is provided for OPC Servers that are setup incorrectly and/or does not support advise subscription mechanisms. OPC Advise is provided under v1.0 with the *IadviseSink* Interface and under v2.x and v3.0 as the *IOPCDataCallback* using a ConnectionPoint. Note: If this box becomes checked during runtime, then BridgeMaster found that the OPC Server did not support v1.0 Advise which may be due to a DCOM security issue or other problems with COM. See the Quick Setup section for details in that case.

The Server Status will show in colorful text the fact that the OPC Server is being polled as a warning to the user.



Start 07/01/07 22:32:50

Last 07/01/07 22:33:03

Current 07/01/07 22:33:03

Warning: Server is being Polled!

Use Synchronous Writes

The *Use Synchronous Writes* checkbox enables writing of (VQT or Value Only) data to be transferred (written to the server) using synchronous OPC Interfaces. Synchronous write calls wait for the OPC Server to actually write the value to the device for each item before returning success or failure, which holds up the interface for that period of time. For slow or otherwise unresponsive DCS systems, like polling, this could be a crippling bottleneck. Also since the synchronous calls may take too much time and cause timeouts with redundancy they are not efficient. If synchronous writes are selected, BridgeMaster will still utilize the Bulk Write functionality and will use the following interfaces to perform the writes:

- IOPCASyncIO interface (if either V1.a interface was requested)
- IOPCASyncIO2 interface (if v2.x interface was requested)
- IOPCASyncIO3 interface (if V3.0 interface was requested)

This option is provided for users who want to absolutely verify that writes go out to the OPC Server and are successfully written for any and all tags of the group on the designated server side. An error will be logged upon failure. However, it should be understood that with all asynchronous OPC versions, write status feedback is also provided back to BridgeMaster via the *IOPCDataCallback* or the *IAdviseSink* interface. However no check is made by BridgeMaster to verify that each and every write actually completed using asynchronous write mechanisms. If asynchronous is being used, and a write fails then normally a write completion event will be sent back to BridgeMaster and BridgeMaster will log the error. In a nutshell, asynchronous calls are the approved method since they are the fastest available means to write data, but since there is no guarantee that the write completion event will occur then there is no guarantee that the write actually did occur and if so if it was successful or not and this is the reason why this option is provided.

It is also noted that for a small group of tags that do not require frequent writes, the setting of this option should have minimal or no effect on the overall interface performance and would not be considered a problem. However, a bottleneck may occur using this option when too many tags (100’s to 1000’s) need to be written too frequently for the server to handle. Again this is all dependent upon the server response and the number of tag values per time that need to be written.

The Server Status dialog will show the OPC Statistics and performance for further diagnosis.

Note all of the group parameters defined above (with the exception of the Data Flow Direction and the Actual Interface Version) can be changed in runtime mode. The new settings can be applied immediately by selecting the Apply or OK button.

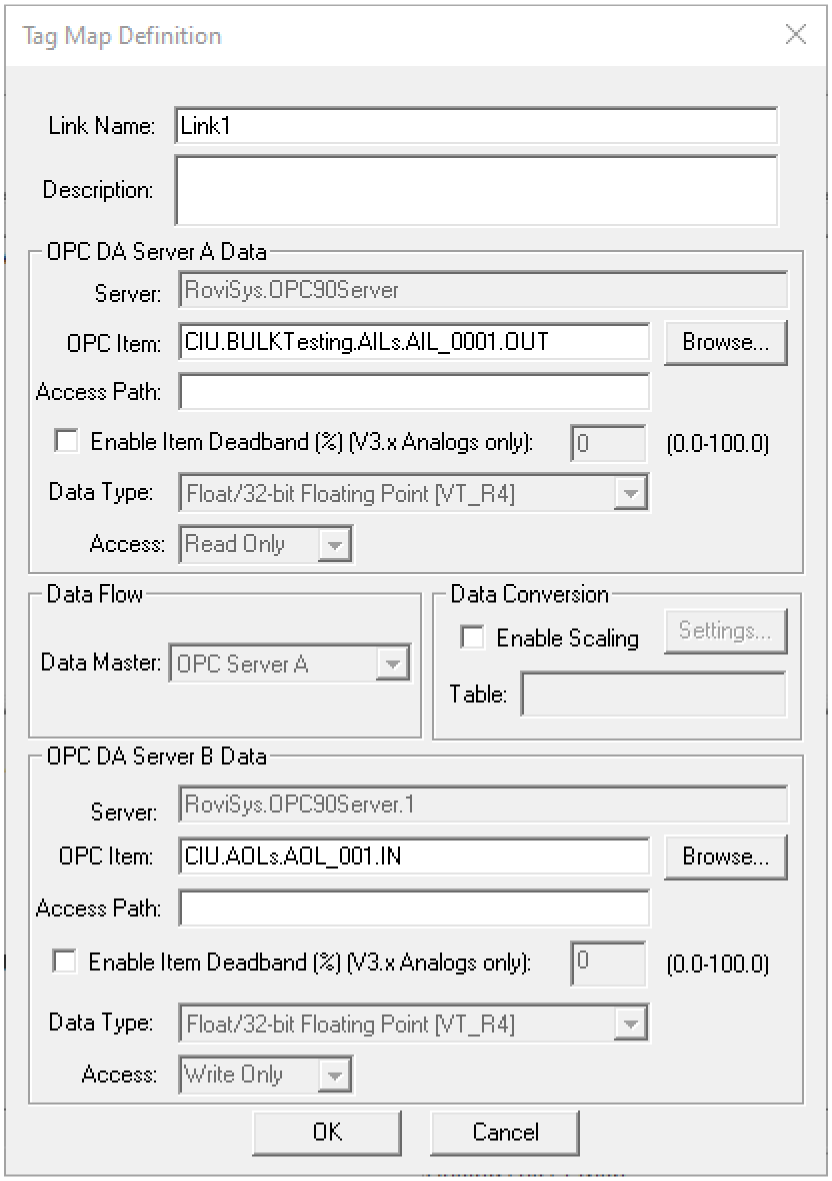
4.6 Tag Map Definition

All Tag Map definitions are associated with a parent BridgeMaster group, which are linked to a parent Connector object from which they are defined. This section provides the necessary information to define a Tag Map or Link correctly.

The Tag Map configuration is implemented as a collection of mappings of tags within OPC Server ‘A’ to/from tags within OPC Server ‘B’. This collection of mappings as stated above is stored as a Tag Map configuration. User configuration consists of simply mapping a tag from one server to the other. A tag within an OPC Server consists of its specific tag or Item ID representation. Specifically, for the RoviSys OPC90 Server it consists of a DEVICE, BLOCK and PARAMETER name, respectively.

4.7 Creating a New Tag Map

A new Tag Map (or Link) can be created by first selecting or highlighting the desired group for which to add the new tag map to and then selecting <*Edit* | *Add Tag Map*...> from the main menu or right mouse clicking on the group and then selecting “Add Single Tag Map”. The Tag Map Definition dialog box (shown below) will be displayed for the new tag map and can then be edited.



Link Name: Link1
Description:

OPC DA Server A Data
Server: RoviSys.OPC90Server
OPC Item: CIU.BULKTesting.AILs.AIL_0001.OUT
Access Path:
☐ Enable Item Deadband (%) (V3.x Analogs only): 0 (0.0-100.0)
Data Type: Float/32-bit Floating Point [VT_R4]
Access: Read Only

Data Flow
Data Master: OPC Server A

Data Conversion
☐ Enable Scaling
Table:

OPC DA Server B Data
Server: RoviSys.OPC90Server.1
OPC Item: CIU.AOLs.AOL_001.IN
Access Path:
☐ Enable Item Deadband (%) (V3.x Analogs only): 0 (0.0-100.0)
Data Type: Float/32-bit Floating Point [VT_R4]
Access: Write Only

Attributes are specified as part of defining the tag map in the above dialog box. The tag map parameters are defined below:

Link Name
Select a meaningful name for the Tag Map or Link or leave it at the default. This name must be unique amongst all groups in a given connector. Note this name can be changed in runtime by

single clicking on the name and simply changing it as you would be in changing a filename in Windows Explorer.

Description

This edit box allows a user-defined description to be entered for the tag map entry. Note this is may be left blank if desired.

Data Flow/ Data master

The Data Master Combo box allows the user to define the initial direction of data transfer or in other words where the data is originating from initially upon startup of runtime mode. This is only user-defined when the data flow of the parent BridgeMaster group is bi-directional as ←→. Otherwise, this is user-disabled and is for "display-only" since the parent group defines the data flow and thus the data master. The entries are "OPC Server A" or "OPC Server B". Note: upon startup of runtime mode for groups with bi-directional data flow both OPC Servers will send their current values of the tags producing two sources from which the data can originate. The Data Master entry eliminates this ambiguity by forcing the client tag to have only one source upon startup. This means the Data Master sends the first value, and if he fails then the other cannot write to him until the situation is cleared up.

Data Conversion

The Data Conversion/Scaling section allows the tag map to be scaled according to a Conversion Table defined from the *Settings*… button. This is discussed in the next section.

The attributes required for defining the specific Link or Tag Map data for each server are defined below:

Server

The Server edit box is used as a display-only reference and displays the defined OPC Server for this tag map.

OPC Item

The OPC Item edit box is used to enter the specific OPC Item ID or tag in the OPC Server. By selecting the *Browse…* button the user can browse the OPC Server database configuration and select the appropriate tag (assuming the OPC Server is browsable). If a tag is selected the OPC Item name will pop back into the tag map definition dialog box and be subsequently saved if the OK button is selected. Note: if the user wishes to only monitor an item from a server then it is not necessary to define the OPC Item for server B, simply leave that OPC Item field blank. In this case, the value of the undefined item will be displayed as <Invalid Tag> in Monitor mode. OPC items can be added, deleted and edited in runtime.

Access Path

The Access Path edit box is used to set the OPC Access path for the specific OPC Item/ tag. Note, in most cases this can be left blank, since most OPC Servers do not use this parameter. If this parameter is implemented within the OPC Server then consult the documentation for specific usage. For example, the access path may be COM1 or COM2.

Enable Item Deadband (%) (v3.x Analogs only)

This Link parameter allows the Item Deadband (in percentage of full scale range) to be set for the individual OPC Item within the group. For analog points (dwEUType of Analog) the EU Full

Scale range exists within the OPC Server as EU_HIGH - EU_LOW. If the OPC Server is v3.0 compliant then this parameter is applicable and can be used to determine when to notify a client of a change in value*. If enabled, this setting overrides that of the Group Deadband for this item.* This setting provides a mechanism to set the deadband on a “noisy” item within the group. This can be changed on the fly in runtime. If you disable this setting then the Item deadband will be cleared in the Server and will revert to the group deadband. Note: an error will be logged if the deadband is not supported by the server. Only Analog points are required to support the Item deadband. Warning: do not set this too high, since you may not ever receive a data change for the item if it doesn’t change by much. Range: 0-100%.

Data Type

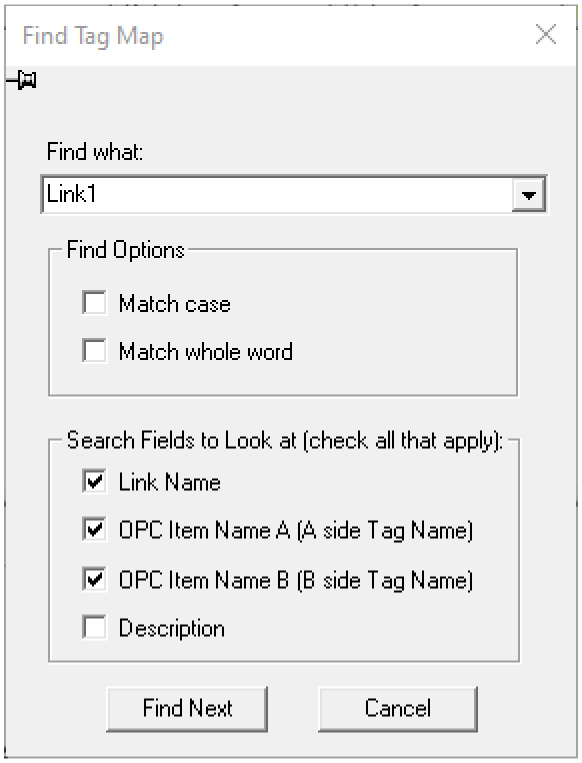
The Data Type combo box displays the default OPC tag type for the selected OPC Item. This is set to “native” initially upon tag map creation. In runtime, the actual OPC Item’s data type is sent from the OPC Server and will thereafter be subsequently displayed correctly. BridgeMaster provides automatic data type conversion between the two items; this includes support for Array types of any type to any other type. However Array data types cannot be transferred to non-array data types without using an Array Element Conversion table.

Access

The Access combo box displays how the OPC Item will be used in the exchange of data for the tag map. Note, this is for display-only since the parents group defines this from the data flow direction. A value of “Read Only” indicates that the OPC Item will be read only, no value can be written to it. A value of “Write Only” means the OPC Item will be written to (and not read). Exception reported values from the OPC Server are disregarded if it is Write Only. A value of “Read/Write” indicates the tag can be both read from and written to. In that case the Data Master entry defines who initially originates the value.

4.8 Finding a Tag Map

The Find Tag dialog box can be used to find text in various fields from any Tag Map. This dialog is displayed by first selecting the desired group or connector for which to start your tag search and then selecting <*Edit* | *Find Tag Map*...> from the main menu or by pressing the keys <Ctrl+F>. The Find Tag Map dialog box (shown below) will be displayed for the user to enter their search criteria into.



Find Tag Map

Find what: Link1

Find Options
- [ ] Match case
- [ ] Match whole word

Search Fields to Look at (check all that apply):
- [x] Link Name
- [x] OPC Item Name A (A side Tag Name)
- [x] OPC Item Name B (B side Tag Name)
- [ ] Description

Enter the Search Criteria fields as defined below and select the *Find Next* button or hit the F3 Key. To keep the dialog box open on subsequent searches check the pushpin in upper left hand corner of the dialog (as shown above).

Find What
Select and enter the string to search for.

Find Options
Select the desired find criteria options:
- Match case
- Match whole word

Search fields to Look at (check all that apply)
Select the desired search fields to look into:
- Link Name- searches for the desired text in the Link Name field of each link.
- OPC Item Name A - searches for the desired text in the Item Name field of side A.
- OPC Item Name B - searches for the desired text in the Item Name field of side B.
- Description –searches the description field of the Link.

4.9 Data Conversion/Scaling

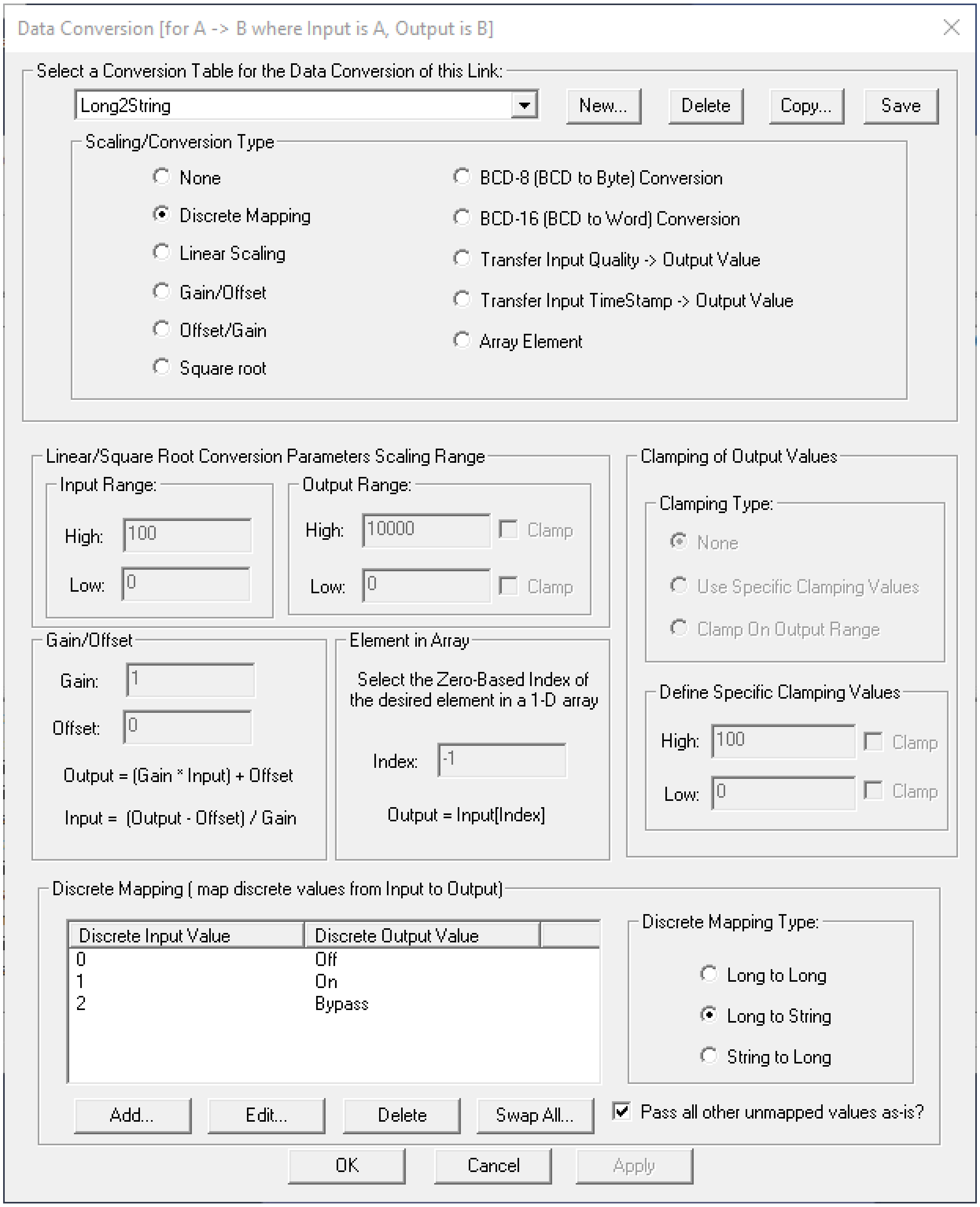
Data conversion tables allow tag values to be scaled or converted from one server to the other. The following data conversion/scaling types are now supported by BridgeMaster:

- Discrete Mapping
  - Long to Long
  - Long to String
  - String to Long
- Linear Scaling
- Gain/Offset
- Square root
- BCD (8 Bit) [BCD Byte conversion]
- BCD (16 bit) [BCD Word conversion]
- Transfer Input Quality to Output Value
- Transfer Input Timestamp to Output Value
- Array Element

4.10 Conversion Tables

To define scaling for a particular tag map, open the tag map dialog box and select *Enable Scaling* checkbox and then click the ***Settings***… button to view or modify the settings. Then select the desired conversion table, or click the ***New…*** button to create a new one. Warning: Conversion tables can be modified, added or deleted on the fly in runtime. As part of enabling scaling for a particular client tag mapping, the user must define and select a converter table along with the scaling type for the particular tag map to use in runtime to convert input values to output values between the two OPC Servers. Input and Output is relative to the parent group's data direction setting.

The user can define as many conversion tables as needed and then simply select the correct one for the given tag mapping. Multiple tag maps can use the same conversion table. Conversion tables are global to the address space within BridgeMaster meaning they can be used across connectors. As such, changes to a given conversion table in runtime may affect multiple tag maps. To define a conversion table, open the Tag Map Dialog box for the desired tag map and click the "*Settings*…" button. The following screen will appear:


Data Conversion [for A -> B where Input is A, Output is B]
Select a Conversion Table for the Data Conversion of this Link:
Long2String
New...
Delete
Copy...
Save
Scaling/Conversion Type
None
Discrete Mapping
Linear Scaling
Gain/Offset
Offset/Gain
Square root
BCD-8 (BCD to Byte) Conversion
BCD-16 (BCD to Word) Conversion
Transfer Input Quality -> Output Value
Transfer Input TimeStamp -> Output Value
Array Element
Linear/Square Root Conversion Parameters Scaling Range
Input Range:
High:
Low:
Output Range:
Clamp
Clamping of Output Values
Clamping Type:
Use Specific Clamping Values
Clamp On Output Range
Gain/Offset
Gain:
Offset:
Element in Array
Select the Zero-Based Index of the desired element in a 1-D array
Index:
Define Specific Clamping Values
Discrete Mapping (map discrete values from Input to Output)
Discrete Mapping Type:
Long to Long
Long to String
String to Long
Add...
Edit...
Swap All...
Pass all other unmapped values as-is?
OK
Cancel
Apply

Selected table: Long2String

Scaling/Conversion Type: [✓ Discrete Mapping]

Input Range: High: 100, Low: 0

Output Range: High: 10000, Low: 0

Clamping Type: [✓ None]

Gain: 1, Offset: 0

Output = (Gain * Input) + Offset

Input = (Output - Offset) / Gain

Index: -1

Output = Input[Index]

Define Specific Clamping Values: High: 100, Low: 0

Discrete Input Value	Discrete Output Value
0	Off
1	On
2	Bypass

Discrete Mapping Type: [✓ Long to String]

[✓] Pass all other unmapped values as-is?

To add a new conversion table, select the ***New…*** button and enter a descriptive name for the table and select the scaling type. To delete a Conversion table select the desired table to delete and click the ***Delete*** button. Note if any tag maps are referencing this table then it cannot be deleted. To copy an existing conversion table, select *Copy*… and enter the new name for the table. Enter a new descriptive name for the new table (spaces are OK, but no commas).

4.10.1 No Conversion

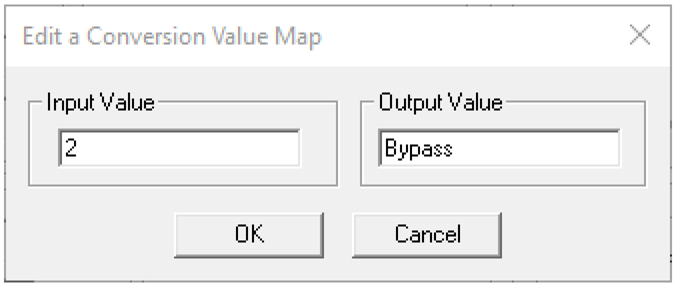
The first scaling type is actually not a scaling conversion type at all, it specifies No conversion. However, it is accommodative to the function of range limiting or clamping a value to a range without any conversion. For instance, if you just want to ensure a changing input value can only be written to an output in the range of 0 to 10 then select *None* for the Scaling/Conversion type and then select the *Use specific clamping values* radio button in the Clamping type section then simply enter the High and Low values for the desired range to limit the output value to. Select **OK** or **Apply** to save your edits. In runtime, your changes will take effect immediately for all tag maps using that Conversion table.

4.10.2 Discrete Mapping

To accommodate the conversion of discrete values from one system to another, tag map definition allows for the selection of discrete mapping. This is useful for instance to convert the discrete mode states of a PID or a Controller Block from one server to another. The following discrete mapping types are now supported:

- *Long to Long*
- *Long to String*
- *String to Long*

To add or edit a new conversion value map for the selected converter table, select the *Add…* button, the following dialog will appear:


Edit a Conversion Value Map

Input Value: 2

Output Value: Bypass

Enter the given discrete conversion values for either side for this conversion value map, repeat for all required discrete value maps. Note, one can also export the configuration and then edit a CSV file and then re-import to create multiple conversion values quickly. See the Import/Export section on how to accomplish this.

To delete a translation map, select the map and click *Delete.* To swap all values from 1 side to the other select the *Swap All…* button.

After the desired Discrete Mapping table and value maps are setup correctly, click OK and the tag map will be set to use this conversion table in runtime. Note, in runtime if a value is received for which there is no conversion map defined then the value is sent as-is unconverted to the other OPC Item defined within the given tag map. This default functionality can now be changed by de-selecting the "*Pass all unmapped values as-is?"* checkbox.

4.10.3 Linear Scaling

Most analog values exchanged between OPC Servers should already be expressed in engineering units and therefore, linear scaling of analog values should not be required. However if not then Linear scaling can be used by selecting the Linear Scaling conversion type. Enter the High and Low values for the Input range and the High and Low values for the Output range. The Output will be scaled according to the following equation:

$$\text{OUT} = (\text{IN} - IN_{LO})\,\frac{(OUT_{HI} - OUT_{LO})}{(IN_{HI} - IN_{LO})} + OUT_{LO}$$

If the group for which the tag map belongs is bi-directional then the data feeding back from the non-Data Master (output) side to the Data Master (input) side is scaled according to:

$$\text{IN} = (\text{OUT} - OUT_{LO})\,\frac{(\,IN_{HI} - IN_{LO})}{(OUT_{HI} - OUT_{LO})} + IN_{LO}$$

Important: if you don't want feedback, then don't use a bi-directional group. Furthermore, if clamping is desired, select one of the desired Clamping Types:

- None
- Use Specific Clamping Values
- Clamp on Output Range.

Then enter the specific clamping values and make sure to use the checkbox next to each clamp. It is also possible to only clamp the LO or only clamp the HI by selecting the desired clamping checkbox.

4.10.4 Gain/Offset

The Gain/Offset scaling type provides the ability to multiply or divide a value according to a desired gain and/or to provide a constant bias via a desired offset. Enter the Gain and Offset values to be used for the conversion. The Output will be scaling according to the following equation:

$$\text{OUT} = (\text{GAIN} * \text{IN}) + \text{OFFSET}$$

If the group for which the tag map belongs is bi-directional then the data feeding back from the non-Data Master (output) side to the Data Master (input) side is scaled according to:

$$\text{IN} = \frac{OUT - OFFSET}{GAIN}$$

Furthermore, if clamping is desired, select one of the desired Clamping Types:

- None
- Use Specific Clamping Values

Then enter the specific clamping values and make sure to use the checkbox next to each clamp. It is also possible to only clamp the LO or only clamp the HI by selecting the desired clamping checkbox.

4.10.5 Square root

The Square root scaling type provides the ability to scale according to a square root function. Enter the High and Low values for the Input range and the High and Low values for the Output range. The Output will be scaled according to the following equation:

$$\text{OUT} = \sqrt{IN - IN_{LO}}\ \frac{(OUT_{HI} - OUT_{LO})}{\sqrt{(IN_{HI} - IN_{LO})}} + OUT_{LO}$$

If the group for which the tag map belongs is bi-directional then the data feeding back from the non-Data Master (output) side to the Data Master (input) side is scaled according to:

$$\text{IN} = (OUT - OUT_{LO})^2\ \frac{(IN_{HI} - IN_{LO})}{(OUT_{HI} - OUT_{LO})^2} + IN_{LO}$$

Important: if you don't want feedback, then don't use a bi-directional group. Furthermore, if clamping is desired, select one of the desired Clamping Types:

- None
- Use Specific Clamping Values
- Clamp on Output Range.

Then enter the specific clamping values and make sure to use the checkbox next to each clamp. It is also possible to only clamp the LO or only clamp the HI by selecting the desired clamping checkbox.

4.10.6 BCD-8 (BCD to Byte Conversion)

BCD-8 (BCD to Byte) conversion is supported to scale according to a Binary Coded Decimal to BYTE function. This type specifies conversion of a BCD in the 1st byte (8 bits) of an integer from the Input side to an 8 bit BYTE on the Output side. The BCD data is read from the input as raw BCD values in hex format 0x00 to 0x99. The range of Engineering Unit values written to the output side is 0-99. If the output data type is VT_R8 or VT_R4 (floating point) then the value will range from 0.0 – 99.0 if the output data type is a string (VT_BSTR) then it will be converted to a string value "0" to "99". Input values outside this range will be clamped to 0-99 and an error DISP_E_OVERFLOW (0x8002000A) may occur.

If the group from which the tag map using this conversion table belongs is a bi-directional group then the opposite calculation will occur when the output changes, namely taking a WORD value in the range 0-99 from the output side and converting to a BCD value in the range of 0x00 to 0x99 (0 to 153) in the input side.

4.10.7 BCD-16 (BCD to Word Conversion)

BCD-16 (BCD to Word) conversion is supported to scale according to a Binary Coded Decimal to WORD function. This type specifies conversion of a 16 bit BCD as an unsigned integer (or the 1st WORD) from the Input side to a 16 bit WORD on the Output side. For analog Input data types the raw BCD word is read from the input tag in hex format as 0x0000 to 0x9999 and then converted to a 16 bit unsigned integer (0-9999) and the WORD result is sent back to the tag of the output side. The range of Engineering Unit values written to the output side is 0-9999. If the output data type is VT_R8 or VT_R4 (floating point) then the value will range from 0.0 – 9999.0 if the output data type is a string (VT_BSTR) then it will be converted to a string value “0” to “9999”. Input values outside this range will be clamped to 0-9999 and an error DISP_E_OVERFLOW (0x8002000A) may occur.

If the group from which the tag map using this conversion table belongs is a bi-directional group then the opposite calculation will occur when the output changes, namely taking a WORD value in the range 0-9999 in the output side and converting to a BCD value in the range of 0x0000 to 0x9999 (0 to 39321) in the input side.

4.10.8 Transfer Input Quality -> Output Value

This conversion allows the OPC Quality value of the tag on the input side to be transferred to the Output value. The OPC Quality is a 16-bit WORD value containing limit bits, status bits and sub-status bits. This value is transferred directly to the output value and coerced to whatever data type the output tag is. Multi-directional value transfer is not supported, if Bi-Directional group is setup then only the side designated as the Data master will source the Input Quality to the value on the other side. (This is due to the fact that V1/V2 OPC Servers do not allow writing to the OPC Quality)

4.10.9 Transfer Input Timestamp -> Output Value

This conversion allows the OPC Timestamp value of the input side to be transferred to the value of the output side. The OPC Timestamp is a 64-bit or (2) DWORD value containing the timestamp from the server of the tag. The FILETIME value is converted to local file time then the local file time is converted from UTC to system time. This value is then transferred directly to the output value as a VT_DATE type. This value may be coerced to whatever data type the output tag is, assuming the data type is large enough to hold the value. Multi-directional value transfer is not supported, if Bi-Directional group is setup then only the side designated as the Data master will source the Input Timestamp to the Value on the other side. (This is due to the fact that V1/V2 OPC Servers do not allow writing to the OPC Timestamp)!

4.10.10 Array Element

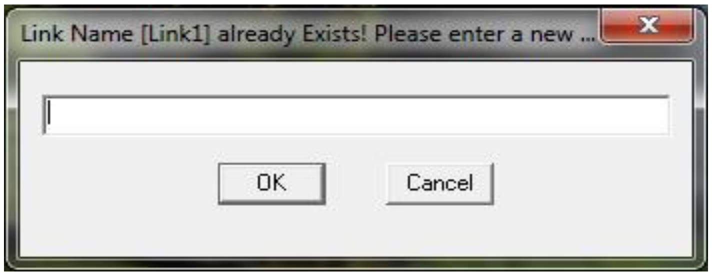
This conversion type is useful for picking out a particular element in an input array to read and then send its value to an output tag. This is useful for cases where a server does not allow individual access to a given element in an array, and only allows reading the entire array. **If the server does support reading individual elements in an array then this conversion is not required.** Otherwise, select the zero (0) based index of the desired element in a 1-D array. Currently BridgeMaster only supports 1-Dimensional arrays. Make sure the OPC Item of the array is addressed as the entire array like DEV1.GRP1.ARRAY and not an element like DEV1.GRP1.ARRAY[3]. Multi-directional value transfer is not supported, in other words you cannot write back to a particular index in an array because the only way to do that would be to write the entire array and that presupposes (or assumes) you have the latest value of each and every other element in the array at the time of the write which may or may not be the case. Also note that no error will occur if the actual array size is less than the index you are trying to index. For instance if you have an array of size 4 (indexes 0 to 3) and you try to address the 5th element, this is not considered an error even though the 5th element does not exist. BridgeMaster assumes the array will eventually be sized accordingly and will resolve the issue.

4.11 Editing a Connector, Group or Tag Map

To edit a Connector, Group or Tag Map simply double-click on the object. Alternatively, you can select the object and then select < *Edit* **|** *Properties*…> from the main menu or right mouse click on the object and select "Properties…" from the drop-down menu. Most object properties can be edited in runtime. The ability to change any OPC Item reference in runtime is also supported.

4.12 Copying/Moving Tag Maps between Groups

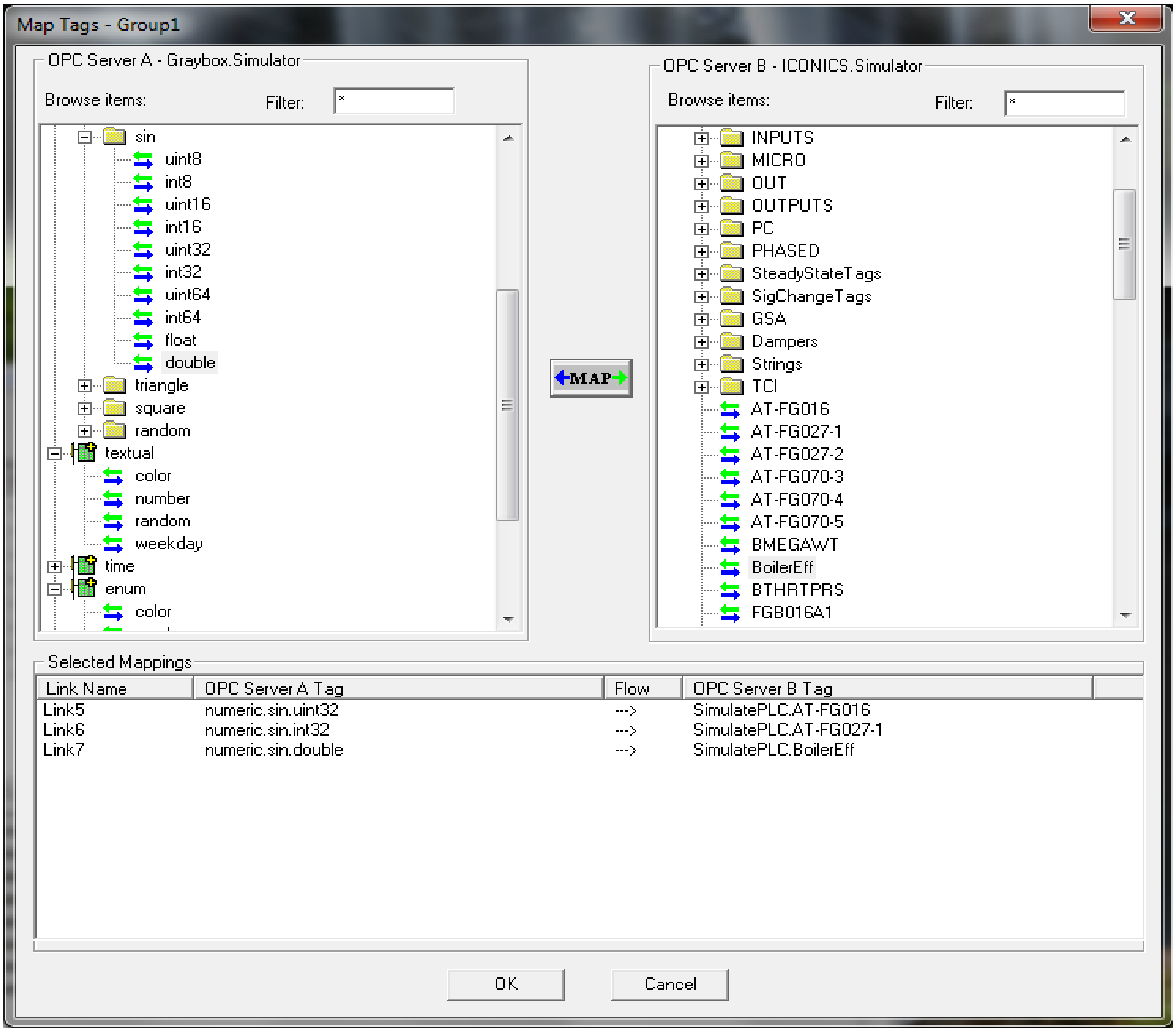
In Configure mode, one or more Tag Maps can be copied or moved from one group to another. Simply select (highlight) the desired Tag Map(s) to copy or move and drag them to the desired group. Multiple Tag Maps can be selected at a time by holding down the Control or shift keys as you would do to drag and drop files. Note if you hold down the <Control> key before you drop the object(s), then they will be copied to the desired group. Otherwise the default operation is to move them from the group they were dragged from. This will first cause them to be deleted from the source group. If you hold the control key down then you are copying the Links in which case since Link Names need to be unique among all links in a given Connector you may be prompted with the following dialog box to rename them if you are copying within the same connector:


Link Name [Link1] already Exists! Please enter a new ...

Simply type in a new unique Link Name for each item that was copied.

4.13 Multiple Tag Map Configuration

Multiple tags can be mapped quickly and effortlessly using the "Map Tags" dialog box. This is accomplished by first selecting or highlighting the desired group for which to add the new tag map(s) to and then selecting <Edit **|** Add Tag Map(s)...> from the main menu or pressing the keys <Ctrl+T> or right mouse clicking on the group and then selecting "Add Tag Map(s)…". The Map Tags dialog box (shown below) will be displayed.



Link Name	OPC Server A Tag	Flow	OPC Server B Tag
Link5	numeric.sin.uint32	--->	SimulatePLC.AT-FG016
Link6	numeric.sin.int32	--->	SimulatePLC.AT-FG027-1
Link7	numeric.sin.double	--->	SimulatePLC.BoilerEff

This dialog box displays a list view tree control on either side of its dialog box (assuming that each server on either side can be browsed). Each list box pane will contain a tree view listing of all available OPC items within each server. The user simply selects an OPC item on one side and the corresponding item to map to on the other side and clicks on the “←MAP→” button. Alternatively, the user can select an item from one side and simply drag it into an OPC Item on the other side to create a tag mapping. The data flow direction is decided by the direction of the parent group. Also, the currently mapped tags of the current dialog box session are displayed in a lower list box along with an arrow indicating data direction. Selecting *cancel* will cancel all selected mappings for the current session. To remove a tag mapping from a given edit session simply select the desired tag map to delete in the “Selected Mappings” section and press the delete button. Selecting the OK button on the “Map Tags” dialog box causes the program to accept the new mappings (if any) and load in the selected tag mappings under the selected Group. The user can then refine or edit the mappings by double clicking on any new tag map to bring up the Tag Map Definition for each individual mapping. Tag Maps can be added in runtime mode.

The Link Names can be changed before selecting the OK button to add them to the group. To change a Link Name simply single click on the name you would like to change and enter the new name as you would change the name of filename in Windows Explorer.

5 BridgeMaster Runtime Operation

In runtime, BridgeMaster acts as a two-headed OPC Client for each connector; that is it acts as an OPC Client to each configured OPC Server for each enabled connector. Upon startup of runtime mode, for each enabled connector a connection with both defined OPC Servers and all defined BridgeMaster groups are created within the OPC Servers. All OPC items defined within the tag maps of these groups are added to their corresponding OPC groups. Most OPC Servers support advise subscriptions, whereby when data changes within the server, it is sent immediately to BridgeMaster via a mechanism called a Connection Point for a particular group. This rate is specified from the Update Rate in the group dialog box. Data is transferred from server to server based on the settings set up within the group and converted based on data type and data conversion settings.

The OPC Groups and Items of Connectors that are not enabled will not be connected to in runtime. The Enable checkbox within the Connector Dialog box sets this property.

5.1 Monitoring Values

In runtime, OPC Item values can be monitored by selecting *View* | *Monitor* from the main menu. Current values will appear under the *Value A* and *Value B* columns in the tag view pane. Note, if the server or group is not connected (or is disabled) then the values will be displayed as "? (Bad – Waiting for Initial Data)". Monitored values are refreshed every 100 milliseconds. Select another group to view tag map values under that group. Note, if the data flow setup in the parents group does not allow reading of the OPC Item then the values may not be displayed correctly since they may not have been "read" from the OPC Server.

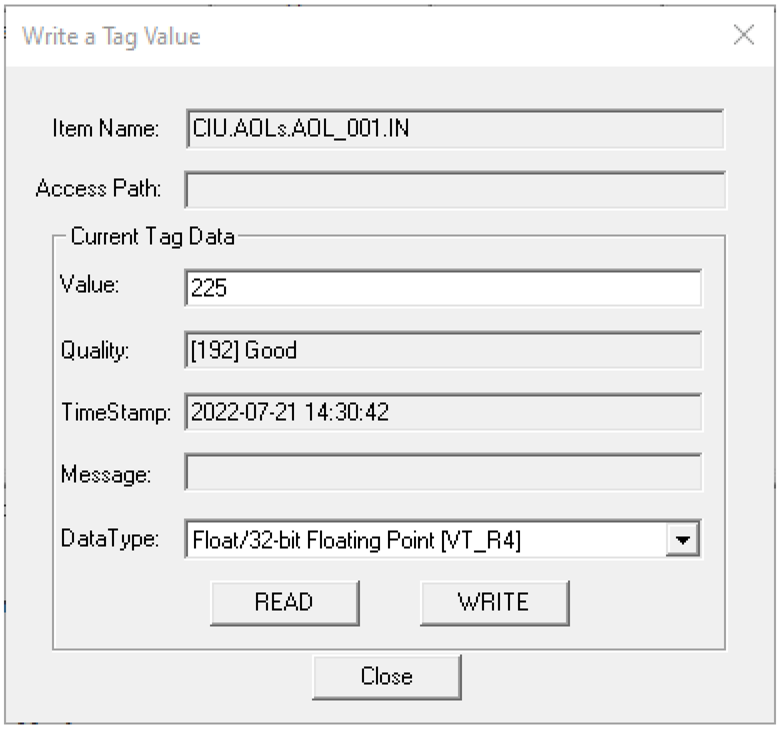
Note, if the tagmap is displayed with a ⚠ symbol then one or both of the OPC Items defined within that tagmap are invalid (or possibly unconfigured). This means that the OPC Item does not exist within the server or is not configured in BridgeMaster or has a blank item name.

5.2 Reading OPC Values

In runtime, OPC Item values can be manually read by forcing an OPC read. This is accomplished by selecting the desired tag map's *Link* column in the tag view pane and then right mouse clicking. Select "*Read>Item A or Item B*" from the corresponding pop-up menu.

5.3 Writing OPC Values

In runtime, a value can be manually written to an OPC Item by forcing an OPC write. This is accomplished by selecting the desired tag map's *Link* column in the tag view pane and then right mouse clicking to bring up the tags menu. Select "Write> Item A or Item B" from the corresponding pop-up menu. This will activate the following dialog box.


Write a Tag Value

Item Name: CIU.AOLs.AOL_001.IN

Access Path:

Current Tag Data

Value: 225

Quality: [192] Good

TimeStamp: 2022-07-21 14:30:42

Message:

DataType: Float/32-bit Floating Point [VT_R4]

Enter the value you wish to write in the *Value* edit box and select the *WRITE* button to force an OPC write to the specified OPC Item. If you wish to read the current value back to verify it was written select the *READ* button. Note, if the value's quality is BAD then this state will be displayed in the *Quality* edit box prefixed by the actual quality value in brackets []. Note, if the data flow setup in the parents group does not allow writing to this OPC Item then the *WRITE* button will be disabled.

Note, if the Current Value and/or Current Quality is displayed with '?', question marks, then the OPC Item (tag) is possibly not connected due to an invalid tag name or a disconnected OPC Server.

The data type written to the server (as converted from the edit box) will be that of the OPC Server Item. You can attempt to coerce the write using a different data type, but there is no guarantee that the server will accept it. An error will logged to log file if the *Show Errors* flag is set that connector.

For Array data types enter square brackets around the whole array with commas separating the individual elements in the array you wish to write. For instance for an array of floats:

[14642.3, 13714.7, 9714.99, 3193.85, 2969.21, 13639.8, 7932.66, 2596.34, 8410.97]

An example of an array of strings has " " surrounding the individual strings:
["TestString1", "TestString2"]

The Write Tag dialog box is also useful to determine the possible cause of a problem with an OPC Item. If the tag's quality is [0] bad, then the sub-status is displayed in the Message edit box (assuming the OPC Server has set the sub-status).

5.4 Redundant OPC Servers / Fail over

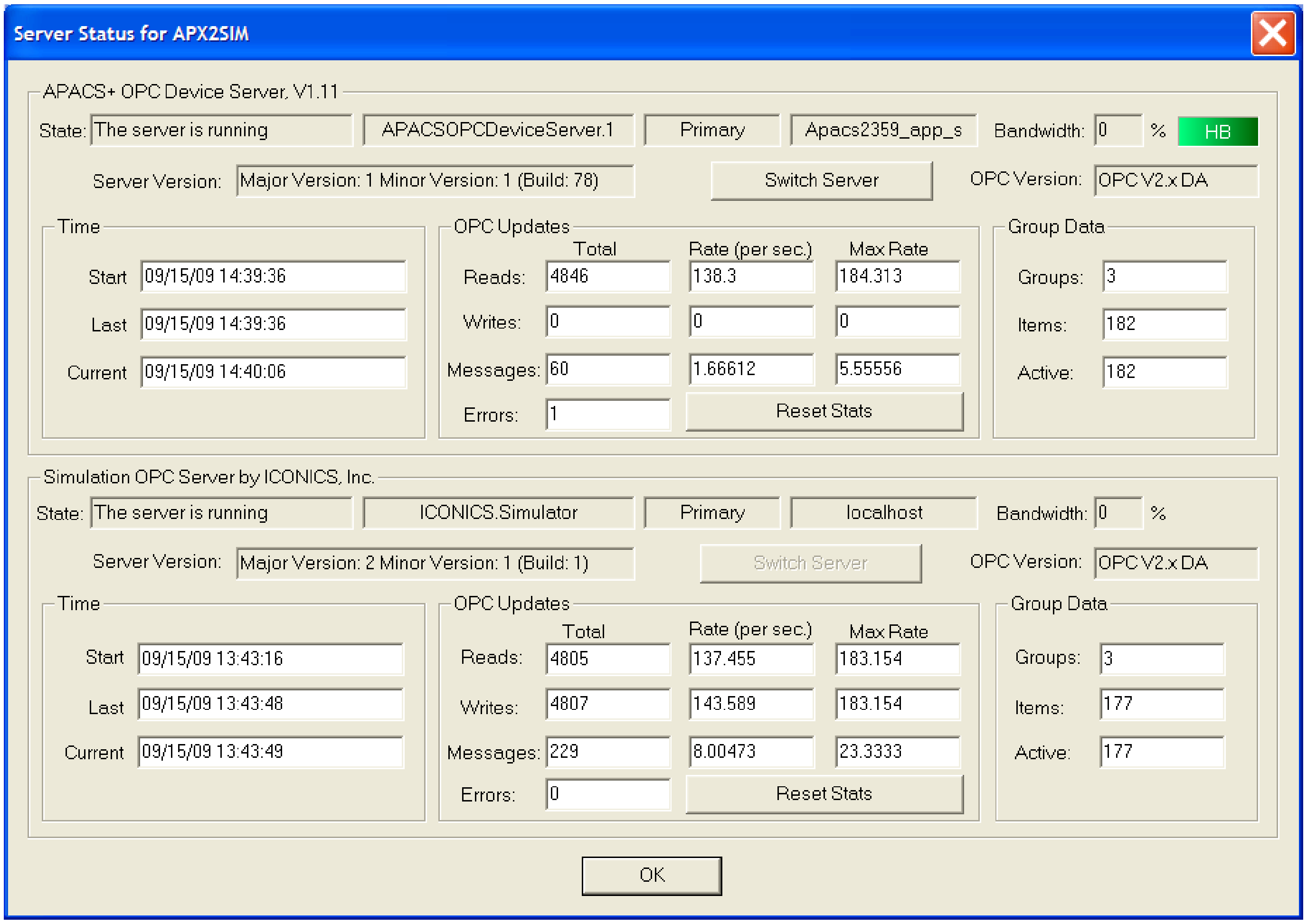
BridgeMaster supports server redundancy of OPC Servers running on multiple networked hosts. The Primary and Secondary hosts for each connector side are configured in the Connector dialog box. Upon startup of runtime mode, for each enabled connector a connection is first attempted with the primary host and if that fails then a connection is attempted with the redundant secondary server node.

In runtime mode, if communication to a connected OPC Server fails for one or more of the user-specified failover reasons then BridgeMaster will implement a “cold fail-over”. During a cold fail-over BridgeMaster basically disconnects all OPC links with the “failed” server and attempts a switch over to the redundant one. This switchover or fail-over consists of verifying that the redundant server is configured and if so it then pings the remote node to verify the node exists. If the remote node exists then a connection attempt is made with the specified OPC Server on that node. If that operation succeeds, BridgeMaster then establishes all groups and all tag maps within each group and then has undergone a switchover. If it does not succeed, it will reattempt the connection the number of times specified in the “*Number of Retries*” in the options dialog box. After the number of retries has been exceeded it then attempts a fail over to the first server/node. In runtime, this connection attempt/ fail over process will last indefinitely until a successful connection has been established. Messages for each failed attempt will be logged to the message pane as well as to the error log (if configured).

In order for Server redundancy to work, each redundant OPC Server pair must contain the same OPC Item names on either redundant node. In essence, if an OPC Item called “DEV1.GRP2.SP1” exists within the OPC Server configuration in the primary OPC server on the local node, then the OPC Item called “DEV1.GRP2.SP1” must also exist within the configuration of the secondary OPC server on the remote node in order for redundancy to work properly if and when a switchover occurs. Otherwise, adding missing OPC Items to groups during a switchover will fail with “Invalid OPC Item” error messages. The server status will still indicate running, but one or more OPC items may not have a connection.

5.5 Server Status

As a runtime option, a user can view the runtime status of each OPC Server within a given connector along with the OPC Statistics for a given connector. Select the connector to view the status of and then select <*Utility* | *Server Status…*> from the main menu or right mouse click on a desired Connector or group and then click the <*Server Status*> option. The following dialog box will be displayed.


Server Status for APX2SIM

APACS+ OPC Device Server, V1.11

State: The server is running | APACSOPCDeviceServer.1 | Primary | Apacs2359_app_s | Bandwidth: 0 % | HB

Server Version: Major Version: 1 Minor Version: 1 (Build: 78) | Switch Server | OPC Version: OPC V2.x DA

Time

Start	09/15/09 14:39:36
Last	09/15/09 14:39:36
Current	09/15/09 14:40:06

OPC Updates

	Total	Rate (per sec.)	Max Rate
Reads:	4846	138.3	184.313
Writes:	0	0	0
Messages:	60	1.66612	5.55556
Errors:	1	Reset Stats	

Group Data

Groups:	3
Items:	182
Active:	182

Simulation OPC Server by ICONICS, Inc.

State: The server is running | ICONICS.Simulator | Primary | localhost | Bandwidth: 0 %

Server Version: Major Version: 2 Minor Version: 1 (Build: 1) | Switch Server | OPC Version: OPC V2.x DA

Time

Start	09/15/09 13:43:16
Last	09/15/09 13:43:48
Current	09/15/09 13:43:49

OPC Updates

	Total	Rate (per sec.)	Max Rate
Reads:	4805	137.455	183.154
Writes:	4807	143.589	183.154
Messages:	229	8.00473	23.3333
Errors:	0	Reset Stats	

Group Data

Groups:	3
Items:	177
Active:	177

Note if redundancy is enabled, this dialog box displays which node is currently active (Primary or Secondary) and can be used to cause a switchover of an OPC Server to the redundant one by simply clicking the “Switch Server” button. If the redundant node is defined then BridgeMaster will attempt a connection to the redundant one. A connection attempt to this server will be made however many times defined in the “Number of Retries” edit box in the Options dialog box before reattempting a connection with the first server node. Also, if the watchdog tag is defined then the changing value will make the “HB” edit box blink as gradient green in the upper right hand corner. Also displayed is the highest OPC Version used by any of the groups of that connector side.

5.6 Verifying OPC Tags

In runtime, a user can verify the status of all OPC Items within all tag maps of all Active Groups in the currently running configuration. Select <*Utility* | *Verify OPC Tags*> from the main menu to check all OPC Items. If an error was found with one or more items then a message will be displayed in the message pane of the main window with the last error. Otherwise, the following messages will be displayed indicating successful connection with these items.

```
MSG :: Verifying OPC Tags in Server A...
MSG :: OPC Tags Verified!
MSG :: Verifying OPC Tags in Server B...
MSG :: OPC Tags Verified
```

5.7 Validate all Connectors

The utility function is useful to validate your configuration including validating all connectors, groups, tag maps and Conversion tables in the current configuration. Select <*Utility* | *Validate All Connectors*> from the main menu to validate all connectors. If an error or warning was found with one or more connectors then a message will be displayed in the message pane of the main window with the last error. Otherwise, the following messages will be displayed indicating successful connection with these items.

```
EVENT:   Connector1>Validating Connector...
EVENT:   System>Validated Connector <Connector1>, 0 errors 0 warnings found!
EVENT:   System>Validating Translation Tables...
EVENT:   System>Validated all active Connectors and Translation Tables: 0 errors 0 warnings found!
```

5.8 Boolean Data Type Value Conversions (VT_BOOL)

In runtime, OPC Items of data type VT_BOOL may be transferred to items of any other variant types. The OPC Foundation has put together rules regarding these conversions which BridgeMaster now follows, which may be different from early versions. In the new version when an item is of type VT_BOOL and when the value is True then it will send -1 to signed integer items and will send the highest supported integer based on number of bits in that integer for unsigned integer items.

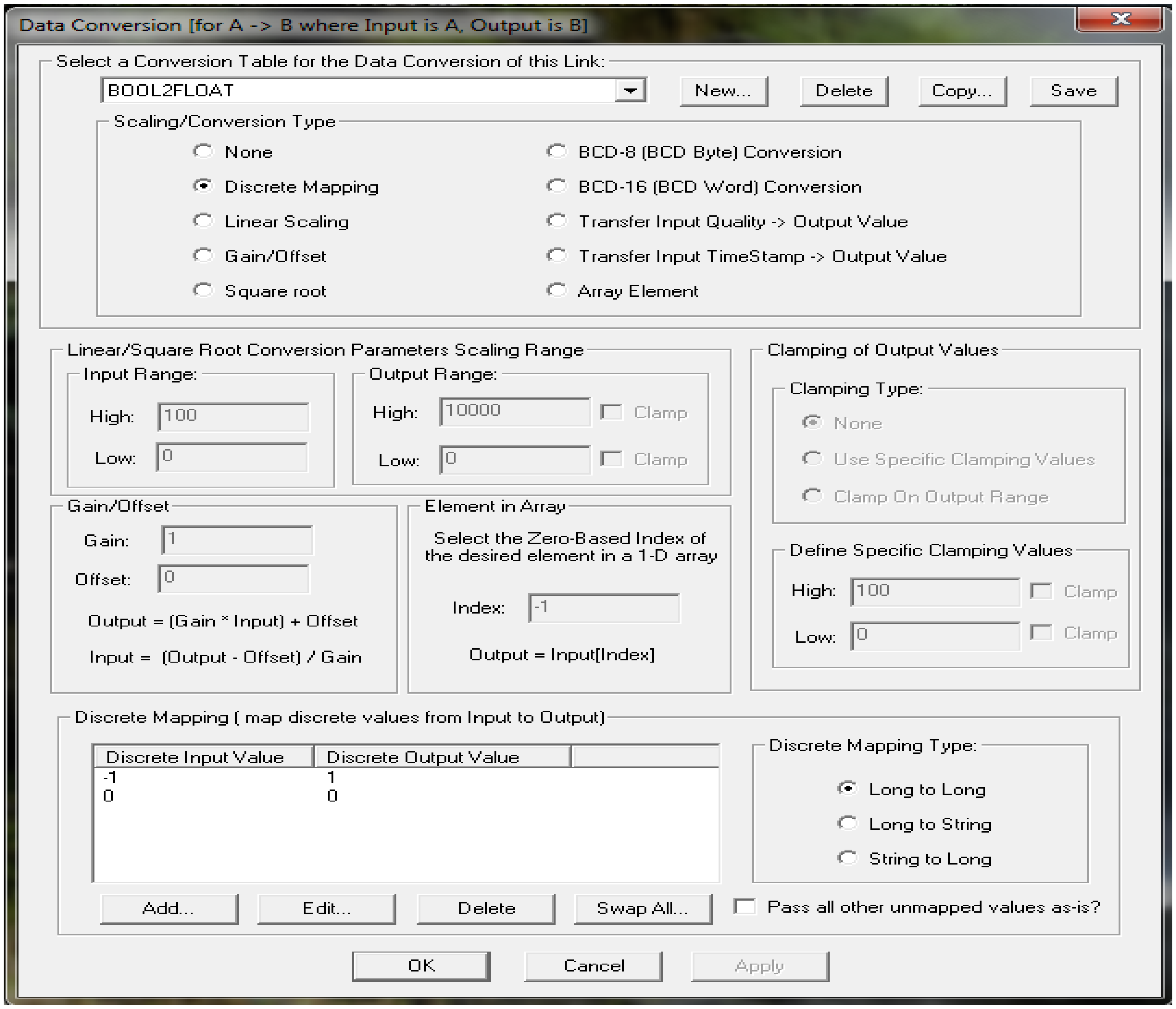
An Item that is of variant type VT_BOOL (Logical Boolean) will convert as follows for the following 'other side' data types:

<u>Conversion VT_BOOL to VT_XX</u>

ITEM A (VT_BOOL) → ITEM B (VT_XX)

Item B Type	If Item A is True then write to Item B side value as	If Item A is False then write to Item B Side 'False' Value as:
VT_BOOL	True	False
VT_UI1	255 (0xFF)	0
VT_I1	-1	0
VT_I2	-1	0
VT_UI2	65535 (0xFFFF)	0
VT_I4	-1	0
VT_UI4	4294967295 (0xFFFFFFFF)	0
VT_R4	-1.0	0
VT_R8	-1.0	0
VT_BSTR	"-1"	0

It should further be noted that within BridgeMaster a VT_BOOL variant TRUE is normally stored as -1. For instance, writing a VT_ BOOL 'True' value from the Write Tag dialog will always send a True {-1} VT_BOOL value. However some (most) servers send back the value of 1 over as VT_BOOL True, so when transferring this to the other tag map side it goes over as 1 not -1, BridgeMaster is simply passing the data along in that case and not explicitly enforcing the rules or filtering those values.
If your application requires a different result from the standard specified above then a user can simply configure a Data Conversion table of Type "Discrete Mapping" and then add 2 maps to compensate for this mandated behavior. For instance, if a VT_BOOL item is on side A of a given tagmap and a VT_R4 item is on Side B and you want the value of "1.00" to be sent to side B (instead of "-1.00") when the Boolean on side A goes 'True' then the following conversion table called "BOOL2FLOAT" could be used for those tag maps:



Select a Conversion Table for the Data Conversion of this Link: BOOL2FLOAT

Scaling/Conversion Type: None [✓ Discrete Mapping] Linear Scaling Gain/Offset Square root BCD-8 (BCD Byte) Conversion BCD-16 (BCD Word) Conversion Transfer Input Quality -> Output Value Transfer Input TimeStamp -> Output Value Array Element

Linear/Square Root Conversion Parameters Scaling Range

Input Range: High: 100 Low: 0

Output Range: High: 10000 Clamp Low: 0 Clamp

Clamping of Output Values

Clamping Type: [✓ None] Use Specific Clamping Values Clamp On Output Range

Define Specific Clamping Values: High: 100 Clamp Low: 0 Clamp

Gain/Offset: Gain: 1 Offset: 0

Output = (Gain * Input) + Offset

Input = (Output - Offset) / Gain

Element in Array: Select the Zero-Based Index of the desired element in a 1-D array

Index: -1

Output = Input[Index]

Discrete Mapping (map discrete values from Input to Output)

Discrete Input Value	Discrete Output Value
-1	1
0	0

Discrete Mapping Type: [✓ Long to Long] Long to String String to Long

Pass all other unmapped values as-is?

Of course the other data flow direction is simple, basically going from VT_XX to VT_BOOL says if the value of VT_XX is NOT 0 then it sends True, and if its 0 then it sends FALSE.

<u>Conversion VT_XX to VT_BOOL</u>

ITEM A (VT_XX) → ITEM B (VT_BOOL)

Item A Type	**In order to write ‘True’ to VT_BOOL Item B Side then Item A needs to be this:**	**In order to write ‘False to VT_BOOL Item B Side then Item A needs to be this:**
VT_BOOL	True	False
VT_UI1	Not 0	0
VT_I1	Not 0	0
VT_I2	Not 0	0
VT_UI2	Not 0	0
VT_I4	Not 0	0
VT_UI4	Not 0	0
VT_R4	Not 0	0
VT_R8	Not 0	0
VT_BSTR	Not 0	0

5.9 Array Support

In runtime, OPC Items of data type VT_ARRAY may be transferred to items of the same or slightly different data types. If Item A is an array to be transferred to Item B which is also an array, the data type of array A will be coerced into the data type of array B. if an error occurs it will be logged assuming logging is enabled. The entire array, meaning each element, will be transferred during a write operation. Using arrays can dramatically increase the performance by providing 1000's of data points as a single OPC Item. The following VT ARRAY data types are supported:

Important: Not all OPC Servers support VT_ARRAY data types. Please check your OPC Server software users manual to see if VT_ARRAYs are supported.

<u>Supported VT_ARRAY types (single dimension SAFEARRAY)</u>

Array Type	**Description**
VT_ARRAY | VT_BOOL	Array of BOOLEAN values
VT_ARRAY | VT_UI1	Array of BYTE values (8 bit values)
VT_ARRAY | VT_I1	Array of CHARACTER values (8 bit values)
VT_ARRAY | VT_UI2	Array of UNSIGNED INTEGER values (16 bit values)
VT_ARRAY | VT_I2	Array of SIGNED INTEGER values (16 bit values)
VT_ARRAY | VT_UI4 or VT_ARRAY | VT_UINT	Array of UNSIGNED LONG values (32 bit values)
VT_ARRAY | VT_I4 or VT_ARRAY | VT_INT	Array of SIGNED LONG values (32 bit values)
VT_ARRAY | VT_UI8	Array of UNSIGNED LONGLONG values (64 bit values)
VT_ARRAY | VT_I8	Array of SIGNED LONGLONG values (64 bit values)
VT_ARRAY | VT_R4	Array of FLOAT values (32 bit single precision values)
VT_ARRAY | VT_R8	Array of DOUBLE values (64 bit double precision values)
VT_ARRAY | VT_DATE	Array of DATE values (DATE values)
VT_ARRAY | VT_STRING	Array of STRING values (BSTR values)
VT_ARRAY | VT_CY	Array of CURRENCY values
VT_ARRAY | VT_VARIANT	Array of VARIANT values

5.10 Ping

As a runtime utility, the ping function is supported to easily verify access to a remote node without shelling out to a DOS prompt. To ping a remote node select <Utility | Ping...> from the main menu (or simply click the ping shortcut in the toolbar ●). In the Ping dialog box enter the host name or IP Address of the host machine to ping to and click OK. The status will be displayed the message pane in the main driver window. If successful the message will display a string "*Ping>Host <localhost> is alive!*" where *localhost* was used as the host in this example, otherwise another message will be displayed indicating that the host is unreachable or is not responding due to a timeout or a network connection problem. This is useful for testing initial configurations and debugging problems.

5.11 Launch Service Manager

As a utility function, the Service manager program can be launched from the main menu by selecting *Utility* **|** *Launch Service Manager* menu. This utility is used to setup and configure security settings for local services such as the *BridgeMasterService*.

5.12 Launch DCOM Configuration

As a utility function, the DCOM Configuration program can be launched from the main menu by selecting Utility **|** Launch DCOM Configuration menu (or simply click the shortcut in the toolbar). This utility is used to setup and configure security settings for local OPC Servers.

5.13 IOPCShutdown

BridgeMaster supports the *IOPCShutdown* interface and the Shutdown Request functionality from OPC Servers. If supported by the OPC Server, when the server needs to shutdown, it can use this interface to send a shutdown request to BridgeMaster. BridgeMaster will then disconnect all items, groups and interfaces connected with that server and will wait the amount of time specified by the ‘Retry Wait Period’ before attempting a connection to the redundant node (if configured). If redundancy if not defined then BridgeMaster will attempt connection to the original server. The server must be designed to not start up again (or deny access) if it is truly ‘shutdown’. BridgeMaster will enter an infinite retry loop where it will continue to retry connection with that server at the ‘Retry Wait Period’ until connection is re-established.

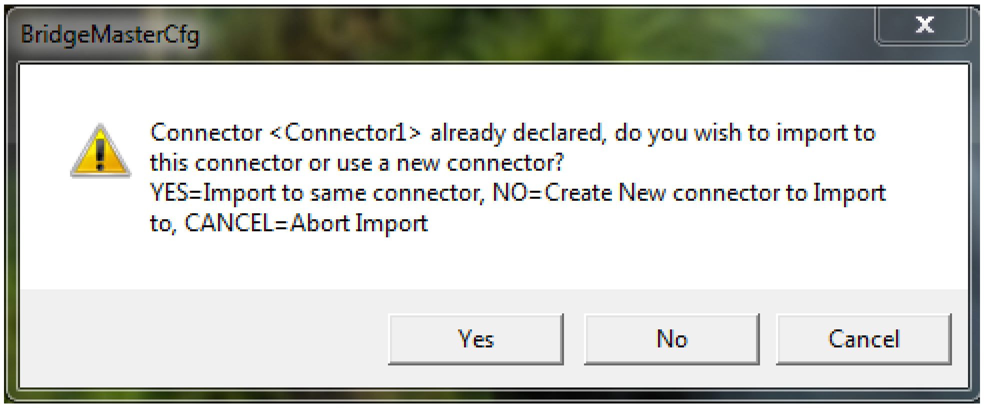
6 Import/Export

This section is provided to give examples on how to quickly setup a BridgeMaster Tag Map configuration. A BridgeMaster configuration is stored in a binary (non-editable) file but by using the Import/Export function this can allow the user to edit the configuration via an ASCII text based CSV file.

To export a BridgeMaster Tag Map configuration, open the desired BridgeMaster Configuration and select <*File* | *Export CSV…*> from the main menu. Enter the desired filename or leave it at the default, which is the same as the BMC file but with a CSV extension, to export it and click OK. The file can then be opened with Notepad or Excel and edited according to the CSV file formats at the end of this section.

After a file has been exported and edited it can then be re-imported into the existing configuration or into a new configuration. Select <*File* | *New…*> from the main menu to create a new configuration or open the desired configuration to import to. Select <*File* | *Import CSV…*> from the main menu and select the desired CSV file to import. After a successful import you can then save the new or existing configuration. If for some reason the import function fails, the error message will be displayed in the bottom message pane. Correct the situation in the import file and try to import again. Note, you may want to back up the existing BridgeMaster Tag Map Configuration first before attempting this operation. This can be done by selecting the <*File* | *Save As…*>

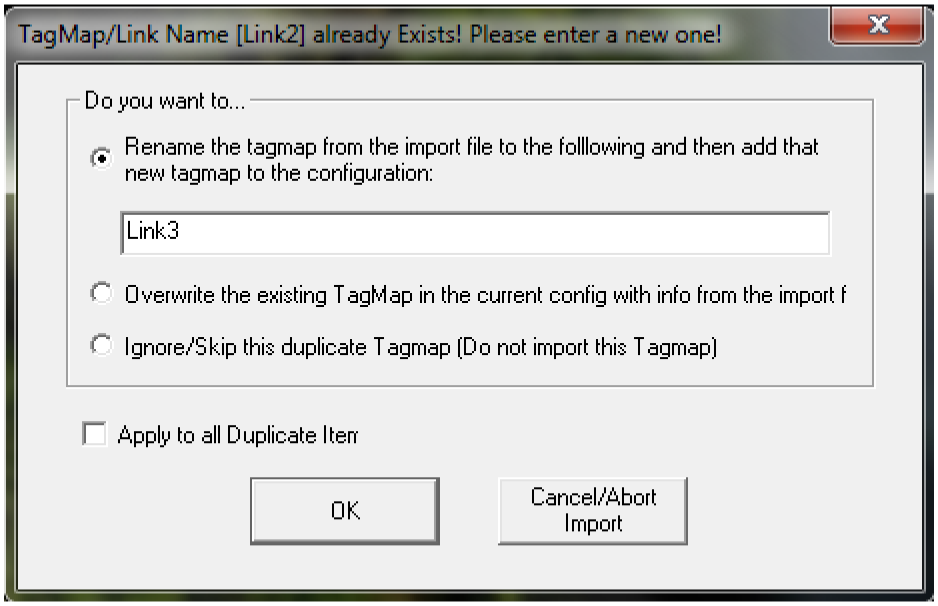
When importing a CSV file, in most cases you will be prompted with the following dialog box:



Connector <Connector1> already declared, do you wish to import to this connector or use a new connector?
YES=Import to same connector, NO=Create New connector to Import to, CANCEL=Abort Import

If the Configuration is new or you wish to import into the designated Connector <Connector1> then click the YES button to import to the same connector. Otherwise select the NO button and BridgeMaster will create and import to a New Connector (Connector2, Connector3…). To cancel the import operation at this time, select CANCEL. If you import to the same connector, all new groups and tag maps will be added to the existing connector, whereas all existing groups will be overwritten with the settings from the CSV file. If you chose to import to a New Connector, the tag maps can always be dragged and dropped to the appropriate group and the new temporary Connector can then be deleted. Also note that all Conversion Tables and Conversion Value Maps are imported at that time, any existing ones will be overwritten by what is being imported.

If the Links (or Tag Maps) already exist within the Connector/Group you are trying to import to then you will be prompted with the following dialog box:


TagMap/Link Name [Link2] already Exists! Please enter a new one!

Do you want to...

[✓ Rename the tagmap from the import file to the folllowing and then add that new tagmap to the configuration:]

Link3

Overwrite the existing TagMap in the current config with info from the import f

Ignore/Skip this duplicate Tagmap (Do not import this Tagmap)

Apply to all Duplicate Iten

At this point you have the following options:

A) Rename the Imported Link: Select the Rename radio button and enter a new Link Name and click OK. If that Link Name exists you will be prompted again and again until you find a new unique Link Name. For each duplicate links, you will again be prompted to rename.
B) Overwrite the existing TagMap- With this option, the existing Tag Map in the current configuration will be overwritten with the data imported from the import file.
C) Ignore/Skip this duplicate TagMap- With this option the existing duplicate Tag Map in the current configuration will be left untouched and the duplicate imported line will be ignored.

The *Apply to all Duplicate Items* checkbox will stop this dialog box from popping up for each duplicate it finds and will remember your last selection.

To import Tag Maps to an existing tag map configuration and overwrite the current configuration with that being imported from file:

- Select the Overwrite the existing TagMap
- Select the Apply to All Duplicate Items checkbox
- Click OK

To ignore all duplicate Tag Maps from the Import file,:

- Select the Ignore/Skip this duplicate TagMap
- Select the Apply to All Duplicate Items checkbox
- Click OK

To cancel (or abort the import operation) select the Cancel/Abort Import.

6.1 Import from other OPC Bridge Software

BridgeMaster supports the ability to easily import from other OPC Bridge software OEM's by simply allowing the user to select that OEM's exported file and import. To accomplish this, open one the products listed below and simply Export the configuration to CSV file. Then use BridgeMaster Configurator to Import using the *File* | *Import CSV* | <*Select the desired OEM format*> from the main menu where this can be any of the following:

- KepWare LinkMaster CSV
- Emerson OPC Mirror CSV
- Matrikon Data Manager CSV
- ICONICS DataWorX32 CSV
- Honeywell Experion PKS (OPC Integrator) TSV Files

Then verify all the connectors (with OPC Server names and nodes), groups (with data direction and update rates), tag maps, items, Conversion Tables and redundancy settings have been successfully recreated in BridgeMaster. Click *GO* to run the imported configuration. If no problems were logged then the conversion procession should take no more than a few minutes.

RoviSys strongly recommends that if you are using one of the OEM Bridging products listed above and/or are having issues or problems with them that you at least try out BridgeMaster using a DEMO version in order to compare differences before making the decision to convert over to BridgeMaster as others have recently and thankfully done!

6.2 Auto-Detect and Import CSV…

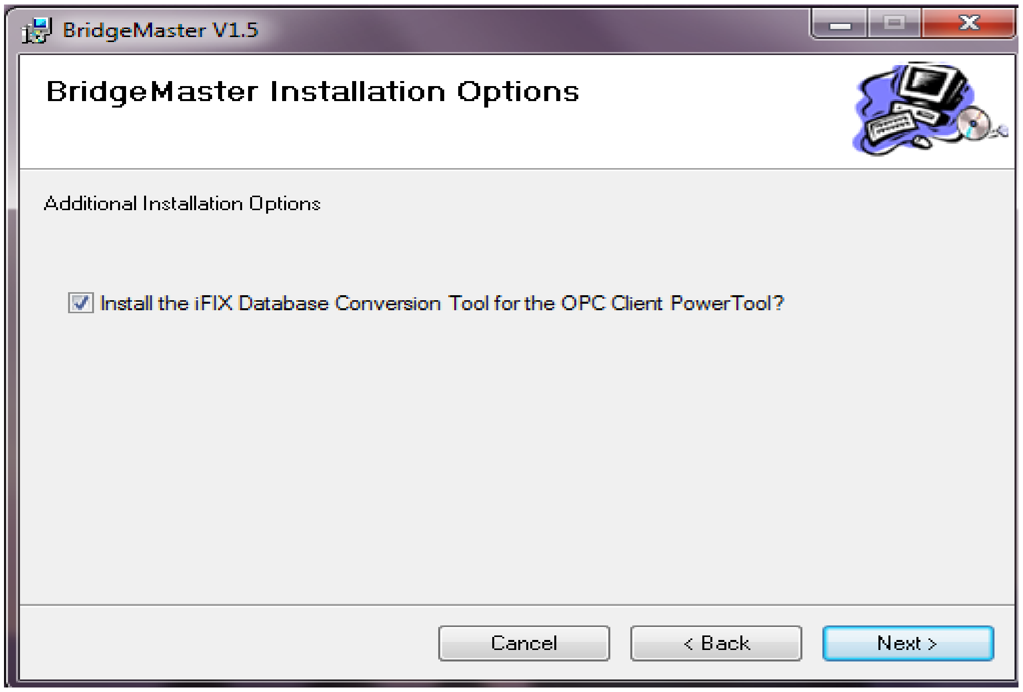
BridgeMaster supports the feature where if you don't know which OEM the CSV file is for then you can simply select *File* | *Import CSV* | *Auto-Detect and Import CSV…* and BridgeMaster will attempt to detect which format the CSV file and automatically import it into the BridgeMaster format.

6.3 iFix Database Conversion Tool

Tired of issues with the iFIX OPC Client Power Tool? BridgeMaster now supports converting GE Fanuc/Intellution iFIX OPC Client Power Tool configurations directly from iFIX into BridgeMaster. You can use BridgeMaster with the built-in iFIX Simulator (SIM or SM2) tags and get:

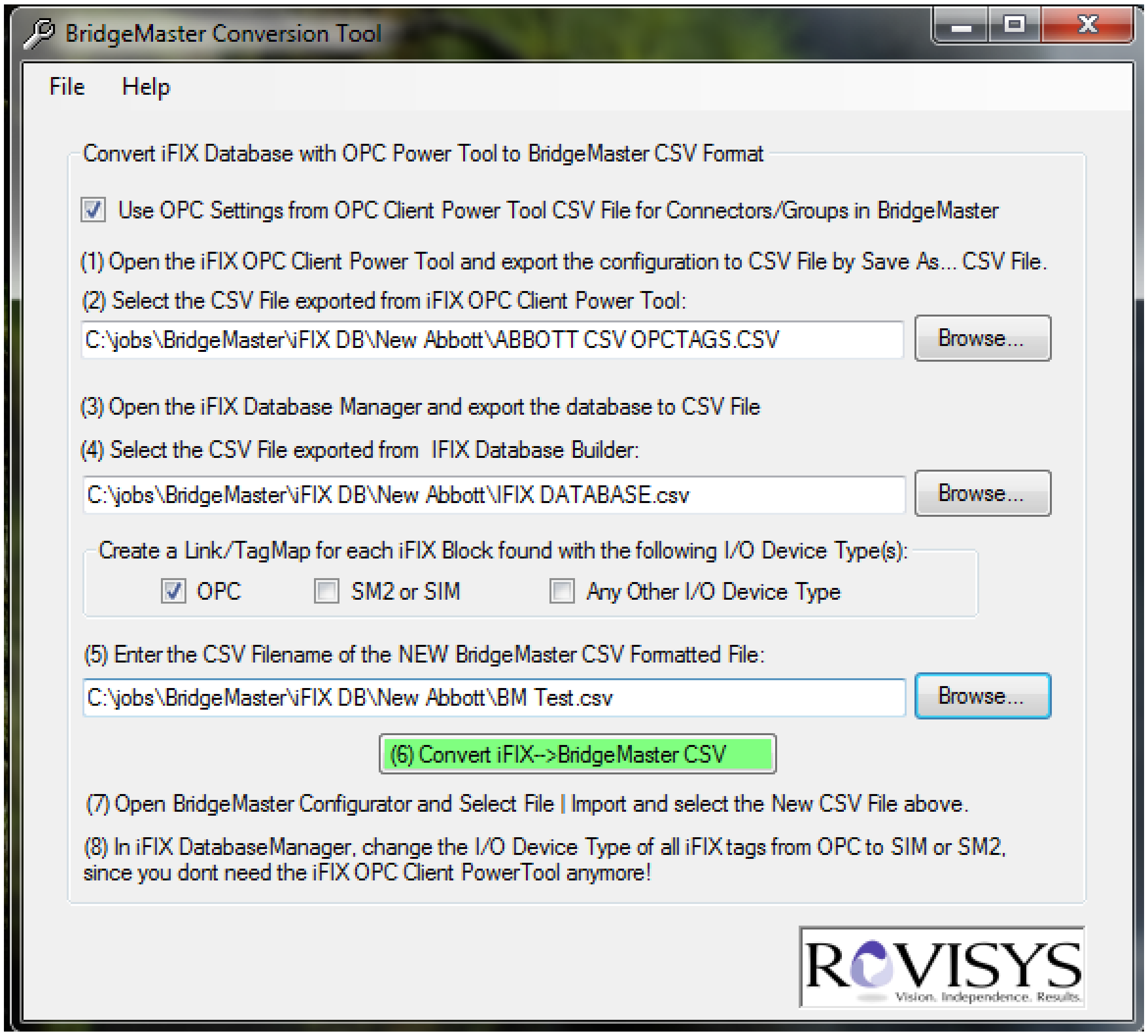
- ✓ Automatic conversion form iFIX OPC Client tags to BridgeMaster Links.
- ✓ Get faster connection startup times!
- ✓ No automatic disabling of tags when you don’t want them!
- ✓ Automatic re-connection and stays connected!
- ✓ No hang-ups when you don’t need them!
- ✓ Improved functionality including more runtime features and options, better stability, increased efficiency and enhanced redundancy.

Since the OPC Client Power Tool has been known to exhibit the above issues as well as other adverse behaviors, RoviSys has developed a solution using BridgeMaster to bypass any of those issues. This conversion feature is supported if you select the following checkbox during the installation process:


BridgeMaster V1.5
BridgeMaster Installation Options
Additional Installation Options
Cancel
< Back
Next >

[✓] Install the iFIX Database Conversion Tool for the OPC Client PowerTool?

After the tool is installed, you can run the tool by selecting *Start* | *All Programs* | *RoviSys BridgeMaster* | *iFIX DB Conversion Tool*. The following main menu will appear:



BridgeMaster Conversion Tool

File Help

Convert iFIX Database with OPC Power Tool to BridgeMaster CSV Format

[✓] Use OPC Settings from OPC Client Power Tool CSV File for Connectors/Groups in BridgeMaster

(1) Open the iFIX OPC Client Power Tool and export the configuration to CSV File by Save As... CSV File.

(2) Select the CSV File exported from iFIX OPC Client Power Tool:

C:\jobs\BridgeMaster\iFIX DB\New Abbott\ABBOTT CSV OPCTAGS.CSV Browse...

(3) Open the iFIX Database Manager and export the database to CSV File

(4) Select the CSV File exported from IFIX Database Builder:

C:\jobs\BridgeMaster\iFIX DB\New Abbott\IFIX DATABASE.csv Browse...

Create a Link/TagMap for each iFIX Block found with the following I/O Device Type(s):

[✓] OPC [ ] SM2 or SIM [ ] Any Other I/O Device Type

(5) Enter the CSV Filename of the NEW BridgeMaster CSV Formatted File:

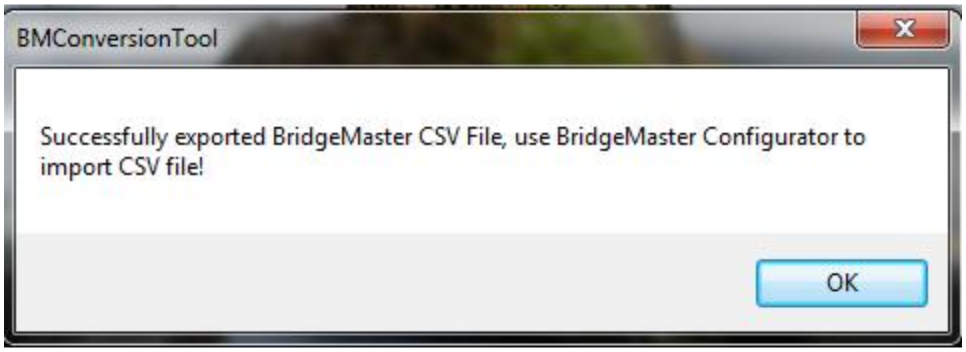
C:\jobs\BridgeMaster\iFIX DB\New Abbott\BM Test.csv Browse...

(6) Convert iFIX-->BridgeMaster CSV

(7) Open BridgeMaster Configurator and Select File | Import and select the New CSV File above.

(8) In iFIX DatabaseManager, change the I/O Device Type of all iFIX tags from OPC to SIM or SM2, since you dont need the iFIX OPC Client PowerTool anymore!

Follow the directions shown above in the main menu. Select the "*Convert iFIX->BridgeMaster CSV*" button to perform the export. If successful, the following dialog should be displayed:



BMConversionTool

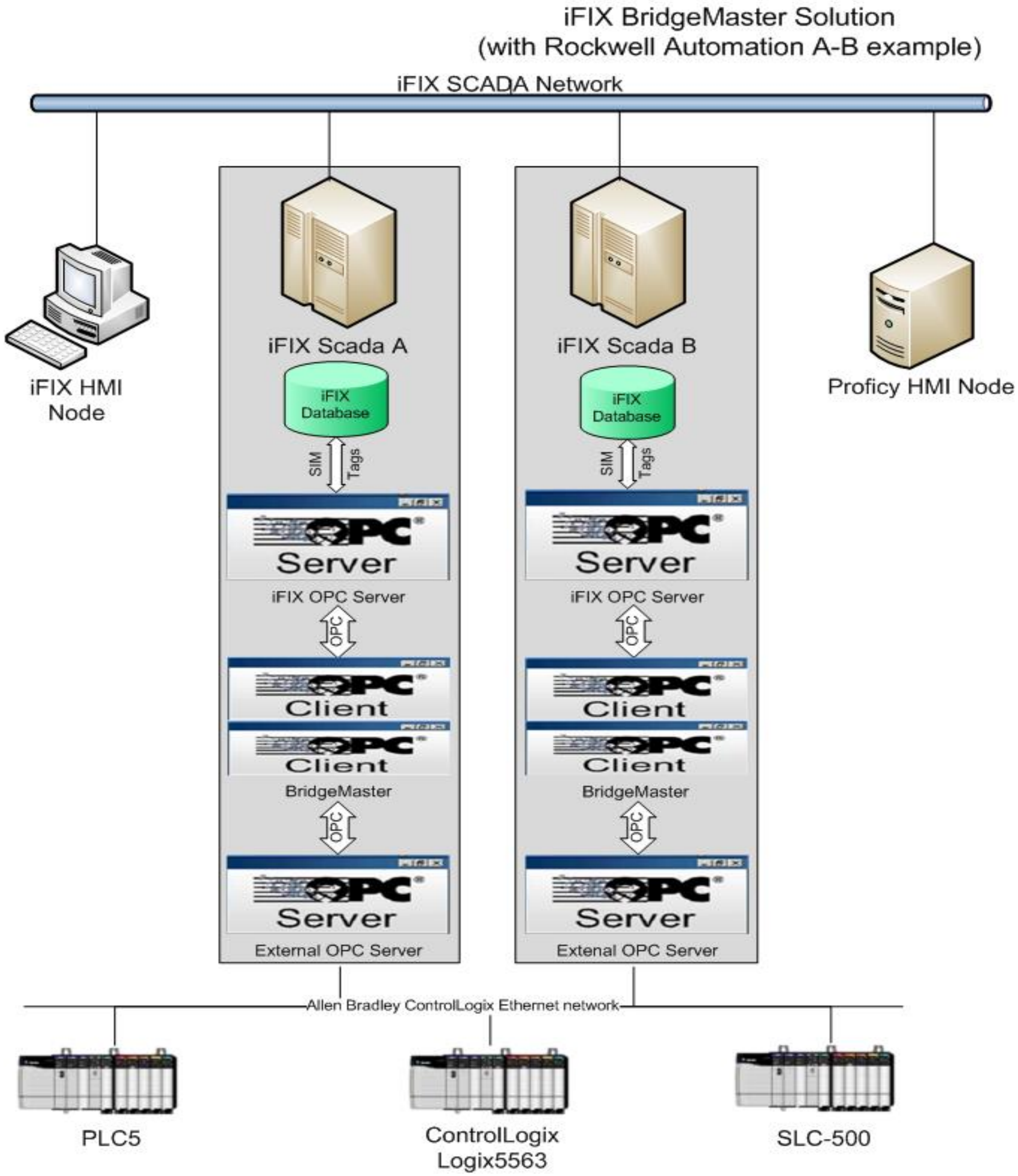
Successfully exported BridgeMaster CSV File, use BridgeMaster Configurator to import CSV file!

OK

You can then view a results log file to see if there were any errors or warnings. Note you will then need to open BridgeMaster configuration and either select a new project or use an existing one and then import the BridgeMaster CSV file above as specified from step 5. After this is accomplished you will need to setup the SIM or SM2 "simulator" tags within iFIX for all the OPC Tags you exported in order to read/write from them in BridgeMaster. Make sure you allow writes if you wish to write to them. Alternatively, if you just want to connect to the SIM or SM2 tags and don't have any OPC tags, then you can check the SM2 or SIM checkbox instead of the

OPC checkbox and then Tag Maps will be created for those tags. You will then need to fill in the other side for BridgeMaster.

An example of using BridgeMaster (a 2-headed OPC Client) to connect a redundant IFIX System via SIM Tags from the *Intellution.OPCiFIX.1* OPC Server to AB Tags in an External OPC Server (RSLinx or KepWare) is shown in the following diagram:



6.4 BridgeMaster CSV File Formats

The BridgeMaster CSV file is an ASCII Text based file containing fields delimited by a comma. Microsoft Excel or Notepad can be used to create this file and simply export it in the CSV file format. Each line (row in Excel) in the file declares exactly one line type command or tag mapping. An example of the new Version 2.0 BridgeMaster CSV file with 1 Connector, 1 group and 1 Conversion table is given below:

```
CSV_VERSION,2.0
BEGIN_CONNECTOR
CONNECTOR,APX2MATSIM,"",1,1,1,0,0,APP-CONNECT,APACSOPCDeviceServer.1,localhost,
APACSOPCDeviceServer.1,30000,localhost,ICONICS.Simulator,localhost,ICONICS.Simulator,30000
REDUNDANCYSETTINGS,A,0,0,1,15,1,7,0,10000,0,10000,""
REDUNDANCYSETTINGS,B,0,0,1,15,1,7,0,10000,0,10000,""
BEGIN_GROUPMAP
GROUPMAP,Group1,"",0,1,0.000000,1000,0,1033,0,2,0,0,0.000000,1000,0,1033,0,0,0,0,0,0
TAGMAP,Link1,"","ACM.TEST.FIC102.SETPOINT.TARGET","",0,0.000000,"SimPLC.Int1","",0,0.000000,,0
TAGMAP,Link2,"","ACM.TEST.MOTOR1.OUT","",0,0.000000," SimPLC.Int2","",0,0.000000,,0
TAGMAP,Link3,"","ACM.TEST.PV1","",0,0.000000," SimPLC.Real4","",0,0.000000,,0
TAGMAP,Link4,"","ACM.TEST.RATIO_SET_LIM.SETPOINT.RAMP","",0,0.000000,"SimPLC.Int3","",0,0.000000,,0
TAGMAP,Link5,"","ACM.TEST.RATIO_SET_LIM.SETPOINT.RRATE","",0,0.000000,"SimPLC.Int4","",0,0.000000,,0
TAGMAP,Link6,"","ACM.TEST.FIC102.AUTO","",0,0.000000,"SimulatorPLC.Boolean","",0,0.000000,,0
TAGMAP,Link7,"","ACM.DCA_TEST.ANALOG_REAL","",0,0.000000," SimulatorPLC.Real8","",0,0.000000,,0
TAGMAP,Link8,"","ACM,DCA_TEST.MODE","",0,0.000000,"SimPLC.Mode","",0,0.000000,MODECONVERSION,0
END_GROUPMAP
END_CONNECTOR
BEGIN_TABLE
TABLE,MODECONVERSION,DISCRETE
TABLE_PARAMETERS,0,0,0,0,0.000000,100.000000,0.000000,10000.000000,0.000000,100.000000,1.000000,0.0
00000,-1,-1,-1,0,0,0,0,,
TABLE_VAL,0,1
TABLE_VAL,1,2
TABLE_VAL,2,0
END_TABLE
```

File formats for each line in the CSV file are given below. The first line tells of the CSV version of the file, which is currently version 2.0. Each connector contained within the CSV file starts with a line type statement BEGIN_CONNECTOR and ends with the line statement END_CONNECTOR. The Connector is declared with a CONNECTOR line type. All groups belonging to that connector are contained within those 2 lines. Groups are defined within the declaration lines BEGIN_GROUPMAP and END_GROUPMAP. Each Group is declared with the GROUPMAP line. All tag maps existing within a group are defined as a line within the group starting with a TAGMAP declaration. Conversion tables are defined at the end of the file after all connectors, groups and tag maps are declared. Each table and table values are encapsulated within their own BEGIN_TABLE and END_TABLE declarations. The table itself is declared with a TABLE line whereas each conversion map value contained within a given table is declared with a TABLE_VAL statement.

The fields contained within the CONNECTOR statement are given in the following table. Note: a comma within the CSV file must separate each field and only the string fields mentioned in the table can contain commas within them. All string data type fields are case-insensitive.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Line Type	String	Contains the string CONNECTOR to define this line in the CSV file as a CONNECTOR declaration.	CONNECTOR
2	Connector Name	String	Contains the unique name of the Connector being declared. This field cannot contain any commas or dots and is case insensitive.	Connector1, Connector2, Interface
3	Description	String	Specifies the description of this connector. This field CAN contain commas and must be surrounded by double quotes “”. May be left blank.	User specified. “”
4	Enabled Flag	Integer	Connector Enable flag, 1-Enable, 0-Disabled.	1
5	Show/Log Errors	Integer	Show (and Log) Errors flag, 1-Yes, 0-No.	1
6	Show/Log Events	Integer	Show (and Log) Errors flag, 1-Yes, 0-No.	1
7	Show/Log Receive	Integer	Show (and Log) Read messages flag, 1-Yes, 0-No.	1
8	Show/Log Send	Integer	Show (and Log) Write Messages flag, 1-Yes, 0-No.	1
9	OPC Server ‘A’ Primary Node	String	Specifies the Host Name or IP Address of the Primary Node for the OPC Server ‘A’.	LocalHost
10	OPC Server ‘A’ Primary Program ID	String	Specifies the OPC Program ID of the Primary Host for the OPC Server ‘A’.	ICONICS.Simulator
11	OPC Server ‘A’ Secondary Node	String	Specifies the Host Name or IP Address of the Secondary Node for the OPC Server ‘A’.	167.34.56.77
12	OPC Server ‘A’ Secondary Program ID	String	Specifies the OPC Program ID of the Secondary Host for the OPC Server ‘A’. *This should be the same as the Primary one.*	ICONICS.Simulator
13	OPC Server ‘A’ Status Validation	Integer	Specifies the Status Validation period for the OPC Server ‘A’ (in milliseconds)	1000
14	OPC Server ‘B’ Primary Node	String	Specifies the Host Name or IP Address of the Primary Node for the OPC Server ‘B’	LocalHost
15	OPC Server ‘B’ Primary Program ID	String	Specifies the OPC Program ID of the Primary Host for the OPC Server ‘B’	RoviSys.OPC90Server
16	OPC Server ‘B’ Secondary Node	String	Specifies the Host Name or IP Address of the Secondary Node for the OPC Server ‘B’	167.34.56.77
17	OPC Server ‘B’ Secondary Program ID	String	Specifies the OPC Program ID of the Secondary Host for the OPC Server ‘B’. *This should be the same as the Primary one.*	RoviSys.OPC90Server

18	OPC Server 'B' Status Val.	Integer	Specifies the Status Validation period for the OPC Server 'B' (in milliseconds)	1000

The fields contained within the REDUNDANCYSETTINGS statement are given in the following table. Note: a comma within the CSV file must separate each field and only the string fields mentioned in the table can contain commas within them. All string data type fields are case-insensitive.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Line Type	String	Contains the string REDUNDANCYSETTINGS to define this line in the CSV file as a Redundancy setting declaration.	REDUNDANCYSETTINGS
2	Server Side	String	Contains A or B for which side of the Connector the settings are being declared.	A or B
3	Use Redundancy	Integer	Use Redundancy with this Server Side? 1=Yes,0-No	1
4	Use Hot Swap Standby	Integer	Use Hot-Swap Standby mode? This is not currently supported. 0=NO	0
5	Use Tag Quality for Failover	Integer	Use Tag Quality for failover? 1=Yes,0-No	1

6	Tag Quality Flags	Integer	Tag Quality flag indicating which sub-status BAD quality to failover on: (OR'd together)
			`REDUN_TAGQUAL_CONFIG_ERR      0x01`
			`REDUN_TAGQUAL_NO_CONN         0x02`
			`REDUN_TAGQUAL_COMM_FAIL       0x04`
			`REDUN_TAGQUAL_OUT_OF_SERV     0x08`
			`REDUN_TAGQUAL_DEV_FAILURE     0x10`
			`REDUN_TAGQUAL_SENSOR_FAILURE  0x20`
				15 decimal

7	Use Server Status for Failover	Integer	Use Server Status for failover? 1=Yes,0-No	1

8	Server Status Flags	Integer	Server Status flag indicating which status to failover on: (ORed together)
			`REDUN_SS_FAILED               0x01`
			`REDUN_SS_NO_CONFIG            0x02`
			`REDUN_SS_COMM_FAULT           0x04`
			`REDUN_SS_SUSPENDED            0x08`
				15 decimal

9	Use Keep Alive mechanism for Failover	Integer	Use Keep Alive Mechanism for failover? (only supported with v3.x servers) 1=Yes,0-No	1
10	Keep Alive Timeout	Integer	Keep Alive Timeout (in ms) only supported with v3.x servers)	10000
11	Use Watchdog	Integer	Use Watchdog tag for failover?	0
12	Watchdog timeout	Integer	Watchdog timeout period (in ms)	20000

13	Watchdog Tag Item ID	String	Specifies the OPC Item ID of the Watchdog tag. . This field CAN contain commas and must be surrounded by double quotes “”.	"ACM040B.DCA_TEST.ADD_OUT"

There are 3 groups defined in the above example file. The fields contained within the GROUPMAP declaration line are specified below.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Line Type	String	Contains the string GROUPMAP to define this line in the CSV file as a Group declaration.	GROUPMAP
2	Group Name	String	Contains the user-defined name of the Group being declared. This field cannot contain any commas or dots and is case insensitive. This field needs to be UNIQUE within the Connector.	Group1
3	Description	String	Specifies the description of this group. This field CAN contain commas and **must be** surrounded by double quotes “”. May be left blank.	User specified. “”
4	Data Flow Direction	Integer	Integer value representing the data flow direction for this group: 0 - (→) Data Flow from OPC Server ‘A’ to OPC Server ‘B’ 1 - (←) Data Flow from OPC Server ‘B’ to OPC Server ‘A’ 2 - (←→) Data Flow Bi-Directional	2
5	Active Flag	Integer	Group Active flag, 1-Active, 0-Disabled.	1
6	OPC Server ‘A’ Deadband	Float	Specifies the Deadband (%) for the OPC Server ‘A’ Server side group.	0.0000
7	OPC Server ‘A’ Update Rate	Integer	Specifies the Update Rate (in milliseconds) for the OPC Server ‘A’ side group.	1000
8	OPC Server ‘A’ Time Bias	Integer	Specifies the Time Bias for the OPC Server ‘A’ side group. (in minutes)	0
9	OPC Server ‘A’ LCID	Integer	Specifies the Locale ID for the OPC Server ‘A’ side group.	1033
10	OPC Server ‘A’ Keep Alive	Integer	Specifies the Keep Alive Rate for the OPC Server ‘A’ side group. (in milliseconds)	2000

11	OPC Server ‘A’ Requested Interface Version	Integer	Specifies the Requested Interface Version for the OPC Server ‘A’ side group. Values are:
			`OPC_DA_USE_BEST=0`
			`OPC_DA_VERSION1=1`
			`OPC_DA_VERSION2=2`
			`OPC_DA_VERSION3=3`
				0

			`OPC_DA_VERSION3_VQT=4`	
12	OPC Server 'A' Do Polling	Integer	Do Polling with OPC Server 'A' side group? 1=YES, 0=NO	0
13	OPC Server 'A' Use Sync Writes	Integer	Use Synchronous Writes with OPC Server 'A' side? 1=YES, 0=NO	0
14	OPC Server 'B' Deadband	Float	Specifies the Deadband (%) for the OPC Server 'B' side group.	0.00
15	OPC Server 'B' Update Rate	Integer	Specifies the Update Rate (in milliseconds) for the OPC Server 'B' side group.	1000
16	OPC Server 'B' Time Bias	Integer	Specifies the Time Bias for the OPC Server 'B' side group. (in minutes)	0
17	OPC Server 'B' LCID	Integer	Specifies the Locale ID for the OPC Server 'B' side group.	1033
18	OPC Server 'B' Keep Alive	Integer	Specifies the Keep Alive Rate for the OPC Server 'B' side group. (in milliseconds)	2000

19	OPC Server 'B' Requested Interface Version	Integer	Specifies the Requested Interface Version for the OPC Server 'B' side group. Values are:
			
			`OPC_DA_USE_BEST=0`
			`OPC_DA_VERSION1=1`
			`OPC_DA_VERSION2=2`
			`OPC_DA_VERSION3=3`
			`OPC_DA_VERSION3_VQT=4`
				0

20	OPC Server 'B' Do Polling	Integer	Do Polling with OPC Server 'B' side group? 1=YES, 0=NO	0
21	OPC Server 'B' Use Sync Writes	Integer	Use Synchronous Writes with OPC Server 'B' side? 1=YES, 0=NO	0
22	OPC Server 'A' Refresh Rate	Integer	Specifies the Refresh Rate (in milliseconds) for the OPC Server 'A' side group.	0
23	OPC Server 'B' Refresh Rate	Integer	Specifies the Refresh Rate (in milliseconds) for the OPC Server 'B' side group.	0

For each group there can be an unlimited number of tag maps defined within it. The fields contained within each TAGMAP declaration line are specified below.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Line Type	String	Contains the string TAGMAP to define this line in the CSV file as a Tag Map.	TAGMAP
2	Link Name	String	Specifies the name of this link or Tag Map. This field cannot contain any commas. The name must be unique among all tag maps within the entire Connector.	Link001
3	Description	String	Specifies the description of this tag map. This field CAN contain commas and **must be** surrounded by double quotes “”. May be left blank.	User specified. “”
4	Tag ‘A’ OPC Item ID name	String	Specifies the OPC Item name to connect within OPC Server ‘A’. This field CAN contain commas and must be surrounded by double quotes “”.	“YHS103_B.DEVCTLA.MODE”
5	Tag ‘A’ Access Path	String	Specifies the OPC Access Path to connect to within OPC Server ‘A’. May be left blank. This field CAN contain commas and must be surrounded by double quotes “”.	“”
6	Tag ‘A’ Use Item Deadband	Int	Use Item Deadband? (1=Yes, 0=NO) *This is only used with Analog points that support OPC Version 3.0*	1
7	Tag ‘A’ Item Deadband	Float	Item Deadband for Tag ‘A’ in %. (0-100) *This is only used with Analog points that support OPC Version 3.0*	2.5
8	Tag ‘B’ OPC Item Name	String	Specifies the OPC Item name to connect to within OPC Server ‘A’. This field CAN contain commas and must be surrounded by double quotes “”.	“Device1.TIC108_B.MODE”
9	OPC Server ‘B’ Access Path	String	Specifies the Access Path to connect to within OPC Server ‘B’. May be left blank. This field CAN contain commas and must be surrounded by double quotes “”.	“”
10	Tag ‘B’ Use Item Deadband	Int	Use Item Deadband? (1=Yes, 0=NO) *This is only used with Analog points that support OPC Version 3.0*	1
11	Tag ‘B’ Item Deadband	Float	Item Deadband for Tag ‘A’. (0-100) *This is only used with Analog points that support OPC Version 3.0*	2.5
12	Conversion Table Name	String	Specifies the Conversion Table name used for scaling by this tag map. May be left blank if no scaling	MODECONVERSION

			is required.	
13	Data Master	Integer	Specifies which server is the Data master if the Group's Data direction is set to bi-directional. Note, this setting is overwritten if the group's data flow direction is not bi-direction. 0- OPC Server 'A' 1- OPC Server 'B'	0 or 1

Each Conversion/Scaling Table can be defined with the TABLE declaration line. The format for this line type is specified below.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Line Type	String	Contains the string TABLE to define this line in the CSV file as a Converter Table.	TABLE
2	Table Name	String	Specifies the unique table Name of this configuration. This field cannot contain any commas.	User specified.
3	Table Type	String	Specifies the Table Type: DISCRETE LINEAR GAINOFFSET BCDBYTE BCDWORD XFERQUALITY XFERTIMESTAMP ARRAYELEMENT NOCONVERSION	DISCRETE

Each Conversion Table needs to be followed by a TABLE_PARAMETERS declaration line. The format for this line type is specified below.

Field #	FIELD NAME	FIELD TYPE	FIELD DESCRIPTION	Examples
1	Discrete Map Type	Integer	Contains the integer number designated what type of discrete mapping table it is (if its Discrete Mapping): 0-Long to Long 1-Long to String 2-String to Long	0
2	Discrete Passthru Type	Integer	Contains the integer number designated what type of	0

			discrete passthru it is: 0-Pass all other values 1-Don't pass unmapped values	
3	Clamping Type	Integer	Contains the integer number designated what type of clamping to use: 0-No clamping 1-Use Clamping values 2-Clamp on output range	0
4	Data Type	Long Integer	VARTYPE output data type (*not currently supported)*	0
5	Input Low	Float	Specifies the input low value if scaling is used.	0
6	Input High	Float	Specifies the input high value if scaling is used.	1000
7	Output Low	Float	Specifies the Output low value if scaling is used.	0
8	Output High	Float	Specifies the Output high value if scaling is used.	10000
9	Clamp Low	Float	Specifies the Clamp low value if scaling is used.	0
10	Clamp High	Float	Specifies the Clamp high value if scaling is used.	1000
11	Gain	Float	Specifies the Gain value if Gain/Offset is used.	1
12	Offset	Float	Specifies the Offset value if Gain/Offset is used.	0
11	Index1	Integer	Specifies the 0-based index in the 1st dimension if Array element is used. (-1 means not defined)	-1
12	Index2	Integer	Specifies the 0-based index in the 2nd dimension if Array element is used. (*not currently supported*)	-1
13	Index3	Integer	Specifies the 0-based index in the 3rd dimension if Array element is used. (*not currently supported*)	-1
14	User Clamp Low	Boolean	Flag to Specify to use the User specified Clamp low value if clamping is used.	0
15	User Clamp High	Boolean	Flag to Specify to use the User specified Clamp High value if clamping is used.	0
16	Output Clamp	Boolean	Flag to Specify to use the	0

	Low		Output specified Clamp low value if clamping is used.	
17	Output Clamp High	Boolean	Flag to Specify to use the Output specified Clamp High value if clamping is used.	0
18	Units	String	User-defined engineering units string (not supported)	
19	Expression	String	String Expression (*not currently supported*)	

For Discrete Mapping scale type, each Conversion Value Map can be defined with a TABLE_VAL declaration line. The format for this line type is specified below.

Field #	**FIELD NAME**	**FIELD TYPE**	**FIELD DESCRIPTION**	**Examples**
1	Line Type	String	Contains the string TABLE_VAL to define this line in the CSV file as a Conversion Value Map.	TABLE_VAL
2	OPC Server 'A' Discrete Value	Long Integer or String	Specifies the discrete value on the OPC Server 'A' side of the conversion.	0
3	OPC Server 'B' Discrete Value	Long Integer or String	Specifies the discrete value on the OPC Server 'B' side of the conversion.	1

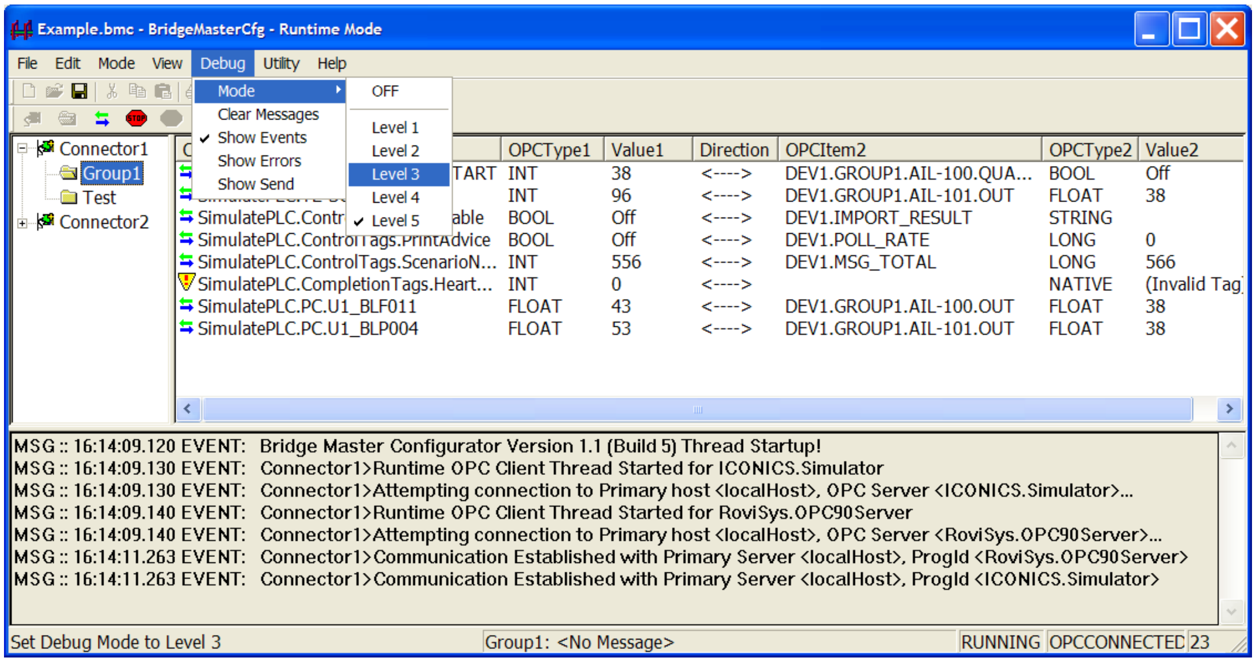
Note, if the CSV file is being exported from an Excel spreadsheet, it will contain commas separating all fields, even with the command line types such as BEGIN_CONNECTOR and END_CONNECTOR, they will have extra commas at the end of their lines. BridgeMaster will still parse the file correctly when it is imported. The extraneous commas are simply ignored.

7 Error Diagnosis and Debugging

This section details how to diagnosis errors and debug BridgeMaster in the event that Runtime operations are not functioning properly. BridgeMaster logs errors along with informational messages to the message log pane in runtime mode only. This log is used to diagnose potential error conditions.

7.1 Debug Operations

Simple debug operations can be executed during runtime from the main menu's debug menu item. The following drop down list in the main menu displays these options:



C	OPCType1	Value1	Direction	OPCItem2	OPCType2	Value2
TART	INT	38	<---->	DEV1.GROUP1.AIL-100.QUA...	BOOL	Off
	INT	96	<---->	DEV1.GROUP1.AIL-101.OUT	FLOAT	38
SimulatePLC.Contr able	BOOL	Off	<---->	DEV1.IMPORT_RESULT	STRING	
SimulatePLC.ControlTags.PrintAdvice	BOOL	Off	<---->	DEV1.POLL_RATE	LONG	0
SimulatePLC.ControlTags.ScenarioN...	INT	556	<---->	DEV1.MSG_TOTAL	LONG	566
SimulatePLC.CompletionTags.Heart...	INT	0	<---->		NATIVE	(Invalid Tag
SimulatePLC.PC.U1_BLF011	FLOAT	43	<---->	DEV1.GROUP1.AIL-100.OUT	FLOAT	38
SimulatePLC.PC.U1_BLP004	FLOAT	53	<---->	DEV1.GROUP1.AIL-101.OUT	FLOAT	38

```
MSG :: 16:14:09.120 EVENT:  Bridge Master Configurator Version 1.1 (Build 5) Thread Startup!
MSG :: 16:14:09.130 EVENT:  Connector1>Runtime OPC Client Thread Started for ICONICS.Simulator
MSG :: 16:14:09.130 EVENT:  Connector1>Attempting connection to Primary host <localHost>, OPC Server <ICONICS.Simulator>...
MSG :: 16:14:09.140 EVENT:  Connector1>Runtime OPC Client Thread Started for RoviSys.OPC90Server
MSG :: 16:14:09.140 EVENT:  Connector1>Attempting connection to Primary host <localHost>, OPC Server <RoviSys.OPC90Server>...
MSG :: 16:14:11.263 EVENT:  Connector1>Communication Established with Primary Server <localHost>, ProgId <RoviSys.OPC90Server>
MSG :: 16:14:11.263 EVENT:  Connector1>Communication Established with Primary Server <localHost>, ProgId <ICONICS.Simulator>
```

```
Set Debug Mode to Level 3    Group1: <No Message>    RUNNING  OPCCONNECTED  23
```

In runtime mode, BridgeMaster supports the following debug options:

Mode (Debug Level)

This option is used to set the debug mode of the BridgeMaster. The higher the debug level the more messages will be printed to the screen. To turn all debug messages off set the MODE to OFF.

Clear Messages

This option is used to clear all messages in the message pane window. Select this option to clear all messages in the window. This should be done from time to time since the message pane window simply uses a buffer which keeps growing as new messages appear. The buffer size is limited only by the amount of memory available on the machine. It is not a circular buffer. Note: you can easily clear all messages by single clicking the ☒ button in the toolbar. If you wish to not worry about this then deselect the Show Events, the Show Errors and the Show Send and no messages will appear in this window.

Show Events

Select this option to enable BridgeMaster to print out all event messages to the message log window (or file). A check mark will appear next to this menu item when it is enabled. To disable this select it again so that the check mark disappears. This option is useful in verifying certain communication events from the BridgeMaster to the OPC Server(s).

Show Errors

Select this option to enable BridgeMaster to print out all error messages to the message log window (or file). A check mark will appear next to this menu item when it is enabled. To disable this select it again so that the check mark disappears. Note if an error log directory was defined then leaving this option in the checked state may fill up that log file quickly. This option is useful in verifying communications with the OPC Server(s) and with the configuration.

7.2 OPCEnum.exe

This section details how setup OPCEnum service. This is used to browse for remote servers and for runtime startup operation in order to obtain the Class ID or Globally Unique ID (GUID) of the remote OPC Server used for connection to the server during startup. If the OPC Server on the remote node is not already also registered on the local PC, then OPCEnum.exe is used on the remote machine in order for BridgeMaster to be able to connect with the server remotely in runtime. ***Note: OPCEnum.exe is now NOT REQUIRED since RoviSys has crafted an alternative mechanism utilizing a simple text file to lookup the GUID.***

To install and setup OPCEnum.exe on the remote node, following the procedure below:

1. Obtain a copy of OPCEnum.exe
2. Log into the remote node as administrator or with administrator privilege.
3. Place OPCEnum.exe in the \\Windows\System32 directory.
4. Register OPCEnum, by running the command from a DOS prompt:
    a. Opcenum.exe /regserver
5. Install OPCEnum as a service by running the command from a DOS prompt:
    a. Opcenum.exe /service
6. Open DCOM Configurator by running dcomcnfg.exe (or using BridgeMaster and selecting *Utility* **|** *Launch DCOM Configurator*.
7. Browse to OPCEnum and add the user that will be running BridgeMaster for Launch and access privilege. Alternatively, use can leave the defaults alone for the OPCEnum in DCOM Config and then just setup the default User privilege for the user account under which BridgeMasterService will be run from. See sections below for additional security setup.
8. If a Windows Firewall is present on the remote machine make sure to:
    a. Add TCP Port 135 to the list of exceptions.
    b. Add OPCEnum.exe to the list of exceptions.
    c. Configure the “Edit Limits” DCOM properties for ANOYNMOUS USER and the user account under which BridgeMasterService will be run from.

7.3 Quick Setup and Troubleshooting Reference

This section details how a user can quickly setup BridgeMaster and diagnose common setup problems. The following checklist should be consulted if startup issues arise:

1. Verify TCP/IP protocol is enabled on the node running BridgeMaster and all OPC Servers.
2. Setup/Install both OPC Servers according to OEM specifications.
3. Attempt to ping the node running the desired OPC server you wish to connect with. If you cannot ping a remote host then a network setup/hardware issue exists.
4. If the network is a Domain, then the user account under which BridgeMaster is running needs to have the proper security setup in order to connect with the OPC Server(s) on the local/remote machine. This user account needs to exist (be created under) that domain and thus have permissions to log in to the domain if both nodes are running under that domain. Consult the System Administrator of the domain for setting up user accounts.
5. If the network is a Workgroup, the easiest way to connect the PCs it to place both nodes in the same workgroup and then define the same username and password on both nodes and then login as the same user on each node. In essence if BridgeMaster is running on node A and an OPC Server is running on node B and both are part of the same

workgroup then both node A and node B need to have a User account (with the same password) setup with the proper security privilege (in DCOM) to access and run the OPC Server on node B. BridgeMaster must be run under that user account if the network is a Workgroup. Getting DCOM to work over a Workgroup is difficult without totally removing all security. A User can be defined on both PCs yet still not work since each PC assigns an SID to the created User account; they will still be a different SIDs even though it is the same user name. This is where having a Domain to do your authentication makes life a lot easier.

6. The user account running BridgeMaster must be added to the list of users in DCOM with Access permissions and Launch permissions for that OPC Server. Consult the Using DCOM OPC Servers for setting up this OPC Server security privilege.
7. Configure each OPC Server if configuration is required. If the OPC90 Server is used, you will need to configure the desired blocks first within the OPC90 Server and then save the configuration. You will also need to setup DCOM for the OPC90 Server to run as Interactive user so only 1 copy runs at a time. (If 2 copies of the OPC90Server are running then the second copy will fail since both will require access to the same COM port)
8. Configure your BridgeMaster configuration. Add a Connector, one or more groups, add the desired tag maps for each group.
9. Place BridgeMaster in runtime mode.
10. If a message appears in message log pane stating that ‘Current connection does not support Advise! Defaulting to polling of OPC Server’, then the problem may be due to a security issue as stated in #4, 5, 6 and 8 above.
11. Upon startup, if BridgeMaster displays the message ‘Error setting up Advise (using ConnectionPoint V2.0) for IOPCDataCallback Interface’ or ‘Could not locate Connection Point for IOPCDataCallback Advise Interface’ then verify #8 above and/or verify that the server and DCOM is setup correctly on the server machine and that any required client software is installed on the BridgeMaster node. In most cases this is either the result of not having the proper security setup in DCOM and/or having older or missing software components in the system of the server.
12. Upon startup, if BridgeMaster displays the message ‘Access Denied’ then verify #6 above and possibly #5.
13. Using the Server Status dialog, in runtime if the Total Reads=0, then it is possible that the Advise subscriptions are not working correctly. If problem cannot be corrected with one of above steps, then the Connector and OPC Server (in the worst case scenario) may need to be setup for ‘Polling’. As stated in #8 above, the advise connections will require security on the BridgeMaster node to be setup such that the user account which runs the OPC Server on the remote node will have ‘permission’ to send back data to the BridgeMaster node and thus to BridgeMaster. This is not the same as the security setup to the Server in step #6. Since BridgeMaster is not an OPC Server, it cannot be setup with DCOM to enable such permission. The network workgroup/domain will enable/define this privilege and thus that is why it is essential to correctly setup steps #4 or 5 and 8.
14. As a quick reference, here are some reasons that BridgeMaster may fail to connect to the OPC Server or get data:
    a. The OPC Server is stopped on the remote node.
    b. Local machine had Norton AntiVirus on it, but it was disabled. The network used the POPProxy.dll provided by the AntiVirus software vendor to send/receive data and since it was disabled, then you could not ping the remote node.
    c. OPC Server machine required a reboot.

d. Network cables were bad.
e. The BridgeMaster Group's 'Active Group' checkbox was unchecked (disabled) meaning no data will be received or sent for that group.

8 Using DCOM OPC Servers

This section details how to setup and diagnosis problems with DCOM Servers over a TCP/IP network. This is not an exhaustive guide and only discusses the common problems found with DCOM Network setup.

Note, in most cases, the TCP/IP network protocol must be installed and enabled on the PC before DCOM can be used. Each OPC Server before it can be connected to must also be registered in the Registry on the PC from which BridgeMaster is running. The OPC Server usually performs this step when it is installed or run for the first time. If not it can be registered by running "regsvr32 <server.exe>" where server.exe is the name of the executable OPC Server with the full path entered. The OPC Proxy DLL must also be registered on each PC using DCOM. This is accomplished by placing a copy of <opcproxy.dll> in the Windows System32 directory. On Windows NT, this directory is usually "C:\WINNT\System32". Again, to register it, run "regsvr32 opcproxy.dll". A message will be displayed stating that the registration succeeded. If the OPC Proxy is not registered on the local or remote node then BridgeMaster upon startup of runtime or upon a server switchover/fail-over will display a message as to this situation.

Warning: The RoviSys Company takes no responsibility from any damage to property or personnel or lost time that may be incurred due to making changes to the Registry.

8.1 OPC Servers on Windows NT/XP/Windows 2000

Servers running on Windows NT/XP/Windows 2000 are no longer supported! Like DDE support from Microsoft which still works but is no longer officially supported by Microsoft, RoviSys BridgeMaster will still run or connect to these systems but it is no longer supported!

Access to the OPC Server is setup using the <dcomcnfg.exe> DCOM Configuration program. For the given OPC Server, it must be verified under the Identity Tab of the server that the Identity is set to the "Interactive User" as opposed to the launching user. The launching user is the default, so this must be changed using the DCOM Configuration program, dcomcnfg.exe.

Also while using the DCOM Configuration program, you must enable the security permissions correctly on the remote machine in order for the BridgeMaster node to access and launch the OPC Server remotely. If problems still persist then select the Security tab and verify that the "Use Custom Access Permissions" is checked and that the user account for which the BridgeMaster node is running under is added as a User of that group. Also verify that the same account is listed under the "Use Custom Launch Permissions" checkbox. If this is a problem to add a specific user in an NT Domain, the easiest way to enable these permissions for BridgeMaster (although not recommended) is to add the User "Everybody" for the two groups mentioned above. Of course, this will allow any client/machine on the network to connect and launch the OPC Server remotely, which has the unfortunate side effect of leaving security wide open. Otherwise consult your Network Administrator. When running BridgeMaster in runtime mode, if the error E_ACCESSDENIED is displayed then this is the cause of that problem.