Training Manual
D301660X412
May, 2010

ObjectServer for Beginners

Remote Automation Solutions
www.EmersonProcess.com/Remote

This page has been left intentionally blank

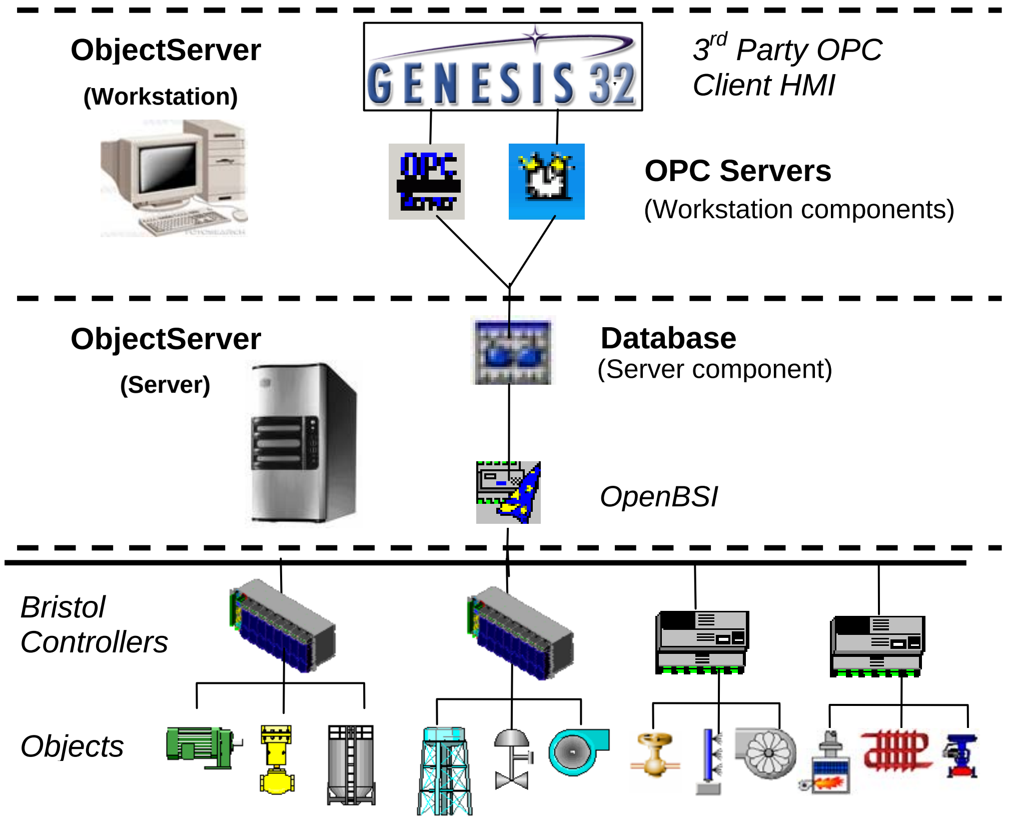
Table of Contents

Chapter 1 – Introduction – What is ObjectServer?

This chapter introduces ObjectServer and provides a broad overview of what it does.

ObjectServer is an **OPC Server** for **Bristol Controllers**, providing OPC real-time and alarm & event data to any OPC Compliant software – normally Human Machine Interface (HMI) or SCADA software packages i

The following graphic shows ObjectServer's relationship to various components in the technical environment:



Expanding on our definition in the box above, *ObjectServer takes real-time and alarm data from objects that Bristol controllers manage and make it available to third-party OPC clients using OPC servers (that provide the data from a central database).*

Let's examine each part of this statement.

Objects

Objects are tanks, pumps, valves, filters, or any kind of plant equipment used to monitor or control a product that requires processing (such as water, gas, or oil).

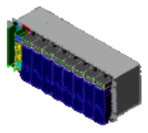
Sensors attached to these objects generate electronic impulses – signals – which they send to controllers. The values the sensors send to the controllers (such as flow, pressure, or temperature) are called “input signals”. Input signals provide information on the state of the objects.

In turn, the controllers send values – “output signals” – to the objects. Output signals can change the state of the object, by opening or closing a valve or switching a pump on or off.

ObjectServer communicates with all of the controllers in the whole process system. It retrieves the object data and places it in a storage location, or database. The OPC server accesses the database and provides this data to OPC clients.

Human supervisors can then monitor and control all the objects in the process from an OPC client workstation.

Bristol Controllers



Automated Control

Bristol controllers use programs you write (called “control strategy files” or “loads”) to control the objects (tanks, pumps, valves, and so on). Currently there are two types of Bristol controllers: the older NW3000 series (which are programmed in ACCOL) and the newer ControlWaves, (which are programmed with an IEC 61131 control language). ObjectServer can send data to and retrieve data from both types of controllers.

Supervisory Control

Even though the controllers are managing the objects, you need to monitor whether a particular pump is actually on or off or whether a particular tank is full yet. You may also need to override the controller program and manually control the objects. ObjectServer enables you to do this by collecting data from the controllers and, using its OPC server, making the data available to any OPC client (through its Human Machine Interface).

ObjectServer Database

ObjectServer ”serves up” or directs data from controllers to OPC clients, but requires only a minimum of ”know how” and configuration effort on your part to get that data collection started.

Data Collection

ObjectServer uses a program called **RDI3000** to communicate with OpenBSI to get real-time controller data, and place it into the database. A section at the end of this chapter explains this process in greater detail.

OPC Servers

OPC Data Server

Unlike OpenEnterprise, ObjectServer does not include its own HMI. Instead, it supplies OPC data to third-party OPC clients (or HMIs) via its OPC servers.

The ObjectServer OPC server and OPC Alarm and Event servers serve OPC Data Access data and OPC Alarm and Event data from the ObjectServer database directly to OPC clients.

Support for Legacy Applications

The ObjectServer OPC server supports legacy OPC client applications that have been configured using the Bristol BSI OPC server (also often referred to as the “Bristol Standalone OPC Server”). OPC tags configured with the Bristol BSI OPC server would normally require reformatting to be read. However, you can configure the ObjectServer OPC server to support tags in the Bristol BSI OPC Server format, reducing the need for display modification. More instructions on this procedure appear later in this guide (see Chapter 7 page 12).

OPC

OPC stands for **OLE for Process Control**. So, firstly we need to understand what OLE is.

OLE

OLE (Object Linking and Embedding) is a Microsoft®-developed technology. It initially allowed you to take objects from one application (such as Excel® spreadsheets) and embed them in another application (such as a Word document). In this example Excel is the server and Word is the client.

OLE technology was later written into Microsoft’s COM (Component Object Model) and then DCOM (Distributed Component Object Model), which enabled programs running on different PCs over a network to communicate and exchange data.

The OPC Standard

The OPC Foundation (www.opcfoundation.org) then added extensions to OLE technology to standardize the process of exchanging data in process control applications in a server-client environment. OPC Servers obtain process control data from a source and make it available to OPC client applications.

OPC Clients

OPC client programs are often called HMI applications because they provide process data in a form that humans can easily understand.

OPC clients are applications that display real-time or alarm data from OPC servers in a graphical form. For instance, a tank level can be drawn as a rectangle that shrinks or grows according to the actual level of the tank.

Some examples of third-party OPC client applications are Genesis32™, iFix™, Citect, or InTouch®. OPC clients can request data from any OPC server. The ObjectServer OPC server provides these OPC clients with data from Bristol controllers. ObjectServer does not have its own OPC client; instead, it provides data to OPC clients from third-party vendors.

When you create a dynamic object on a display, you can browse available OPC servers for OPC tags. The OPC client will list the ObjectServer OPC server. If you select the ObjectServer OPC server, you can browse for OPC tags, which you can then drop into the data source field for the dynamic object.

What are OPC Tags?

OPC tags are strings that the OPC server uses to identify the object values that you want to view. Typically, an OPC tag includes the OPC server's ProgramID, the PrimaryID of the signal, and the signal property. The OPC server allows you to browse for controller tags as objects from any OPC client. It then inserts the selected object into the display as a tag. There will be more about tags later.

WebTookit

WebToolkit is a web based application from OpenEnterprise Development that enables you to display ObjectServer data in a web browser without using OPC. This removes the need for dedicated OPC client workstations.

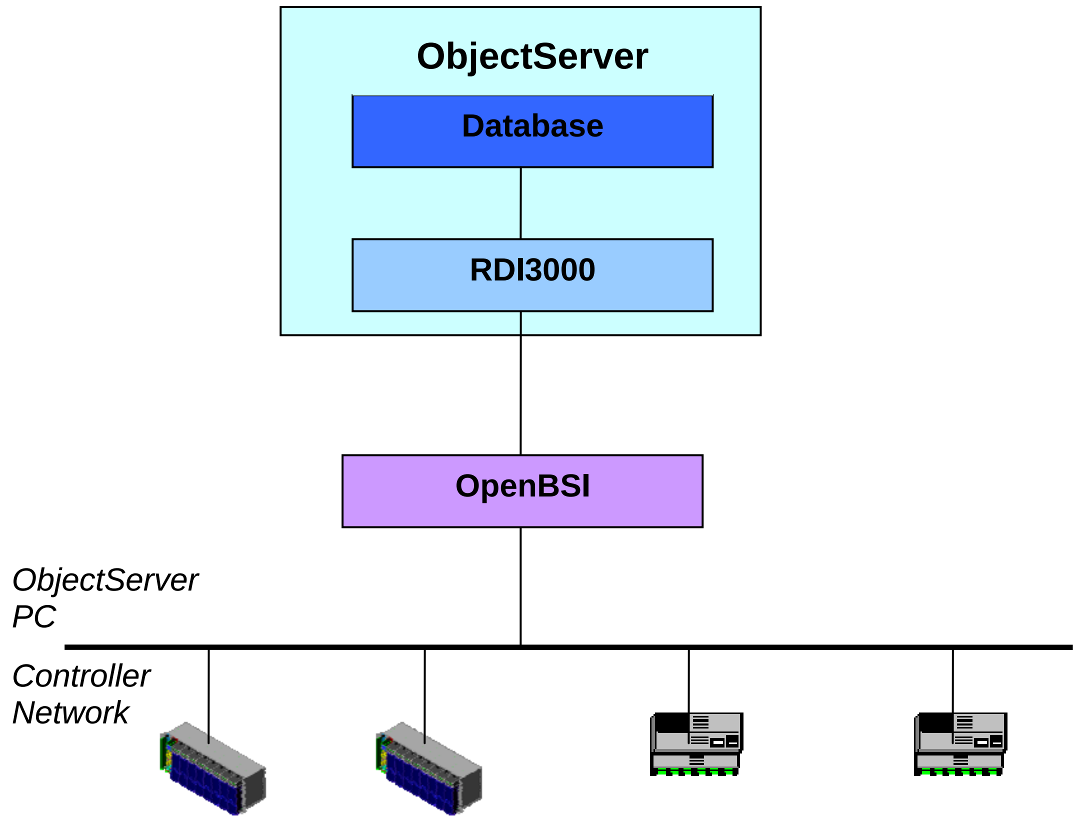
If you would like to view your ObjectServer data from any location over the Internet, consider setting up a web server running WebToolkit rather than investing in one or more dedicated OPC client workstations.

For more information on WebToolkit, see the manual *WebToolkit for Beginners*.

Getting Data into the ObjectServer Database

The ObjectServer database stores data collected from your network of remote process controllers. OpenBSI, using the communications driver program RDI3000, handles the actual communication between both types of Bristol controllers and the ObjectServer database. Typically, OpenBSI and RDI3000 both run on the ObjectServer computer.

The Controller-to-ObjectServer Communication Chain



Data goes from the controllers into the ObjectServer database by three methods:

- **Alarm Data Collection**
  RTU sends an alarm message to ObjectServer when a signal value passes a predetermined limit
- **Polled Data Collection**
  ObjectServer asks RTU for specific signal values at regular intervals
- **Report By Exception (RBE) Collection**
  RTU sends a change of value report to ObjectServer

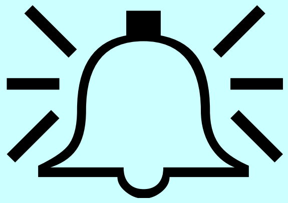
Note that the RTU initiates two of the methods while ObjectServer initiates one. Let's look at these three methods in more detail.

Alarm Data Collection

Alarms occur in a controller when a particular signal goes outside a pre-defined range or changes state into an alarm state. Typical alarm conditions might be that a liquid level is too high, a temperature is too low, or that a pump has failed to start.

The Bristol controller sends a message to ObjectServer when an alarm condition occurs. Once ObjectServer receives the alarm message, it raises the alarm. The ObjectServer OPC Alarms and Events Server then allows the third-party OPC client software to display the alarm message to the monitoring operator.

An alarm means something just happened and needs attention. For example, the controller detects that a pump has failed or a pressure signal is too high….

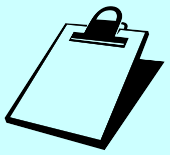


The alarm message tells the operator that something potentially serious has happened.

Polled Data Collection

Most people are familiar with the term **polling** in connection with elections. Every two years, for example, your town might have an election for mayor, and people go to the polls to vote. That's similar to the type of polling we're discussing here. When using the polled data collection method, ObjectServer sends data requests to the controllers according to a pre-defined schedule. For example, you may want to collect a certain group of signals every two hours.

Polled Data Collection operates on a <u>schedule, such as</u> "Collect all flow total signals every hour" or "Collect all logical signals every minute". It doesn't matter whether the data changes; Polled Data Collection just collects the values anyway.



All signals collected as part of the same scheduled collection are said to be in the same scan **"timeclass"**. For example, if you need to collect hourly flow totals you define an hourly timeclass, and all hourly flow totals are collected as part of that timeclass. An ObjectServer tool called Poll List Builder automatically includes signals for a particular scan timeclass into structures called **"poll lists"**. Similarly, if you had other signals that you wanted collected every minute, you would create a 1-minute scan time class, and so on.

Report by Exception (RBE) Collection

The controller initiates Report by Exception (RBE) Collection. Important differences exist between data collected by polling and data collected by RBE.

The RBE module in the controller runs at regular intervals. It says, “*Check to see if the data is different from the last time we collected it. If it hasn’t changed, don’t bother collecting it. If it has changed, collect it and send a report of the change to ObjectServer.*”

Value:	Collect it?:
SAME	☐
CHANGED	☑
SAME	☐
SAME	☐
SAME	☐
CHANGED	☑
SAME	☐

With polling-collected data, the ObjectServer requests the data from the Bristol controller at regular intervals. However, with RBE-collected data, the controller reports to ObjectServer only when a signal’s value changes.

With polling, ObjectServer collects the data at the specified time regardless of whether there has been a change. With RBE, if a signal doesn’t change, no communication occurs between the controller and ObjectServer. If your data changes slowly, this can be a much more efficient use of network resources.

With polling, if a signal’s value changes just after a poll, it could be a comparatively long time before ObjectServer registers that change (based on the frequency of polling). However, with RBE, as soon as the signal’s value changes, the RTU notifies ObjectServer of the change.

For logical (Boolean) signals, a report is sent to ObjectServer when the signal changes state (from on to off or off to on). For analog signals, a report transmits only if the signal’s value changes significantly from its previous value. The determination of whether or not a change is significant is controlled by the **deadband**. The deadband is a range above and below the signal’s value, and you must configure that range for **every** signal within the RTU (though multiple signals can use the same deadband, making configuration easier).

If the value of an analog signal has not changed more than the deadband since it was last sent to the ObjectServer, any change is considered insignificant, and no report is sent to the ObjectServer.

As you can see, RBE collection reduces the amount of data that has to be collected while allowing changed data to be displayed more rapidly than would be possible via polled collection.

Well, that’s enough basic information on what ObjectServer is and how it accomplishes its task in conjunction with OpenBSI. Let’s get started by installing these programs. The next chapter describes how to do this.

[This page intentionally left blank.]

Chapter 2 – OpenBSI and ObjectServer Installation

This chapter describes the installation options for ObjectServer and the process of installing ObjectServer on your computer.

Before You Begin

Before installing ObjectServer, you need to know and decide what functions or roles ObjectServer should have on this computer. The options are:

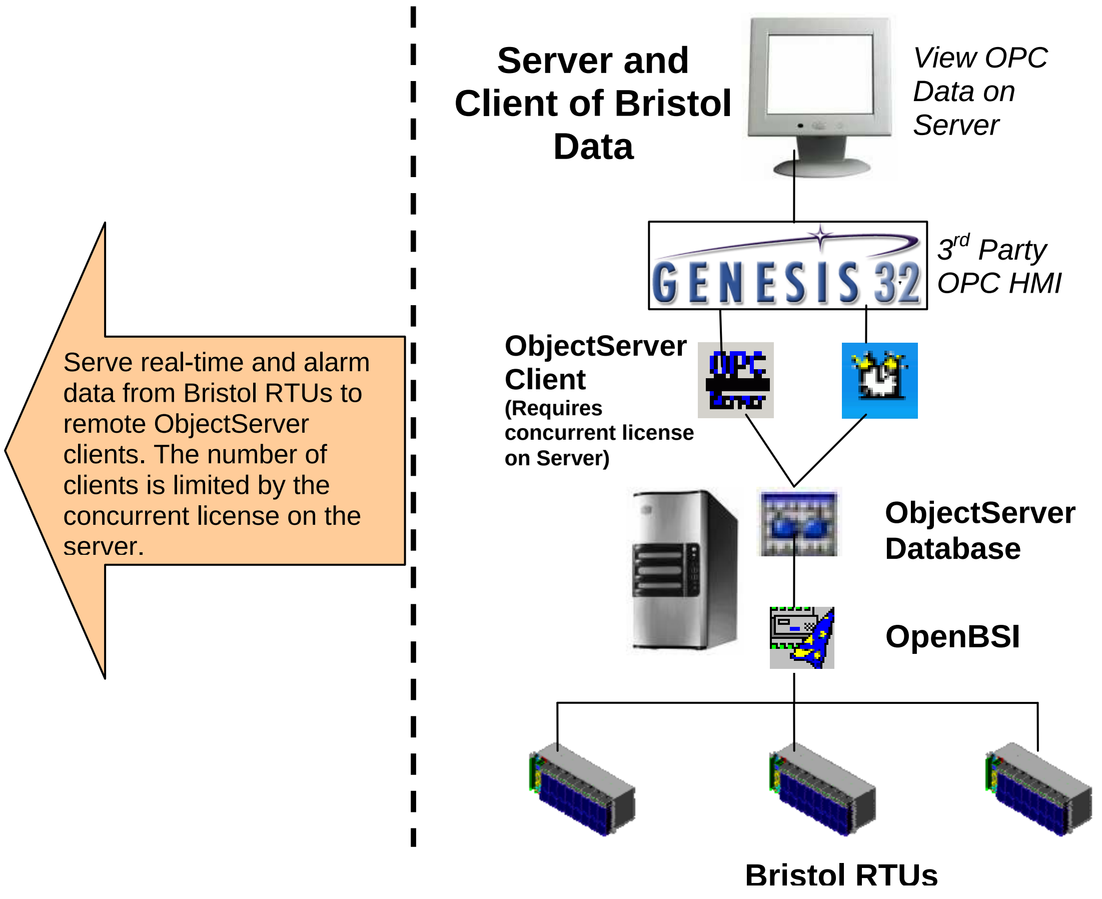
- **Complete OPC server**, collecting data from Bristol controllers and making the data available through OPC
- **Centralised data collector**, collecting data from Bristol controllers and storing within a local repository
- **OPC access to another computer**, sourcing OPC data from a centralised data collector

Role 1 – Complete OPC Server

In this role, the computer acts as both an ObjectServer server and client. It runs OpenBSI and the ObjectServer database to collect data from Bristol controllers and serves that data to the ObjectServer client that is installed on this machine.

The ObjectServer client on this computer is an ObjectServer workstation, even though it exists on the same computer. In order to serve tags, the client requires a concurrent license from the server that is running on the same machine. When the ObjectServer client starts, it runs the Workstation License Manager, which connects with the server’s Concurrent License server to determine that a concurrent license is available.

A third-party OPC client can then display the data on this machine. Of course, the ObjectServer database on this machine also serves any remote ObjectServer workstations that request data, as long as the number of connected workstations does not exceed the number allowed on the server’s concurrent license.


Serve real-time and alarm data from Bristol RTUs to remote ObjectServer clients. The number of clients is limited by the concurrent license on the server.
Server and Client of Bristol Data
View OPC Data on Server
GENESIS 32
3rd Party OPC HMI
ObjectServer Client (Requires concurrent license on Server)
ObjectServer Database
OpenBSI
Bristol RTUs

Role 2 – Centralised Data Collector

If this is the intended role, then you only need to install the ObjectServer database. Of course, you also need to install OpenBSI and configure any RTUs with it **before** you run and configure ObjectServer. This computer then becomes a server of Bristol RTU data to ObjectServer workstations.

The ObjectServer database installation includes a number of components that are required for serving Bristol data:

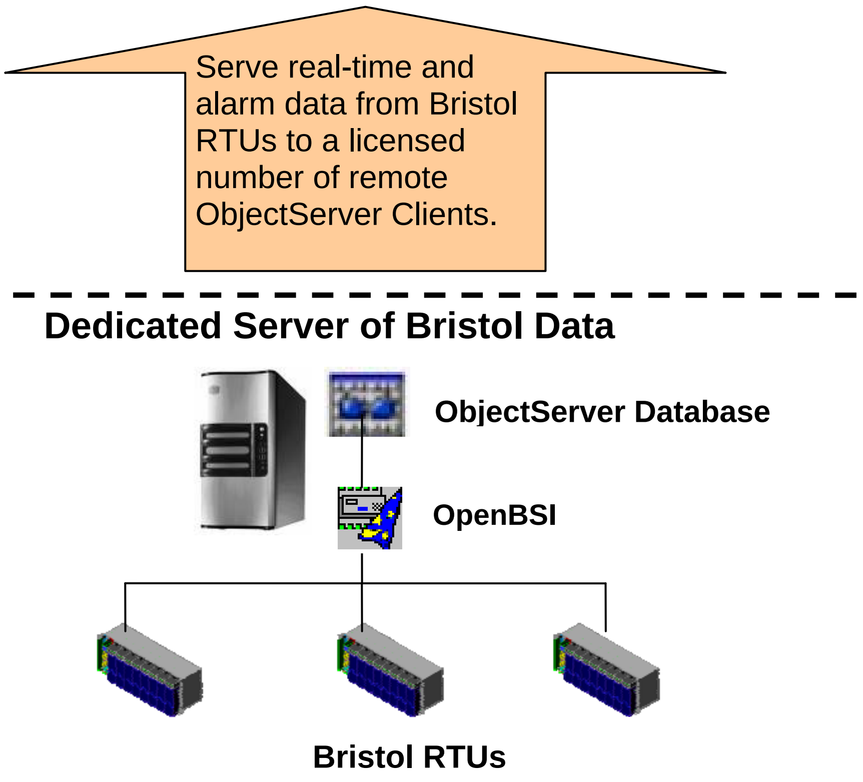
- The ObjectServer database
- The Session Manager
- The NW3000 and ControlWave device interface
- Database configuration tools
- The Concurrent License Server (CSL)
- The License Manager

For this role, this computer requires the ObjectServer database but does **not** need the ObjectServer client. The ObjectServer client consists of the Data Access OPC server, the Alarm and Event OPC server, and a client license verification tool called the Workstation License Manager.

You also need to apply for a concurrent license for any connected ObjectServer clients (also known as workstations) that want to serve tags to third-party OPC clients. For example, if you require five ObjectServer workstations to provide tags, the server must have a concurrent license for five ObjectServer workstations. The

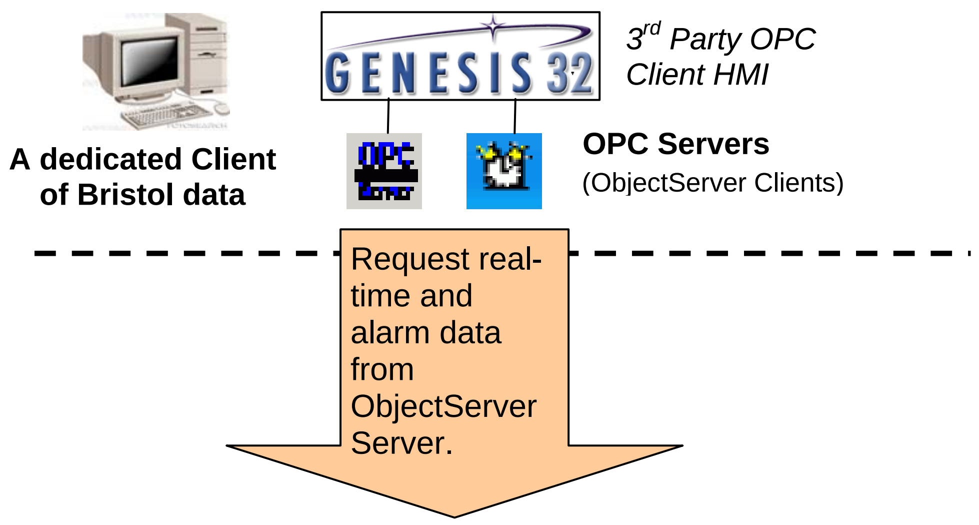
ObjectServer server uses an application called the Concurrent License Server (CSL) to determine the number of connected workstations. The concurrent license installed on the server determines how many ObjectServer workstations can connect at the same time and serve tags.

The following diagram shows how the dedicated server role works.


Serve real-time and alarm data from Bristol RTUs to a licensed number of remote ObjectServer Clients.
Dedicated Server of Bristol Data
ObjectServer Database
OpenBSI
Bristol RTUs

Role 3 – OPC Access to another Computer

For this role, the computer becomes an ObjectServer client (also known as an ObjectServer workstation). You install the Workstation License Manager on this machine, along with the Data Access and Alarm & Event OPC servers. The ObjectServer client serves OPC tags to local third-party OPC clients as long as the server to which it connects can provide it with a concurrent license.


3rd Party OPC Client HMI
GENESIS 32
A dedicated Client of Bristol data
OPC Servers
(ObjectServer Clients)
Request real-time and alarm data from ObjectServer Server.

So, before you install ObjectServer, it's essential to determine which options the computer should perform.

More about "servers and clients"

Like people, computer programs can be either clients or servers. Sometimes the computer running the program is called a server or a client, but strictly speaking it is the **program** that is either a server or a client. The rules are always the same: a server provides something and a client requests something from the server. With computers, the commodity is usually data, but with human beings, it could be anything. Take for instance the chain of servers and clients involved when we eat out at a restaurant. The restaurant supplies us with a meal when we request it. So from our point of view, the restaurant is the server.

However, the restaurant has to get the ingredients for our meal from a supplier. So when the restaurant orders food from its suppliers, the restaurant becomes a client to the food supplier. The food supplier is now a server to the restaurant.

In the same way, a computer program can be a server in one scenario and a client in another. For instance, the OPC servers provide OPC data to any third-party OPC HMI that requests it, so they are servers in this context. However, to the ObjectServer database, the OPC servers are always seen as clients, because they request the data from the database. So, the ObjectServer installation process refers to the OPC servers as the "ObjectServer client". They are collectively referred to as "client" (singular). This expression includes both the real-time and alarm-event OPC servers.

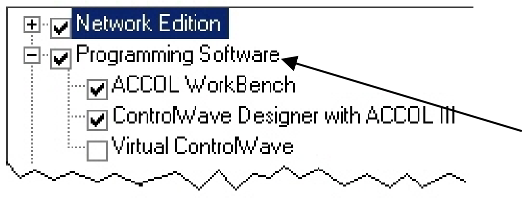
The following headings show the ObjectServer installation options you can select based on the role you've chosen for the computer.

Installation Option 1 - Complete OPC Server

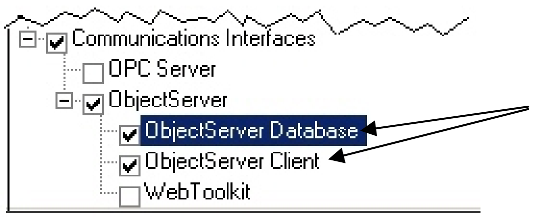
If the installation computer runs the ObjectServer database (collecting data from Bristol RTUs and displaying data from the database via a third-party OPC client HMI), then select the following options from the OpenBSI CD when you install ObjectServer:

- **OpenBSI Network Edition** – enables the Bristol controller network to collect Bristol controller data.
- **ACCOL Workbench** – enables you to write control strategy files for Bristol 33x controllers.
- **ControlWave Designer** – enables you to write control strategy files for ControlWave controllers.
- **ObjectServer database** – stores collected data that the ObjectServer client (that is, the OPC servers) can access.
- **ObjectServer client** – provides the real-time and alarm-event OPC servers, which enable third-party OPC HMIs to obtain and display Bristol OPC data from the ObjectServer database.

Select these options from the Select Features screen to enable your computer to be a dual server and client of Bristol OPC data.


☑ Network Edition
☑ Programming Software
☑ ACCOL WorkBench
☑ ControlWave Designer with ACCOL III
☐ Virtual ControlWave

Select **Network Edition**, along with **ACCOL Workbench** (if using Bristol 33x RTUs), and **ControlWave Designer** (if using ControlWave RTUs)


☑ Communications Interfaces
☐ OPC Server
☑ ObjectServer
☑ ObjectServer Database
☑ ObjectServer Client
☐ WebToolkit

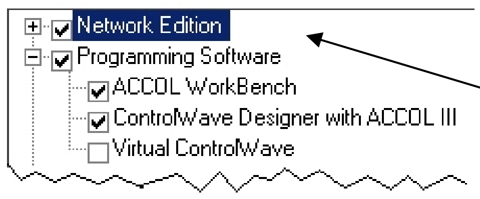
Also, select the **ObjectServer Database** and the **ObjectServer client** option from here.

Installation Option 2 – Centralised Data Collector

If the computer is dedicated solely to serving Bristol RTU data to clients, then select the following options from the OpenBSI CD when you install ObjectServer:

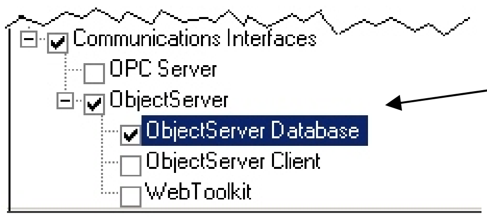
- **OpenBSI Network Edition** – enables the Bristol controller network to collect controller data.
- **ACCOL Workbench** – enables you to write control strategy files for Bristol 33X controllers.
- **ControlWave Designer** – enables you to write control strategy files for ControlWave controllers.
- **ObjectServer Database** – stores collected data that the ObjectServer client (that is, the OPC servers) can access..

When you start the installation program, it presents an automatic sequence of screens (called a “wizard”) from which you select the options you want. One of these screens is Select Features. Following is a screen shot of the installation options you need to select from that screen so the computer can function as a dedicated server of Bristol RTU data.



☑ Network Edition
☑ Programming Software
☑ ACCOL WorkBench
☑ ControlWave Designer with ACCOL III
☐ Virtual ControlWave

Select **Network Edition**, along with **ACCOL Workbench** (if using Bristol 33x RTUs), and **ControlWave Designer** (if using ControlWave RTUs)



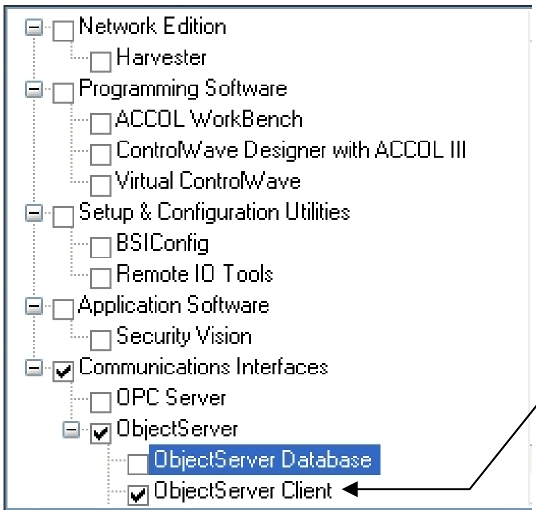
☑ Communications Interfaces
☐ OPC Server
☑ ObjectServer
☑ ObjectServer Database
☐ ObjectServer Client
☐ WebToolkit

Also, select the **ObjectServer Database** option from here.

Installation Option 3 - OPC Access to Another Computer

If you want the computer to display ObjectServer data that another computer running the ObjectServer database collects, then when you install ObjectServer you only need to select the **ObjectServer Client** option from the installation wizard's Select Features screen.

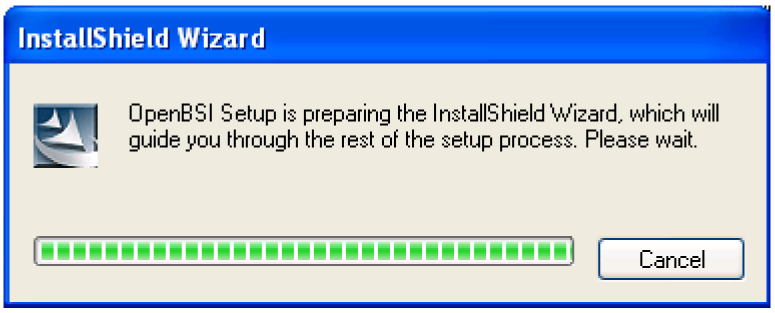
This option installs the real-time and alarm-event OPC servers, which enable third-party OPC HMIs obtain and display Bristol OPC data. Of course, you also need to install the third-party HMI software on this computer.


- [ ] Network Edition
  - [ ] Harvester
- [ ] Programming Software
  - [ ] ACCOL WorkBench
  - [ ] ControlWave Designer with ACCOL III
  - [ ] Virtual ControlWave
- [ ] Setup & Configuration Utilities
  - [ ] BSIConfig
  - [ ] Remote IO Tools
- [ ] Application Software
  - [ ] Security Vision
- [✓] Communications Interfaces
  - [ ] OPC Server
  - [✓] ObjectServer
    - [ ] ObjectServer Database
    - [✓] ObjectServer Client

Select only the **ObjectServer Client** option if this is a dedicated ObjectServer OPC client workstation.

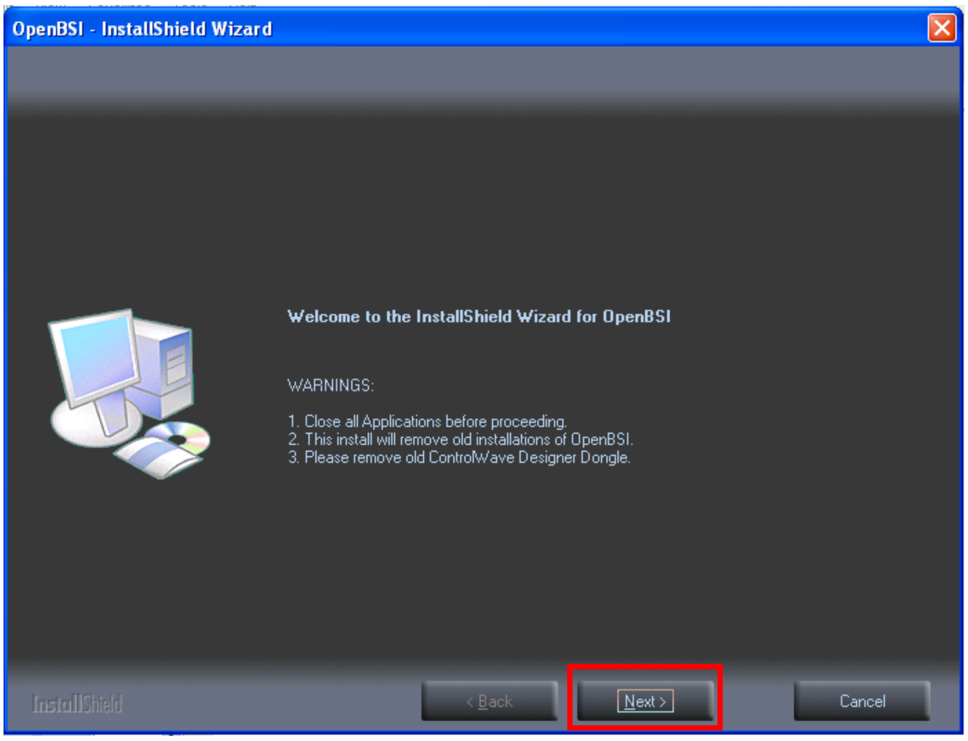
ObjectServer Installation – Step by Step

1. Place the OpenBSI installation CD into the CD drive on your computer and close the drive. If "autoplay" is turned off on your computer, when the contents of the CD are displayed, double click on the OpenBSI.exe program. If you have "autoplay" turned on, the InstallShield Wizard dialog displays automatically:



OpenBSI Setup is preparing the InstallShield Wizard, which will guide you through the rest of the setup process. Please wait.

2. The OpenBSI InstallShield Wizard Welcome screen displays:-



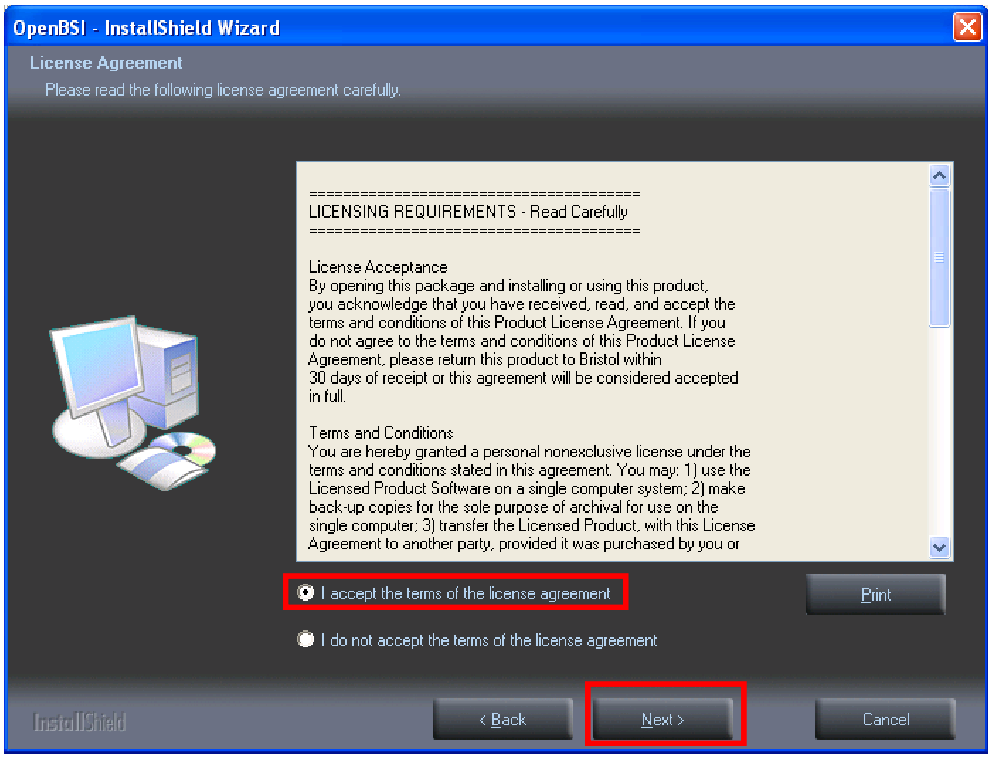
Welcome to the InstallShield Wizard for OpenBSI

WARNINGS:

1. Close all Applications before proceeding.
2. This install will remove old installations of OpenBSI.
3. Please remove old ControlWave Designer Dongle.

Review the warnings (performing any necessary actions) and click **Next>** (highlighted above).

3. Read the Licensing Requirements carefully (click **Print** to print the requirements). Select **I accept the terms of the license agreement** and click **Next>**.

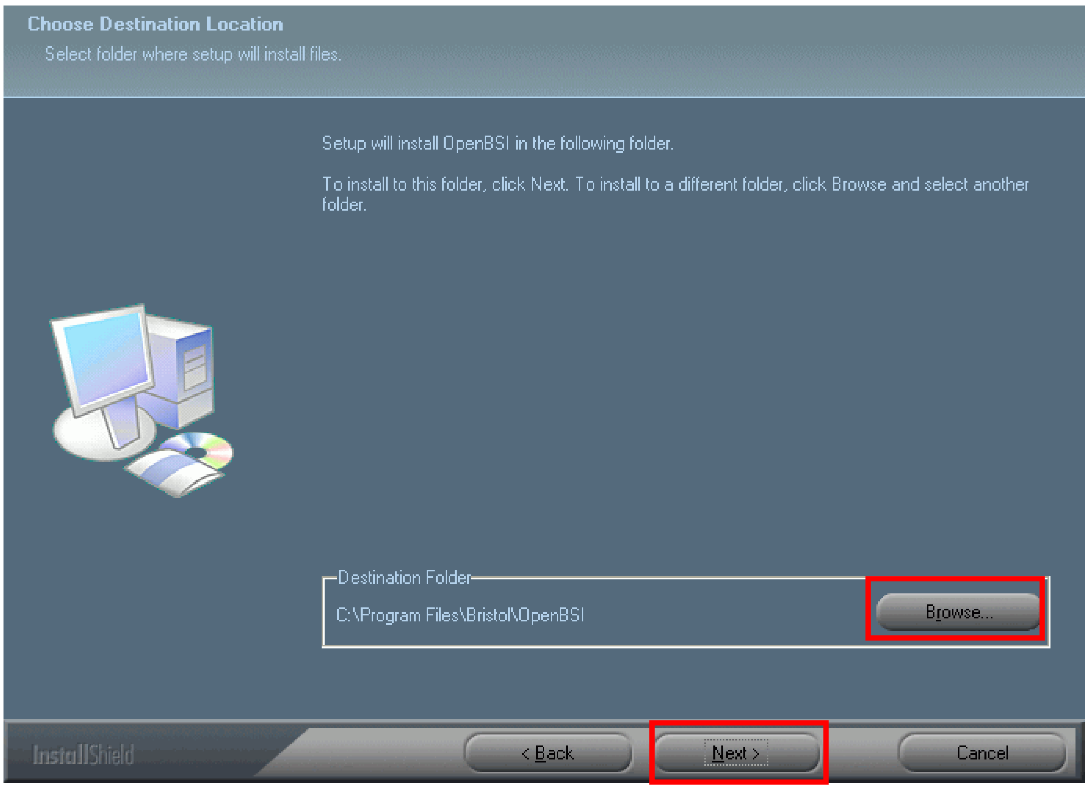

OpenBSI - InstallShield Wizard
License Agreement
Please read the following license agreement carefully.
Print
InstallShield
< Back
Next >
Cancel

```
=======================================
LICENSING REQUIREMENTS - Read Carefully
=======================================

License Acceptance
By opening this package and installing or using this product,
you acknowledge that you have received, read, and accept the
terms and conditions of this Product License Agreement. If you
do not agree to the terms and conditions of this Product License
Agreement, please return this product to Bristol within
30 days of receipt or this agreement will be considered accepted
in full.

Terms and Conditions
You are hereby granted a personal nonexclusive license under the
terms and conditions stated in this agreement. You may: 1) use the
Licensed Product Software on a single computer system; 2) make
back-up copies for the sole purpose of archival for use on the
single computer; 3) transfer the Licensed Product, with this License
Agreement to another party, provided it was purchased by you or
```

[✓ I accept the terms of the license agreement]

I do not accept the terms of the license agreement

4. The Choose Destination Location dialog shows the default installation location for the software (c:\Program Files\OpenBSI), but you can change this location by clicking **Browse**. Click **Next>**.



Choose Destination Location

Select folder where setup will install files.

Setup will install OpenBSI in the following folder.

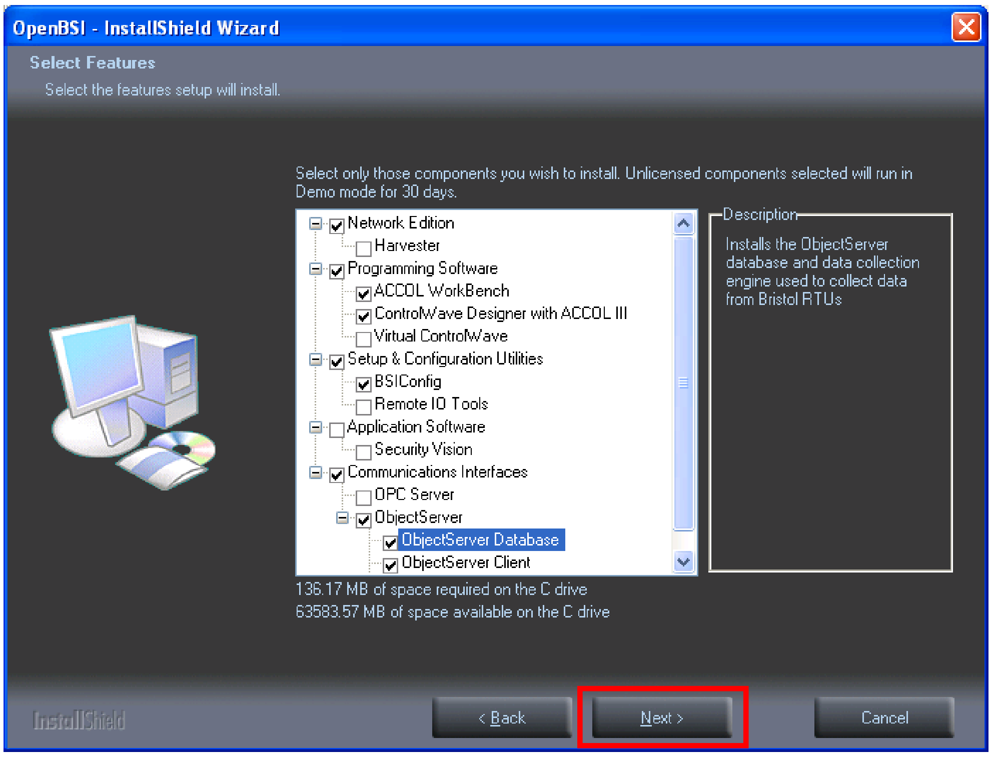
To install to this folder, click Next. To install to a different folder, click Browse and select another folder.

Destination Folder

C:\Program Files\Bristol\OpenBSI

5. Once the Select Features screen displays, scroll through it carefully to determine the programs you want to install. (If you are uncertain which options to select, review the *Before You Begin* section at the beginning of this chapter.)

You can click **Cancel** at this point if you need to review the installation options (you can restart the installation later). Once you know what options you need, click **Next>** to continue.


OpenBSI - InstallShield Wizard

Select Features

Select the features setup will install.

Select only those components you wish to install. Unlicensed components selected will run in Demo mode for 30 days.

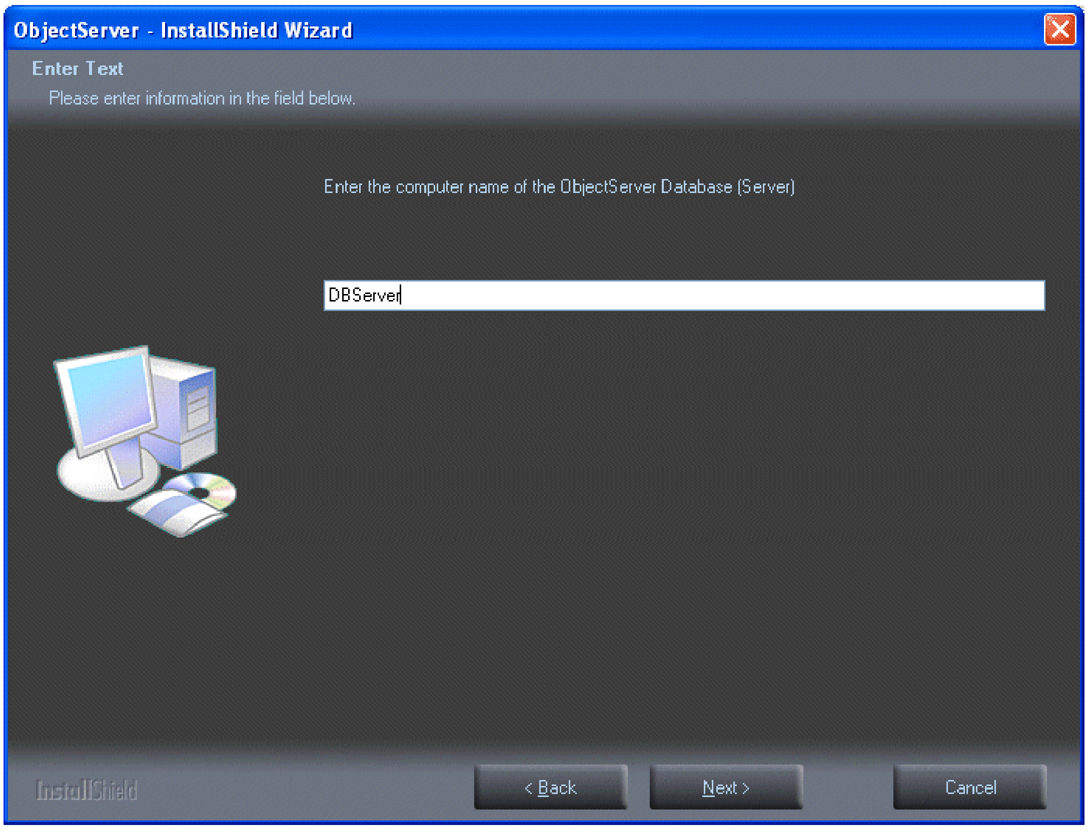
- [✓] Network Edition
  - [ ] Harvester
- [✓] Programming Software
  - [✓] ACCOL WorkBench
  - [✓] ControlWave Designer with ACCOL III
  - [ ] Virtual ControlWave
- [✓] Setup & Configuration Utilities
  - [✓] BSIConfig
  - [ ] Remote IO Tools
- [ ] Application Software
  - [ ] Security Vision
- [✓] Communications Interfaces
  - [ ] OPC Server
  - [✓] ObjectServer
    - [✓] ObjectServer Database
    - [✓] ObjectServer Client

Description

Installs the ObjectServer database and data collection engine used to collect data from Bristol RTUs

136.17 MB of space required on the C drive
63583.57 MB of space available on the C drive

6. If you are installing only the ObjectServer client, one more selection screen displays before the installation begins. Use this screen to identify the name of the computer that houses the ObjectServer database to which we want to connect. Remember: use the server's Windows **domain name** rather than an alias configured in the Hosts file (see below). Click **Next>**.


ObjectServer - InstallShield Wizard

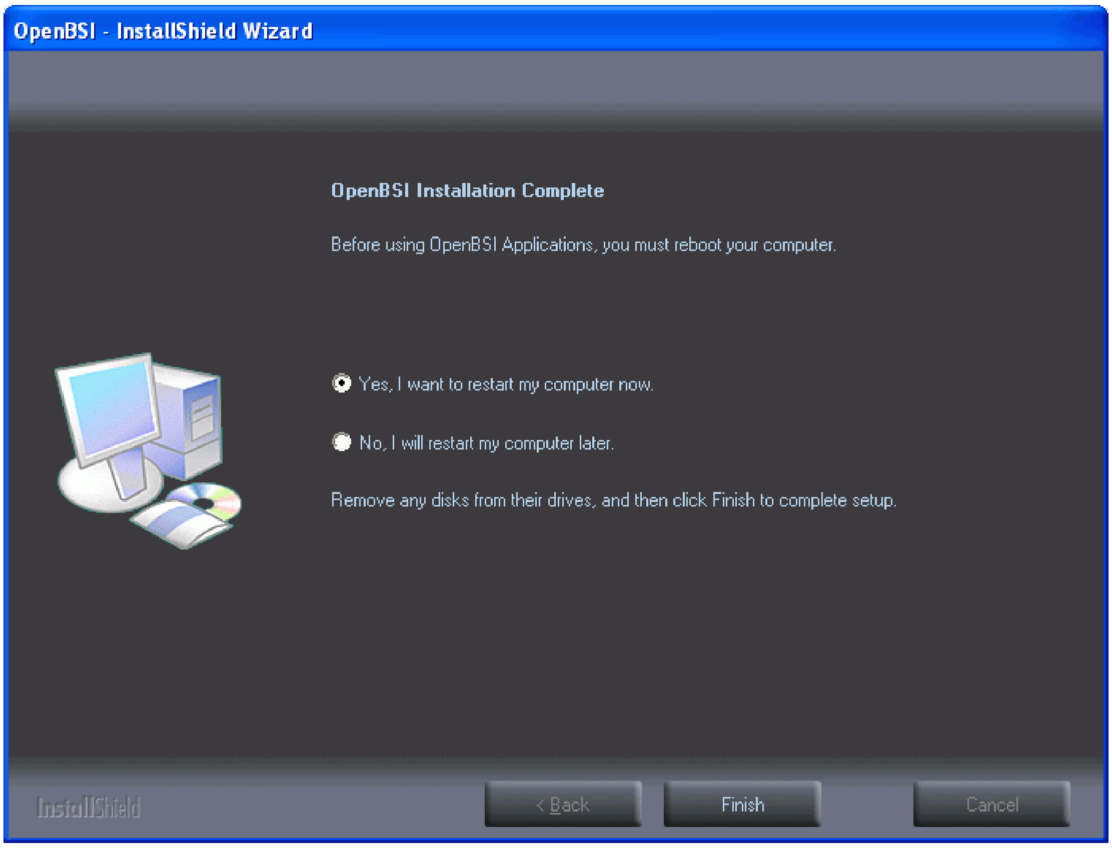
Enter Text

Please enter information in the field below.

Enter the computer name of the ObjectServer Database (Server)

DBServer

7. The Installshield Wizard begin installing the OpenBSI and ObjectServer products that you selected. The wizard displays informative messages as the installation proceeds, but you do not need to do anything until the final page (OpenBSI Installation Complete) of the wizard displays.



OpenBSI Installation Complete

Before using OpenBSI Applications, you must reboot your computer.

[✓ Yes, I want to restart my computer now.]

No, I will restart my computer later.

Remove any disks from their drives, and then click Finish to complete setup.

Finalizing the installation requires that you restart the computer, so the recommended actions are to leave the **Yes, I want to restart my computer now** option selected and click **Finish**.

However, if you need to save any work or do anything else first, select the **No, I will restart my computer later** option and click **Finish**. You will need to restart the computer after you finish your tasks.

When the computer reboots, you can begin coding the control strategy programs for controllers. That is the subject for the next chapter.

[This page intentionally left blank.]

Chapter 3 – Basic Bristol Controller Preparation

This chapter briefly explains how you ensure that the control strategy files you create and download to the Bristol controllers work with ObjectServer.

Bristol RTU/Controllers and their Control Strategy Files

As we have seen, Bristol controllers come in two basic forms: the older NW3000 Series (that includes the DPC 3330, DPC 3335, RTU 3305, or RTU 3310) and the newer ControlWave series (that includes the ControlWave, ControlWave MICRO, and ControlWave LP, among others).

These controllers execute a pre-defined program (called a **control strategy)** which reads data from process instrumentation (flow meters, pressure transmitters, etc.), performs calculations based on the data collected, and sends out commands to instrumentation (switches, valves, etc.)

Are RTUs and Remote Controllers the same?

If you study the theory of control systems there are slight differences between these terms, but for our purposes, they all mean the same thing. **RTU** is just an abbreviation for the term **remote terminal unit**, which is basically the same thing as a r**emote process controller**. Sometimes people will just use the term **femote**. You might also hear people say **DPC** (**distributed process controller**.) You may even hear someone say **node,** which is a reference to the fact that a controller can serve as part of a network.

All of these terms refer to a small-computerized device located at a remote site that collects data from instrumentation and performs control operations based on the data it collects.

NW3000 Series Controllers

Network 3000 series Controller

You program the NW3000 Series devices with a language called **ACCOL II**, using the **ACCOL Workbench** program.

You download the resulting file (called the ACCOL “load”) into the computer. The primary structure for storing an individual data value (pump status, flow reading, etc.) in the ACCOL load is called an **ACCOL signal**.

ControlWave Controllers

For ControlWave-series controllers, you write the control strategy in any of five **IEC 61131-3** languages, using a program called **ControlWave Designer.** (IEC 61131-3 is an international standard for process control programming languages.)

ControlWave series Controller

You download the resulting file (called a **ControlWave project**) into the ControlWave controller. The primary structure for storing a data value (temperature reading, valve position, etc.) in the ControlWave project is called a **variable**, and is equivalent to a "signal" in a 33xx controller.

What's "downloading"?

Downloading means transferring programs and/or data from one device to another. In this case, we're downloading a file from the computer to the controller.

NOTE: ObjectServer makes no distinction between the terms "signal" and "variable"; they are all referred to as "signals". Other manufacturers may use the word:"tag" to refer to the same thing. These three terms—tag, signal and variable—are often used interchangeably

The principles for ensuring that ObjectServer can collect signal data are the same no matter what language is used.

Marking Signals for Collection by ObjectServer

Important - For ObjectServer to collect and store a signal (in an ACCOL load) or a variable (in a ControlWave project) in its database, that signal must fall into one of the following categories.

- **Alarm** (**ACCOL and ControlWave loads):** Any signal or variable configured as an alarm is automatically collected when it enters an alarm state or returns to a normal state. Normally these are signals important for alerting operators, such as when a pre-determined condition (like a maximum tank level) is exceeded. Analog alarm signals are also automatically collected via polling, unless you configure them as RBE within the RTU.

- **Global (ACCOL loads only):** A global signal in ACCOL is marked for collection via Polled List collection. ObjectServer simply collects these signals at a fixed rate (such as once per minute). You configure this rate once for each RTU, so all polled signals for a particular RTU are collected at the same frequency or rate.

NOTE

A signal marked as “global” in ACCOL is included in the .SIG file. This means that ObjectServe marks it for automatic polling collection. This is **not** the case for ControlWave variables. ControlWave variables marked as “global” are simply marked as being *available* to all program organization units (POUs) and are not automatically marked for ObjectServer collection. To mark a ControlWave variable for ObjectServer collection, be sure the check **both** its PDD **and** OPC parameters.

- **RBE** (**ACCOL and ControlWave loads):** The controller sends a report by exception (RBE) signal to ObjectServer whenever a signal value changes by more than a pre-configured amount (called the ”deadband”). You can configure a deadband of zero, which ensures that **all** signals go to the ObjectServer.

- **PDD (ControlWave loads only):** Marking a variable as PDD (Process Data Directory) and OPC (OLE for Process Control) means that the variable is marked for polled collection by ObjectServer and is also viewable using OpenBSI’s Dataview utility.

- **OPC (ControlWave loads only):** Marking a variable as OPC (OLE for Process Control) and PDD (Process Data Directory) means that the variable is marked for polled collection by ObjectServer, and is also viewable using OpenBSI’s Dataview utility.

Following are the steps within ACCOL Workbench and ControlWave Designer that you need to complete to identify the data (signals or variables) ObjectServer collects.

NOTE

This section does not tell you everything you need to know when using ACCOL Workbench or ControlWave Designer. Instead, it just highlights the parts which are related to whether a signal or variable becomes a part of the ObjectServer database. It doesn’t give detailed instructions. If you need help using ACCOL Workbench, see the *ACCOL Workbench User Manual* (document# D4051). If you need help using ControlWave Designer, see the *Getting Started with ControlWave Designer Manual* (document# D5085).

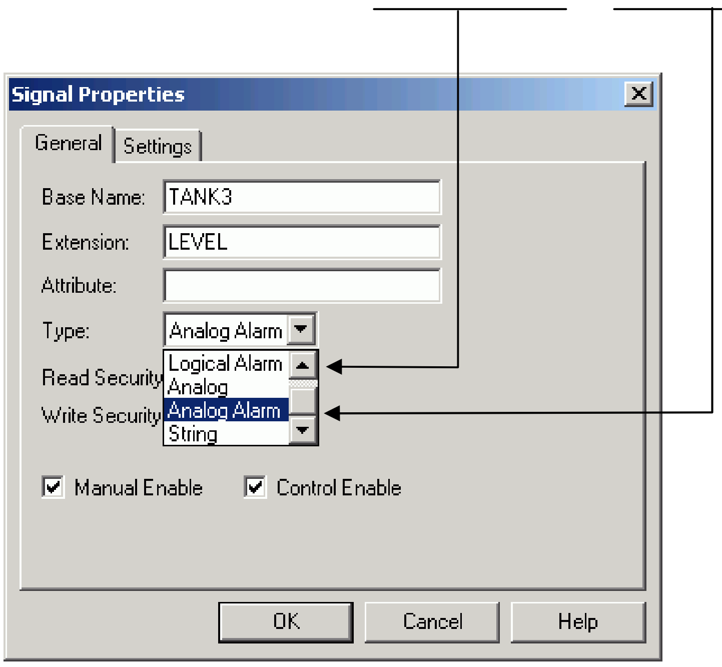
Using ACCOL Workbench

First, let's use ACCOL Workbench to see how to create signals ObjectServer collects. Use ACCOL Workbench to create control strategy files for NW3000 Series controllers.

1. Creating an Alarm Signal

On the General tab of the Signal Properties dialog box for the signal, choose either Logical Alarm or Analog Alarm as the signal's **Type**.

The Type should be either Logical Alarm or Analog Alarm.



Signal Properties

General | Settings

Base Name: TANK3

Extension: LEVEL

Attribute:

Type: Analog Alarm

Logical Alarm
Analog
Analog Alarm
String

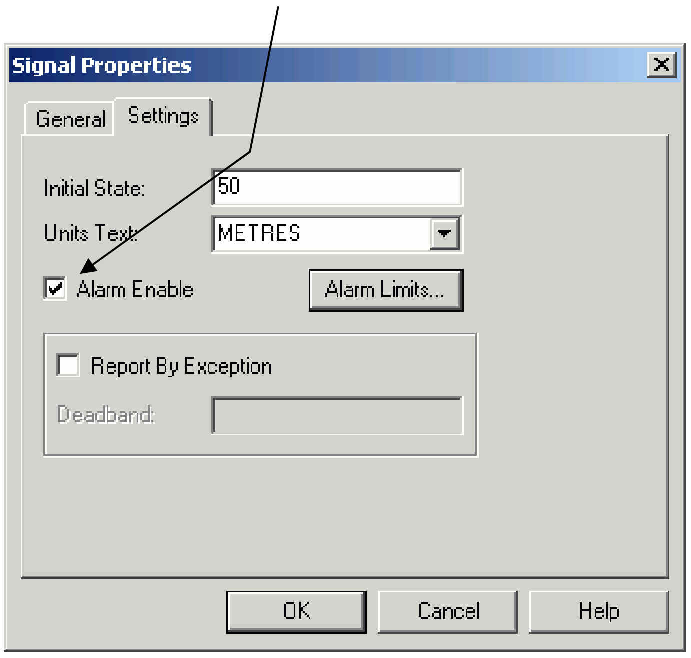
Read Security

Write Security

☑ Manual Enable ☑ Control Enable

On the Settings tab of the Signal Properties dialog box for the signal, select **Alarm Enable**.

Select the Alarm Enable check box.



Signal Properties

General | Settings

Initial State: 50

Units Text: METRES

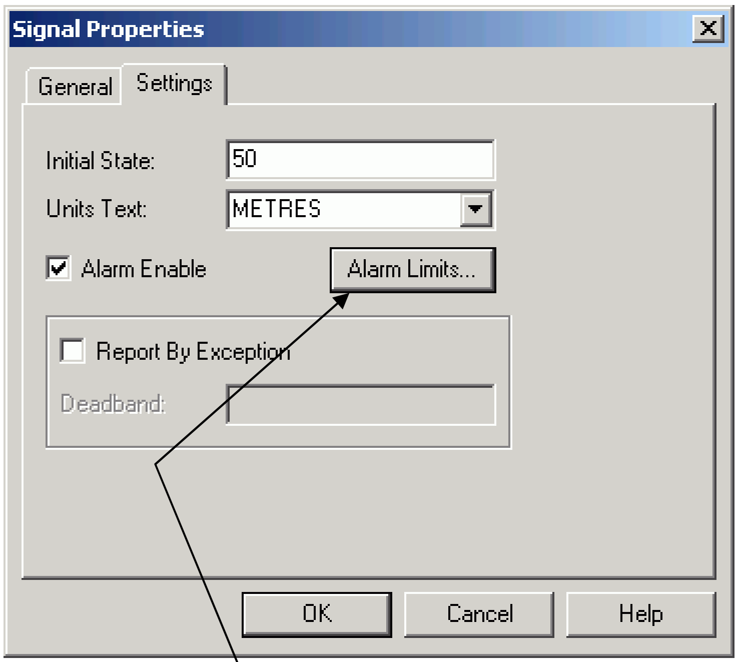
☑ Alarm Enable

☐ Report By Exception

Deadband:

For analog alarm signals, you must also specify at least one alarm limit. You should also specify an alarm deadband.

Click **Alarm Limits** to begin this configuration.


Signal Properties

General | Settings

Initial State: 50

Units Text: METRES

☑ Alarm Enable

Alarm Limits...

☐ Report By Exception

Deadband:

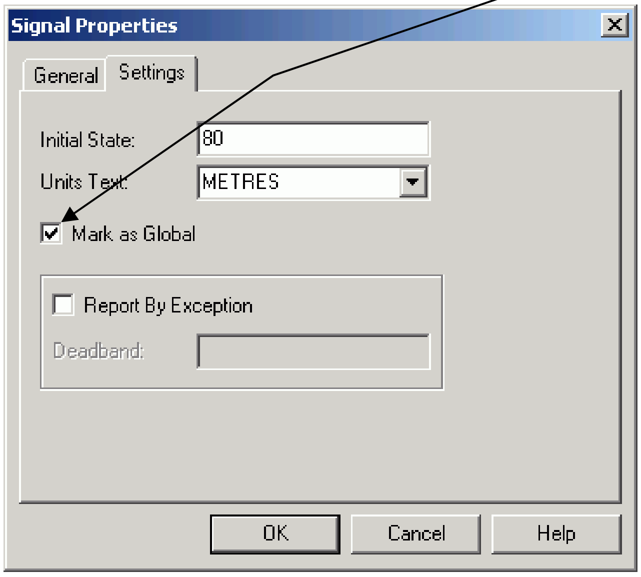
OK | Cancel | Help

Click **Alarm Limits** to configure one or more alarm limits for analog alarm signals.

2. Creating a Global Signal

To make a signal global, select the **Mark as Global** check box.

On the Settings tab of the Signal Properties dialog box you should select the **Mark as Global** check box.


Signal Properties

General | Settings

Initial State: 80

Units Text: METRES

☑ Mark as Global

☐ Report By Exception

Deadband:

OK | Cancel | Help

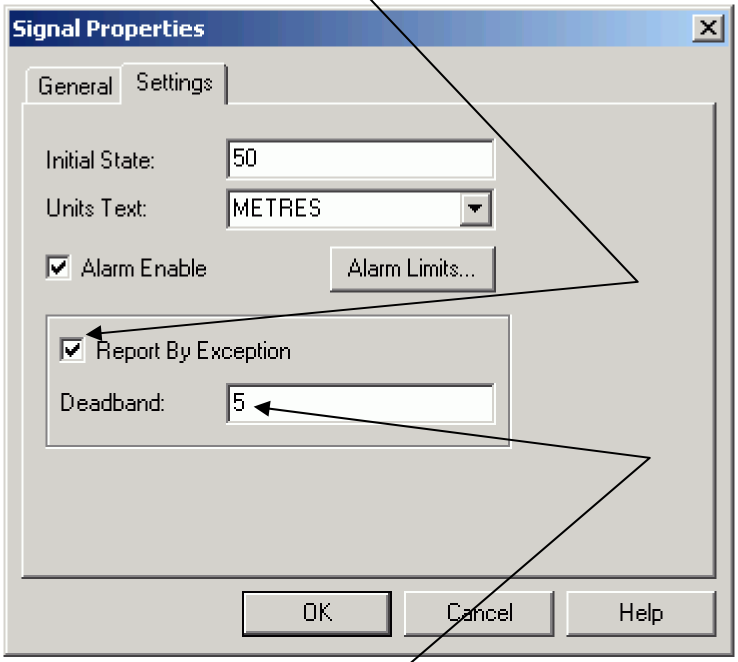
3. Creating an RBE Signal

On the Settings tab of the Signal Properties dialog box for the signal, select the **Report By Exception** check box.

If this is an analog signal, you **must** enter a deadband. **Do not** leave the deadband field blank, or you will overload your communications network!

You must also add an RBE module to the RTU load, typically in task zero. This is discussed in the *RBE* section of the *ACCOL II Reference Manual* (document# D4044).

Select the **Report By Exception** check box to ensure this is an RBE signal.



Initial State: 50
Units Text: METRES
☑ Alarm Enable
☑ Report By Exception
Deadband: 5

You must specify a deadband if this is an analog signal.

Saving, Compiling, and Downloading the ACCOL Load

- When you have finished editing in ACCOL Workbench, save the ACCOL source file. Click **File → Save** or click the Save icon.
- You must then build the ACCOL load (*.ACL) file from the ACCOL source file using the ACCOL Workbench's Build command. Click **Actions → Build** or click the Build icon.
- When the ACCOL load builds successfully without errors, download it into the Network 3000 controller using the Open BSI Downloader. You access the Downloader utility from within ACCOL Workbench. Click **Actions → Download** or click on the Download icon. For full instructions on downloading, see Chapter 7 of the *Open BSI Utilities Manual* (document# D5081).

WARNING! Test *Before* Downloading

You should always test your control strategy file (ACCOL load or ControlWave project) *before* using it in a live running plant or process. This is especially true when you're first learning things. We strongly recommend you do this using a controller that is currently disconnected from the process, or for which manual overrides are ready and staffed should something go wrong. This is **really** important, because if you download an untested control strategy, you could potentially lose control of your plant or process due to a programming logic error and *you could damage your equipment or—even worse—hurt somebody!* So if you want to avoid flooding something, blowing something up, or dealing with personal injury lawyers, **always** *test before you download to a live RTU connected to a plant!*

Using ControlWave Designer

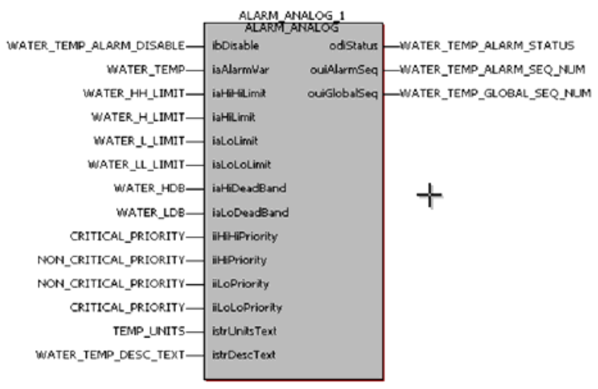
ControlWave Designer enables you to create variables (signals) that ObjectServer collects. Use ControlWave Designer to create control strategy files for ControlWave controllers.

1. Creating an Alarm Variable

In your ControlWave Designer project, each variable which you want to serve as an alarm must have *its own alarm function block* configured*.* The alarm function blocks available are:

ALARM_ANALOG, ALARM_STATE, ALARM_LOGICAL_ON and ALARM_LOGICAL_OFF.

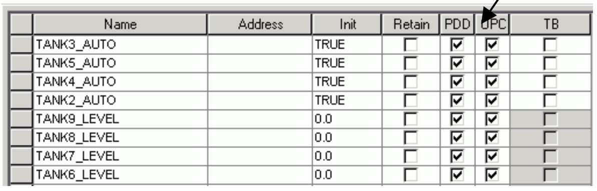
Descriptions on how to configure these function blocks are included in the ControlWave Designer online help and in the *ControlWave Designer Programmer's Handbook* (document# D5125).



An alarm condition can only be detected at the time its Alarm function block is executed.

You must also mark the alarm variable for **PDD** and **OPC** when you create it. See the variable grid worksheet for the program organization uint (POU) to the right.

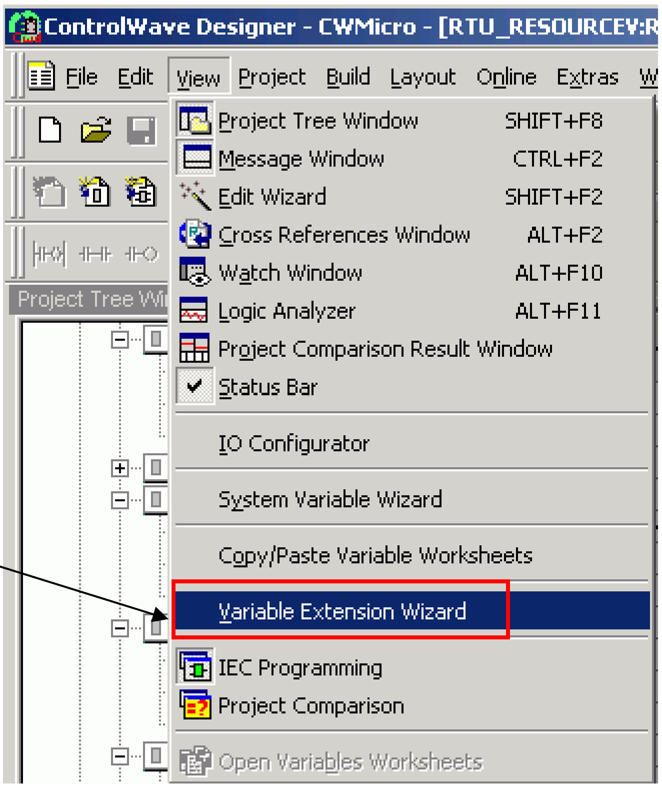
Select the **PDD** and **OPC** check boxes if you also want the variable to be collected by polling.



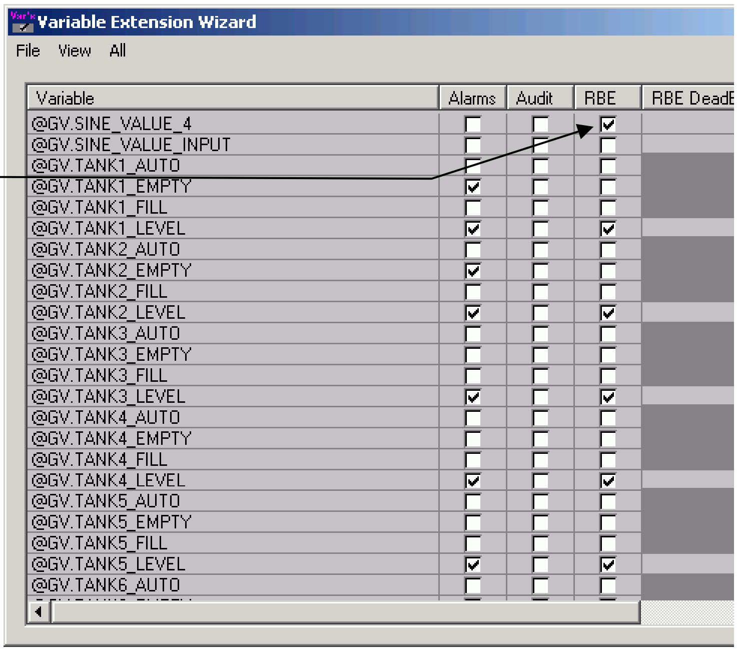
Name	Address	Init	Retain	PDD	OPC	TB
TANK3_AUTO		TRUE	☐	☑	☑	☐
TANK5_AUTO		TRUE	☐	☑	☑	☐
TANK4_AUTO		TRUE	☐	☑	☑	☐
TANK2_AUTO		TRUE	☐	☑	☑	☐
TANK9_LEVEL		0.0	☐	☑	☑	☐
TANK8_LEVEL		0.0	☐	☑	☑	☐
TANK7_LEVEL		0.0	☐	☑	☑	☐
TANK6_LEVEL		0.0	☐	☑	☑	☐

2. Creating an RBE Variable

To designate a variable for RBE collection in a ControlWave project, select the **View** menu option and then the Variable Extension Wizard option..

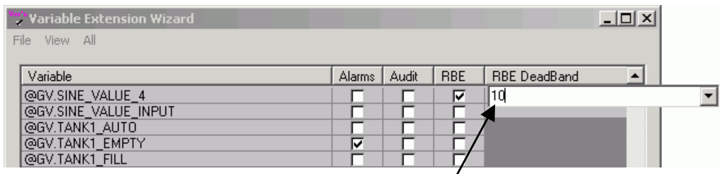


When the Variable Extension Wizard dialog appears, select the **RBE** check box for each variable you want collected via RBE.



Variable	Alarms	Audit	RBE	RBE Dead
@GV.SINE_VALUE_4	☐	☐	✓	
@GV.SINE_VALUE_INPUT	☐	☐	☐	
@GV.TANK1_AUTO	☐	☐	☐	
@GV.TANK1_EMPTY	✓	☐	☐	
@GV.TANK1_FILL	☐	☐	☐	
@GV.TANK1_LEVEL	✓	☐	✓	
@GV.TANK2_AUTO	☐	☐	☐	
@GV.TANK2_EMPTY	✓	☐	☐	
@GV.TANK2_FILL	☐	☐	☐	
@GV.TANK2_LEVEL	✓	☐	✓	
@GV.TANK3_AUTO	☐	☐	☐	
@GV.TANK3_EMPTY	☐	☐	☐	
@GV.TANK3_FILL	☐	☐	☐	
@GV.TANK3_LEVEL	✓	☐	✓	
@GV.TANK4_AUTO	☐	☐	☐	
@GV.TANK4_EMPTY	☐	☐	☐	
@GV.TANK4_FILL	☐	☐	☐	
@GV.TANK4_LEVEL	✓	☐	✓	
@GV.TANK5_AUTO	☐	☐	☐	
@GV.TANK5_EMPTY	☐	☐	☐	
@GV.TANK5_FILL	☐	☐	☐	
@GV.TANK5_LEVEL	✓	☐	✓	
@GV.TANK6_AUTO	☐	☐	☐	

For non-BOOL variables (REAL, INT, etc.), you would also specify a deadband.

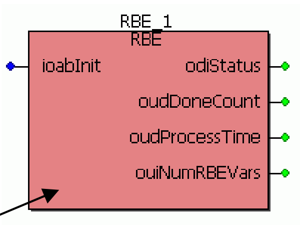


Variable	Alarms	Audit	RBE	RBE DeadBand
@GV.SINE_VALUE_4	☐	☐	✓	10
@GV.SINE_VALUE_INPUT	☐	☐	☐	
@GV.TANK1_AUTO	☐	☐	☐	
@GV.TANK1_EMPTY	✓	☐	☐	
@GV.TANK1_FILL	☐	☐	☐	

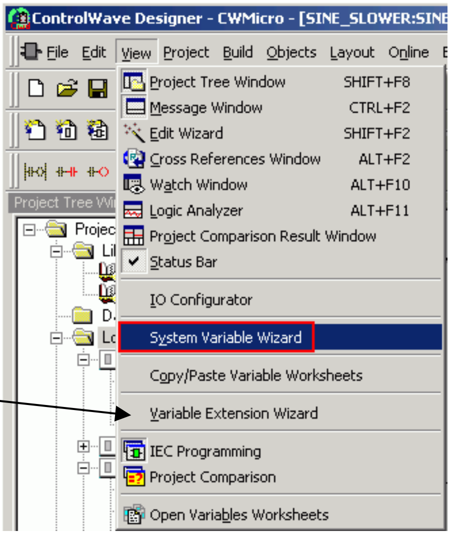
This information is saved in a file called "RBE.INI' that you download to the ControlWave with the bootproject.

- If you do not specify a deadband, this could strain your network bandwidth when the value changes rapidly over a period of time.
- If you set the deadband to zero, this may produce the same effect as having it not set a deadband at all.
- If the deadband is too small (too narrow a range), you can again strain your network bandwidth when a value changes rapidly over a period of time.
- When setting deadbands, keep in mind the total range (span) of values that the variable can have (such as 0-100), as well as the possible speed and duration of changes the value may experience. For instance, a variable with a total range between 0 and 100 may experience a maximum rate of change of 10 units per minute. So, you may feel that a deadband of 5 is sufficient to keep you informed without flooding the network.

You must also add and configure a **POU** containing the **RBE** function block.



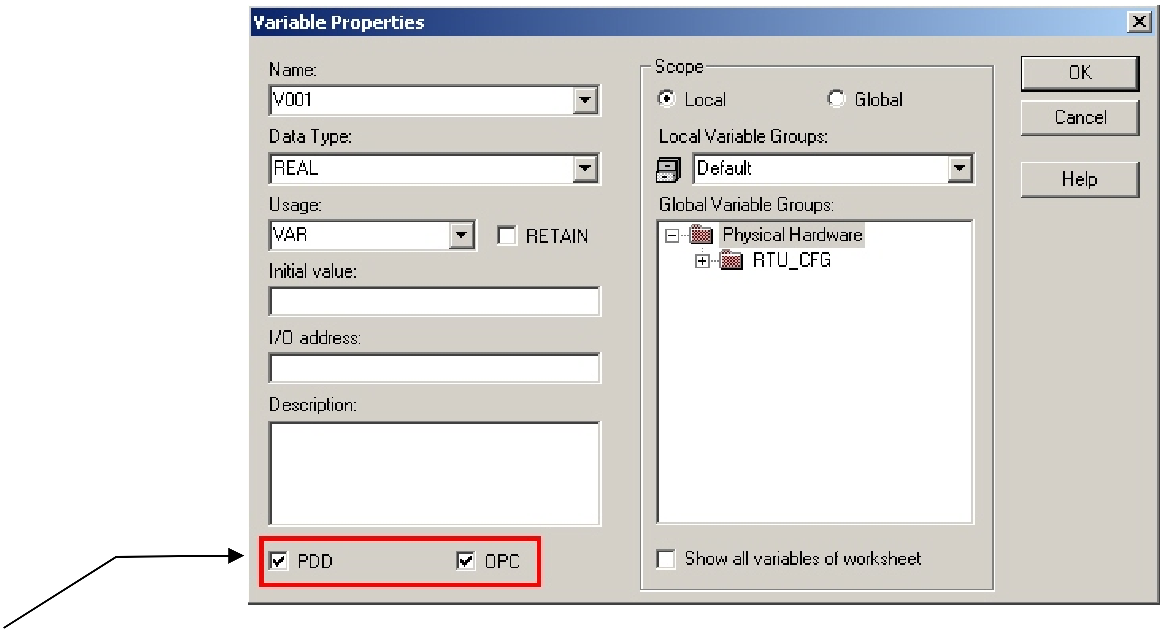
Finally, configure the appropriate RBE system variables depending upon whether RBE is running on a serial or IP port. Use the **Variable Extension Wizard** to do this.


ControlWave Designer - CWMicro - [SINE_SLOWER:SINE
File Edit View Project Build Objects Layout Online
Project Tree Window SHIFT+F8
Message Window CTRL+F2
Edit Wizard SHIFT+F2
Cross References Window ALT+F2
Watch Window ALT+F10
Logic Analyzer ALT+F11
Project Comparison Result Window
Status Bar
IO Configurator
System Variable Wizard
Copy/Paste Variable Worksheets
Variable Extension Wizard
IEC Programming
Project Comparison
Open Variables Worksheets
Project Tree Wi
Projec

For details on all these subjects, please see the online help in ControlWave Designer.

3. Marking Local Variables for Collection

Mark local variables for collection when you create them.



Variable Properties

Name: V001

Data Type: REAL

Usage: VAR ☐ RETAIN

Initial value:

I/O address:

Description:

☑ PDD ☑ OPC

Scope: [✓ Local] Global

Local Variable Groups: Default

Global Variable Groups:
- Physical Hardware
  - RTU_CFG

☐ Show all variables of worksheet

When you create a local variable, you can specify that you want ObjectServer to collet the variable by selecting the **PDD** and **OPC** check boxes on this dialog.

4. Manually Marking Global Variables for Collection

ControlWave Designer global variables are variables that all the POUs in the project can access. You designate variables as global when you create them. Marking these global variables as **PDD** and **OPC** guarantees that ObjectServer collects these signals during polling at regular intervals. (If you don't know what a POU is, you probably need to review the documentation accompanying ControlWave Designer.)

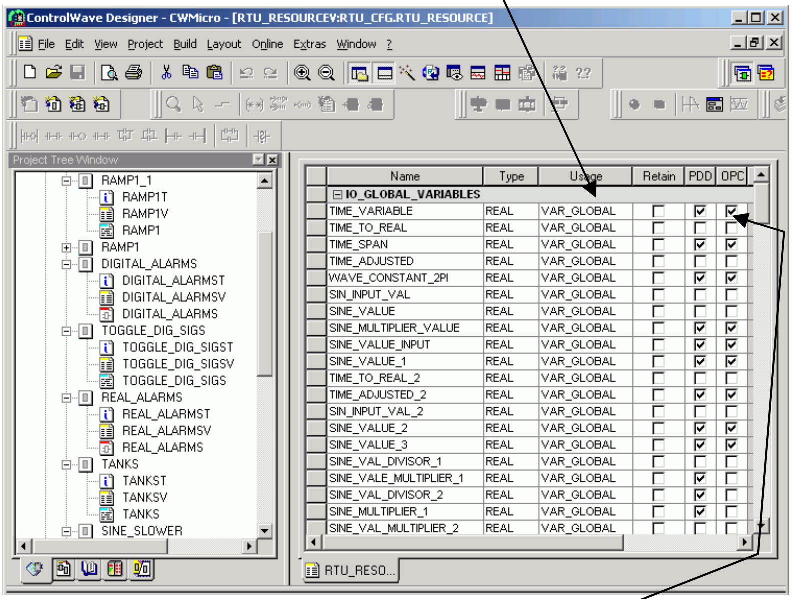
In your ControlWave Designer project, we recommend that *you designate only I/O variables as global*. Configure your I/O variables in the ControlWave Designer's I/O Configurator *first*. You can create them as global variables, and then use the RTU_RESOURCEV worksheet to verify they are marked as **PDD** and **OPC**.

The RTU_RESOURCEV Global worksheet (with VAR_GLOBAL selected in the **Usage** column) lists all designated global variables in ControlWave Designer.

If you need to collect only specific global variables, you must manually check their **PDD** and **OPC** options (shown on the right).

Otherwise, use the Device Resource Settings dialog to specify that all global variables should have their **PDD** and **OPC** options selected See *5. Setting All Global Variables for* for more information.

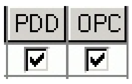
VAR_GLOBAL in the Usage field identifes a global variable:



Name	Type	Usage	Retain	PDD	OPC
⊟ **IO_GLOBAL_VARIABLES**					
TIME_VARIABLE	REAL	VAR_GLOBAL	☐	☑	☑
TIME_TO_REAL	REAL	VAR_GLOBAL	☐	☐	☐
TIME_SPAN	REAL	VAR_GLOBAL	☐	☑	☑
TIME_ADJUSTED	REAL	VAR_GLOBAL	☐	☐	☐
WAVE_CONSTANT_2PI	REAL	VAR_GLOBAL	☐	☑	☑
SIN_INPUT_VAL	REAL	VAR_GLOBAL	☐	☐	☐
SINE_VALUE	REAL	VAR_GLOBAL	☐	☐	☐
SINE_MULTIPLIER_VALUE	REAL	VAR_GLOBAL	☐	☑	☑
SINE_VALUE_INPUT	REAL	VAR_GLOBAL	☐	☑	☑
SINE_VALUE_1	REAL	VAR_GLOBAL	☐	☑	☑
TIME_TO_REAL_2	REAL	VAR_GLOBAL	☐	☐	☐
TIME_ADJUSTED_2	REAL	VAR_GLOBAL	☐	☑	☑
SIN_INPUT_VAL_2	REAL	VAR_GLOBAL	☐	☐	☐
SINE_VALUE_2	REAL	VAR_GLOBAL	☐	☑	☑
SINE_VALUE_3	REAL	VAR_GLOBAL	☐	☑	☑
SINE_VAL_DIVISOR_1	REAL	VAR_GLOBAL	☐	☐	☐
SINE_VALE_MULTIPLIER_1	REAL	VAR_GLOBAL	☐	☑	☐
SINE_VAL_DIVISOR_2	REAL	VAR_GLOBAL	☐	☑	☐
SINE_MULTIPLIER_1	REAL	VAR_GLOBAL	☐	☑	☐
SINE_VAL_MULTIPLIER_2	REAL	VAR_GLOBAL	☐	☐	☐

If you select the Marked variables option on the device’s Resource Settings dialog (see heading 5 below) **only** global variables which also have **PDD** and **OPC** selected appear in the ObjectServer database.

5. Setting All Global Variables for Collection

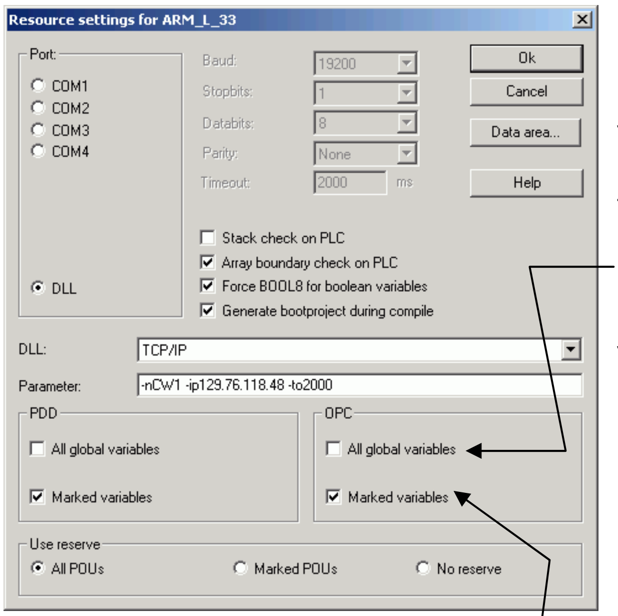


Any variable which you want to include in the ObjectServer database must have its own specific PDD and OPC parameter check box selected (as described previously under *4. Manually Marking Global Variables for Collection).*

In addition, you must specify (on the device’s Resource Settings dialog) whether you want ObjectServer to collect only those global variables that you have specifically marked as **PPD** and **OPC** or whether you want ControlWave Designer to mark all global variables for collection by ObjectServer. If you choose this latter option ControlWave Designer automatically marks **all** global variables as **PDD** and **OPC**.

The practice we recommend is that you first explicitly mark the global variables that you want ObjectServer to collect as **PDD** and **OPC.** **T**hen select the **Marked variables** check boxes on this dialog so that **only** those variables which you have explicitly marked for collection are included in the ObjectServer database.

However, if you want **all** global variables to be part of the ObjectServer database, select the **All global variables** check boxes in the PDD and OPC sections on this dialog. This *automatically* marks all global variables for ObjectServer's polled collection.



Select the **All global variables** check boxes to ensure that ObjectServer collects all global variables marked as **PDD** and **OPC**.

Select the **Marked variables** check boxes to ensure that ObjectServer only collects the signals you have explicitly marked as **PDD** and **OPC**.

Compiling and Downloading the ControlWave Project

Finally, compile the ControlWave project. Click **Build → Make** or click the Make icon.

When the ControlWave project compiles successfully, you must download it into the ControlWave controller. Please heed the warning about downloading on page 3-7. For details on performing a download into the ControlWave, see the *Downloading* section of the *ControlWave Designer Programmer's Handbook* (document# D5125).

Chapter 4 – ObjectServer Preparation

This chapter explains how to prepare ObjectServer so it can run and begin collecting data from your ControlWave and NW3000 controllers.

Note: The terms "controller", "RTU", and "device" are interchangeable and, for our purposes all mean essentially the same thing.

Controller = RTU = Device.

Before you take the steps described in this chapter, you must first complete the following pre-requisite activities:

1. Configured your controller network using the Netview application.
2. Created your control strategy files and marked all signals or variables for ObjectServer collection (as described in *Chapter 3*).
3. Downloaded your control strategy files to all your controllers (as described in *Chapter 3*).
4. Installed ObjectServer.

With those pre-requisites accomplished, these are the steps you need to accomplish to get the ObjectServer database functioning.

Step 1: Start OpenBSI

Start OpenBSI (which you have previously installed and configured). OpenBSI must be running for the ObjectServer database to function correctly. How you start OpenBSI depends upon your particular system configuration and requirements, but one way is to run NetView:

Start → Programs → OpenBSI Tools → NetView

For further information, see the OpenBSI documentation (available on the OpenBSI CD).

Step 2: Start ObjectServer Database

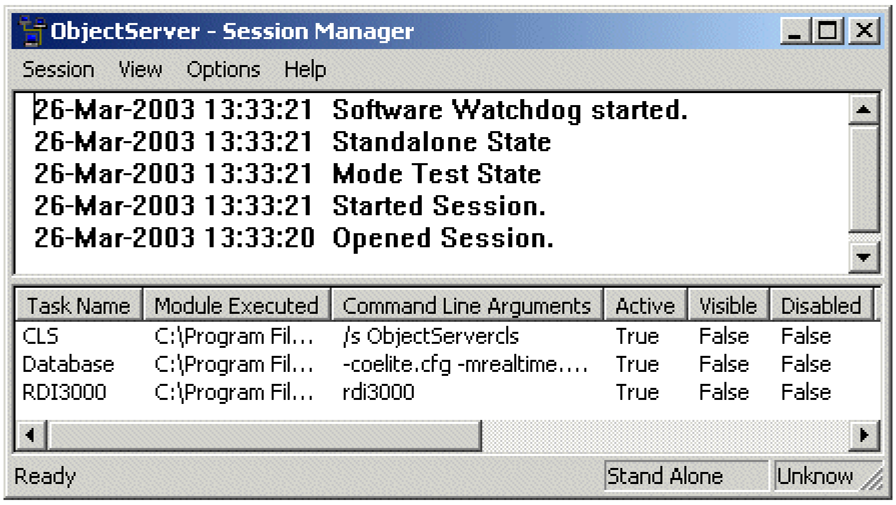
By default, the ObjectServer database is configured to start automatically when the PC reboots. You can also start it manually using the Session Manager.

Start → Programs → OpenBSI Tools → ObjectServer → ObjectServer Session

Note: The program folder shown is the default; you may have changed the folder to suit particular requirements during installation.

This icon displays in the System Tray when the ObjectServer database is running.

Double-clicking on this System Tray icon displays the Session Manager window:



```
26-Mar-2003 13:33:21  Software Watchdog started.
26-Mar-2003 13:33:21  Standalone State
26-Mar-2003 13:33:21  Mode Test State
26-Mar-2003 13:33:21  Started Session.
26-Mar-2003 13:33:20  Opened Session.
```

Task Name	Module Executed	Command Line Arguments	Active	Visible	Disabled
CLS	C:\Program Fil...	/s ObjectServercls	True	False	False
Database	C:\Program Fil...	-coelite.cfg -mrealtime....	True	False	False
RDI3000	C:\Program Fil...	rdi3000	True	False	False

Ready | Stand Alone | Unknow

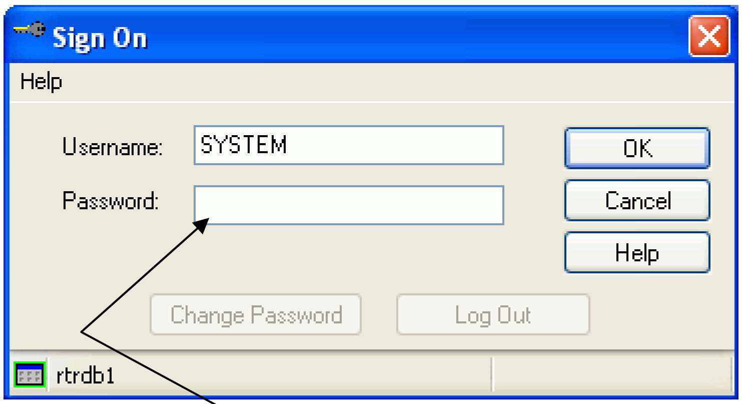
Step 3: Import Device and Signal Information from OpenBSI

Open up the Toolbox as follows:

Start → Programs → OpenBSI Tools → ObjectServer → ToolBox

If you are not currently logged on, then the Sign On dialog displays:

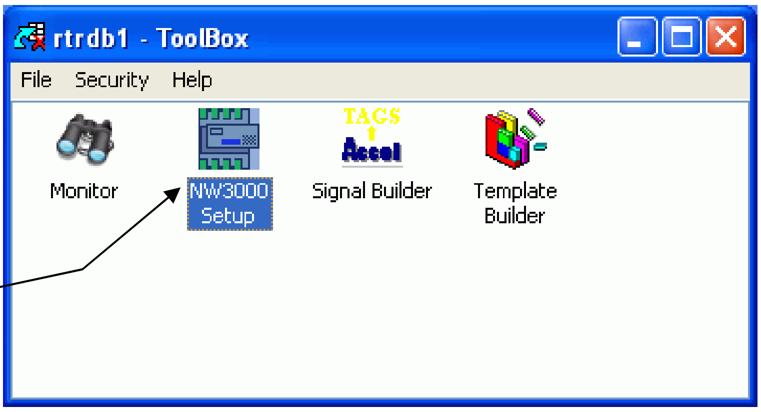
When you first install OpenBSI, you can sign on with a username of SYSTEM and a password of SYSTEM (the password is case sensitive).
Note: We recommend you change this default password as soon as possible after installation to something more secure.



You need to sign on as a user with administrative privileges in order to perform this step. Type the password here.

Once you sign on, the Toolbox displays. Note the NW3000 Setup Tool icon in the Toolbox window in the screenshot to the right.

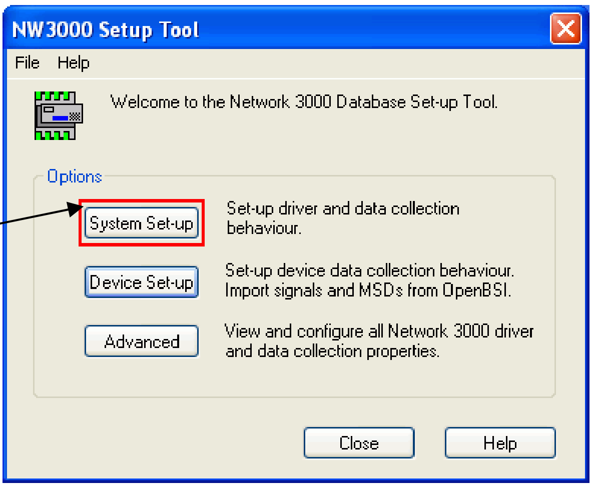
Double-click the icon.



NW3000 Setup Tool

We now need to run two wizards to set up the ObjectServer RDI3000 communication driver.

To begin the first wizard, click **System Set-up**.


NW3000 Setup Tool

File Help

Welcome to the Network 3000 Database Set-up Tool.

Options

System Set-up — Set-up driver and data collection behaviour.

Device Set-up — Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

Advanced — View and configure all Network 3000 driver and data collection properties.

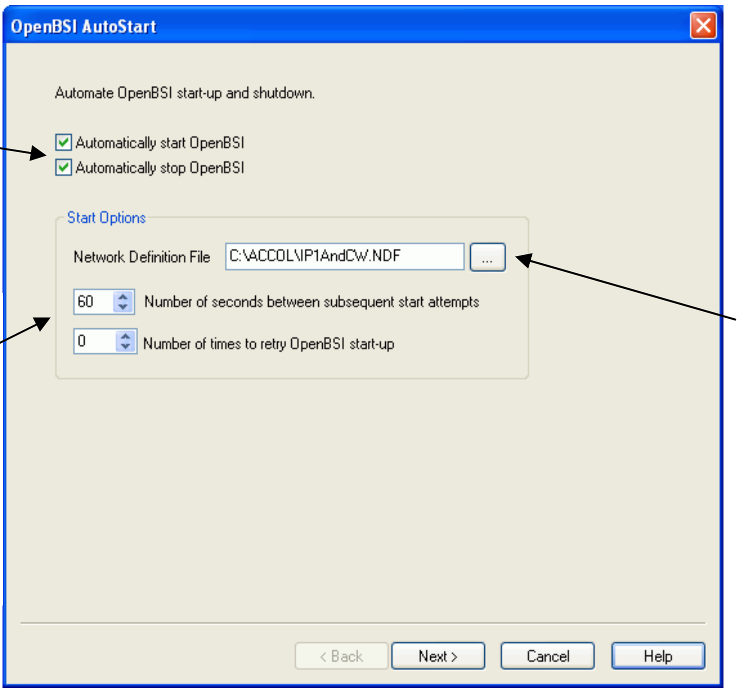
The NW3000 System Set-up Wizard

This wizard guides you through all of the steps required to set up basic ObjectServer data collection behaviour. This section describes the options on the individual wizard pages. Additionally, each page has a Help button which provides you with online help.

OpenBSI AutoStart Page

Use this page to define some basic ObjectServer settings.

Check these boxes to automatically start and stop OpenBSI when ObjectServer starts and stops.


OpenBSI AutoStart

Automate OpenBSI start-up and shutdown.

- [✓] Automatically start OpenBSI
- [✓] Automatically stop OpenBSI

Start Options

Network Definition File: C:\ACCOL\IP1AndCW.NDF

60 Number of seconds between subsequent start attempts

0 Number of times to retry OpenBSI start-up

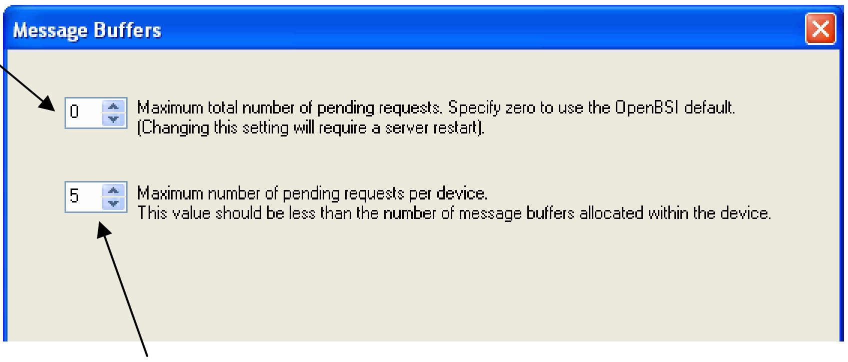
You can change these default settings if you need to. They are fine for most purposes.

Browse and identify your Network definition file using this button.

Message Buffers Page

Use this page to specify the maximum number of messages issued to your RTU(s) before waiting for a reply. A small number slows the system down significantly, especially if you are collecting a lot of data from an individual RTU. However it is very important that the number doesn't exceed the number of buffers available within the RTU **or** within OpenBSI.

If you do not choose the default value, the selected maximum number of pending requests should be less than the total number available within OpenBSI.



0 Maximum total number of pending requests. Specify zero to use the OpenBSI default. (Changing this setting will require a server restart).

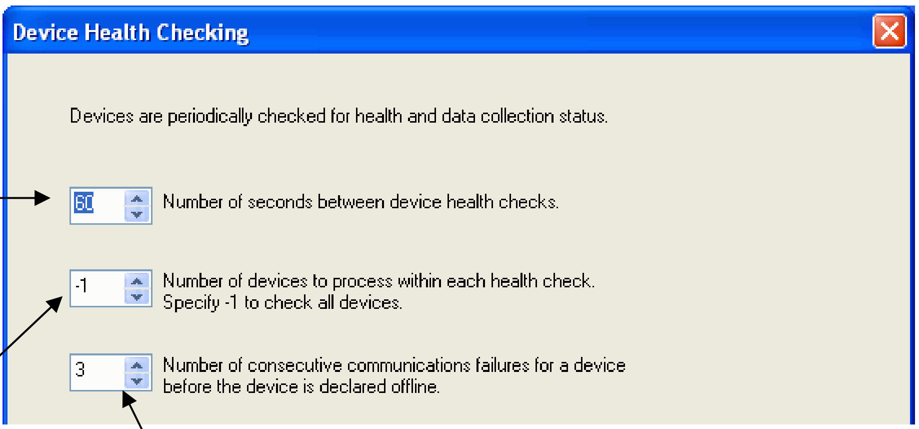
5 Maximum number of pending requests per device. This value should be less than the number of message buffers allocated within the device.

For most purposes this default value is fine. If you need to change it, bear in mind that the value should be significantly less than the number of message buffers allocated within the device. See the online help file for more information.

Device Health Checking Page

Use this page to define how often you want ObjectServer to check communications with the controllers.

Increase this number if you have lots of controllers or more than one level in the controller network.



Devices are periodically checked for health and data collection status.

60 Number of seconds between device health checks.

-1 Number of devices to process within each health check. Specify -1 to check all devices.

3 Number of consecutive communications failures for a device before the device is declared offline.

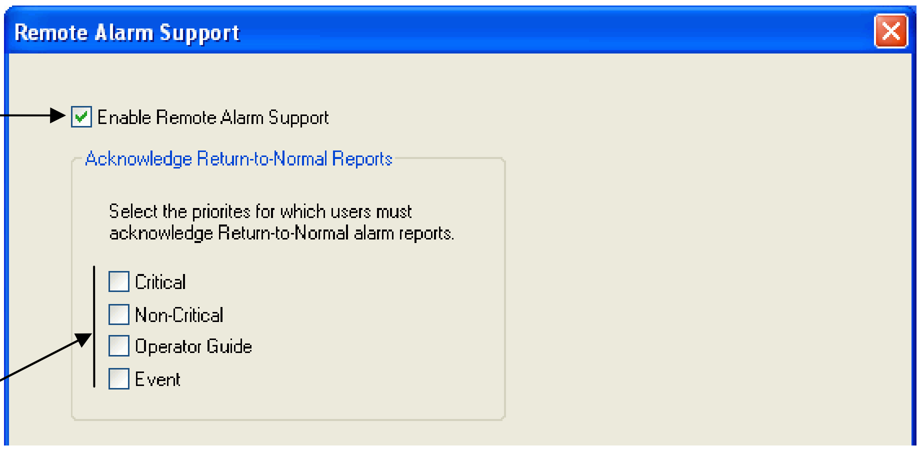
On large telemetry systems, you may need to reduce this value. This would cause a different block of devices to be processed during each health check period. Minus one (-1) signifies **all** devices.

Adjust this number depending on network loading. If you are unsure, accept the default; you can change it later.

Remote Alarm Support Page

Use this page to define collection criteria for remote alarms.

ObjectServer assumes that you want to collect alarm data from your Controllers.



[✓] Enable Remote Alarm Support

Acknowledge Return-to-Normal Reports

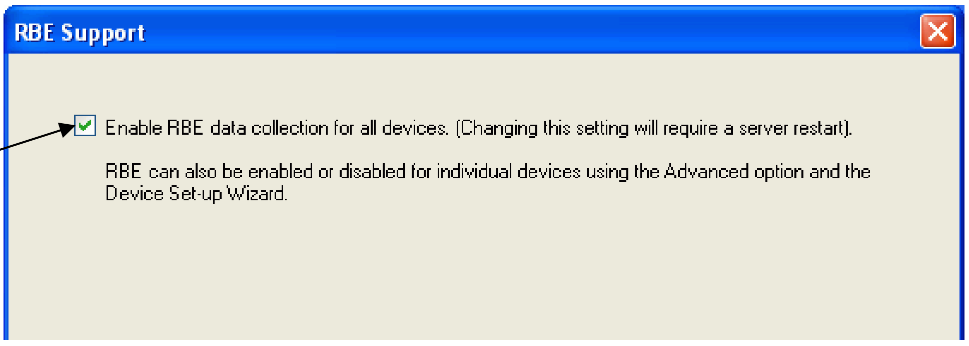
Select the priorites for which users must acknowledge Return-to-Normal alarm reports.

[ ] Critical
[ ] Non-Critical
[ ] Operator Guide
[ ] Event

Check these options **only** if you need acknowledgement for Return-to-Normal alarm reports

RBE Support Page

Use this page to define report-by-exception (RBE) collection criteria.



[✓] Enable RBE data collection for all devices. (Changing this setting will require a server restart).

RBE can also be enabled or disabled for individual devices using the Advanced option and the Device Set-up Wizard.

ObjectServer assumes that you require support for RBE Signals/Variables.

Note: For this to work, you must configure RBE within the RTU as well as here.

Polling Support Page

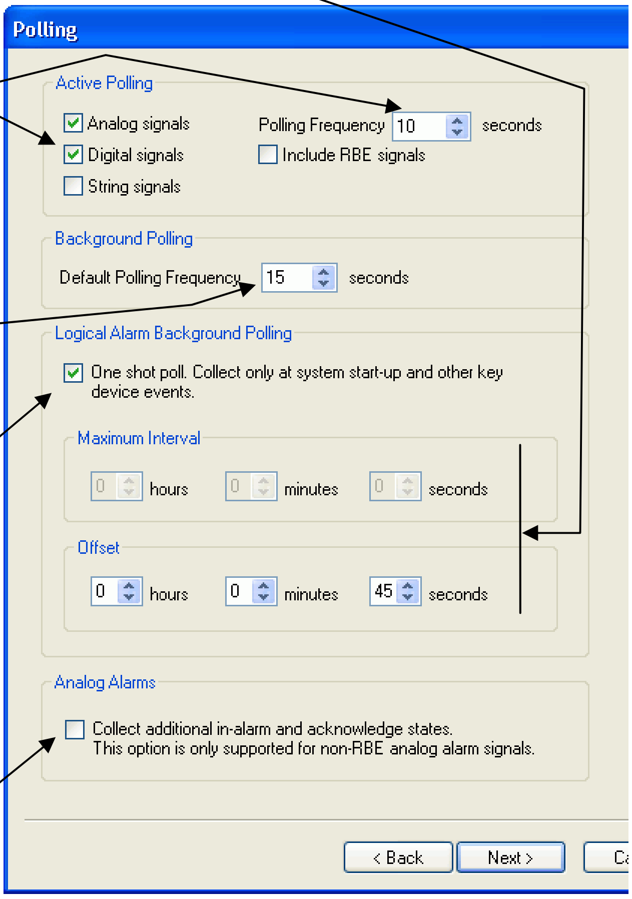
Use this page to define polling support.

If you select **One shot poll**, the wizard disables the Maximum Interval field. Leave the **One shot poll** option blank to set up a logical alarm polling frequency using the Maximum Interval and Offset fields.

These settings define Active Polling. ObjectServer temporarily polls the selected signal types using this faster frequency when data is requested by the OPC Server.

This setting defines a default background polling frequency. This can be overridden for specific devices using the Device Set-up wizard.

If checked, ObjectServer only polls in-alarm and acknowledged states for logical alarm signals when it starts up and at other key device events. We recommend this setting to ensure that logical alarm values correctly initialize.


Polling

Active Polling
- [✓] Analog signals
- [✓] Digital signals
- [ ] String signals

Polling Frequency 10 seconds
- [ ] Include RBE signals

Background Polling

Default Polling Frequency 15 seconds

Logical Alarm Background Polling
- [✓] One shot poll. Collect only at system start-up and other key device events.

Maximum Interval: 0 hours 0 minutes 0 seconds

Offset: 0 hours 0 minutes 45 seconds

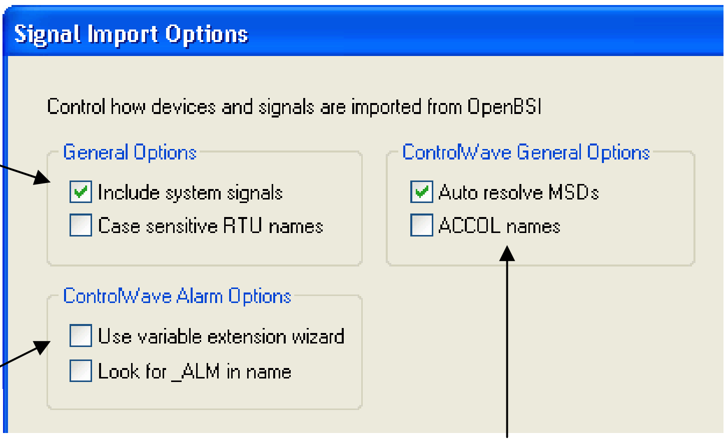
Analog Alarms
- [ ] Collect additional in-alarm and acknowledge states. This option is only supported for non-RBE analog alarm signals.

If checked, ObjectServer collects in-alarm and acknowledged states for non-RBE analog alarm signals/variables. This information is collected at the rate defined for the device to which the signal belongs and is **in addition to** the data collected when an alarm signal goes into alarm. You can define device polling rates which override the default frequency on this page using the Device Set-up wizard.

Signal Import Settings

Use this page to define how the Database Builder will import signals from your control strategy programs into the ObjectServer database.

Using these general options you can import system signals and/or use case sensitive RTU names.



Control how devices and signals are imported from OpenBSI

General Options
- [✓] Include system signals
- [ ] Case sensitive RTU names

ControlWave General Options
- [✓] Auto resolve MSDs
- [ ] ACCOL names

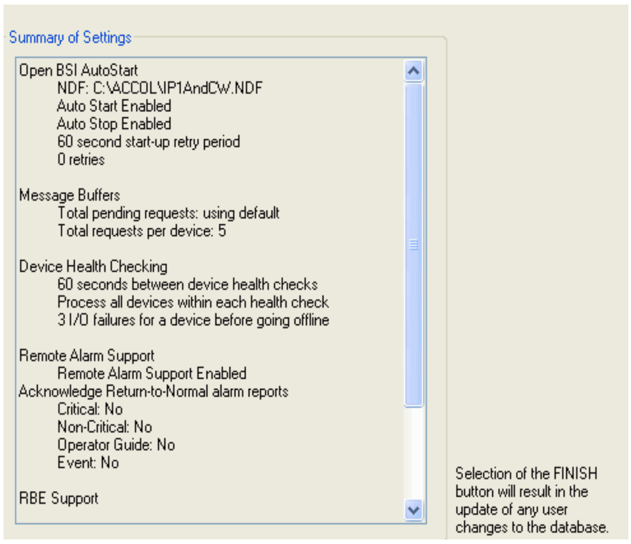
ControlWave Alarm Options
- [ ] Use variable extension wizard
- [ ] Look for _ALM in name

These options allow you to choose whether to use the “Control Wave Designer” variable extension wizard to determine your alarm signals or identify them by using “_ALM” in the name.

These options tell the Database Builder to automatically resolve MSD (Master Signal Directory) data. The MSD number specifies a signal's memory location and is used for efficient template and RBE data collection when importing signals. You can also tell it to import signals using ACCOL type names (base, extension and attribute).

Summary of Settings

This page displays the configuration details you have just defined. If you want to change anything, click **<Back**.



```
Open BSI AutoStart
        NDF: C:\ACCOL\IP1AndCW.NDF
        Auto Start Enabled
        Auto Stop Enabled
        60 second start-up retry period
        0 retries

Message Buffers
        Total pending requests: using default
        Total requests per device: 5

Device Health Checking
        60 seconds between device health checks
        Process all devices within each health check
        3 I/O failures for a device before going offline

Remote Alarm Support
        Remote Alarm Support Enabled
Acknowledge Return-to-Normal alarm reports
        Critical: No
        Non-Critical: No
        Operator Guide: No
        Event: No

RBE Support
```

Selection of the FINISH button will result in the update of any user changes to the database.

When you are sure you have selected all the right options, click **Finish** to apply this configuration to ObjectServer. and display the NW3000 Setup Tool dialog.

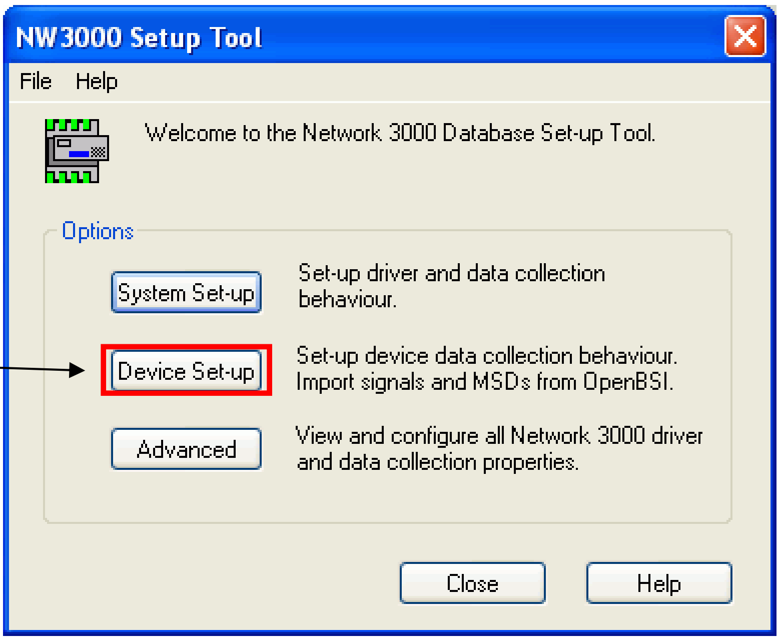
NW3000 Device Set-up Wizard

Click **Device Set-up** to start the device set-up wizard.

You can now go on to set up your individual devices. ObjectServer refers to your controllers as ”devices”.

Click **Device Set-up**.

Each page of the NW3000 Device Set-up wizard has a Help button which provides online help.



NW3000 Setup Tool

File Help

Welcome to the Network 3000 Database Set-up Tool.

Options

System Set-up — Set-up driver and data collection behaviour.

Device Set-up — Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

Advanced — View and configure all Network 3000 driver and data collection properties.

Close Help

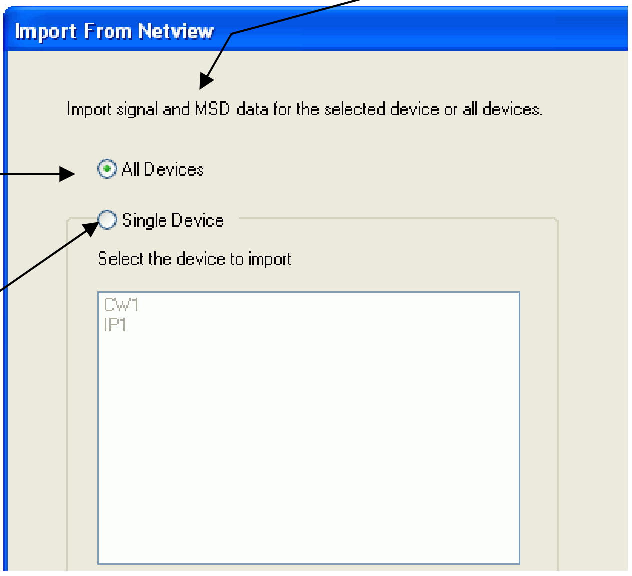
Import from Netview Page

Use this page to select the devices that import data into ObjectServer.

ObjectServer imports signal and MSD data for the selected devices.

By default, the wizard selects **All Devices**. ObjectServer assumes that you want to collect data from all of your controllers.

If you want to import data from a single device, select this option.



Import From Netview

Import signal and MSD data for the selected device or all devices.

[✓ All Devices]

Single Device

Select the device to import

CW1
IP1

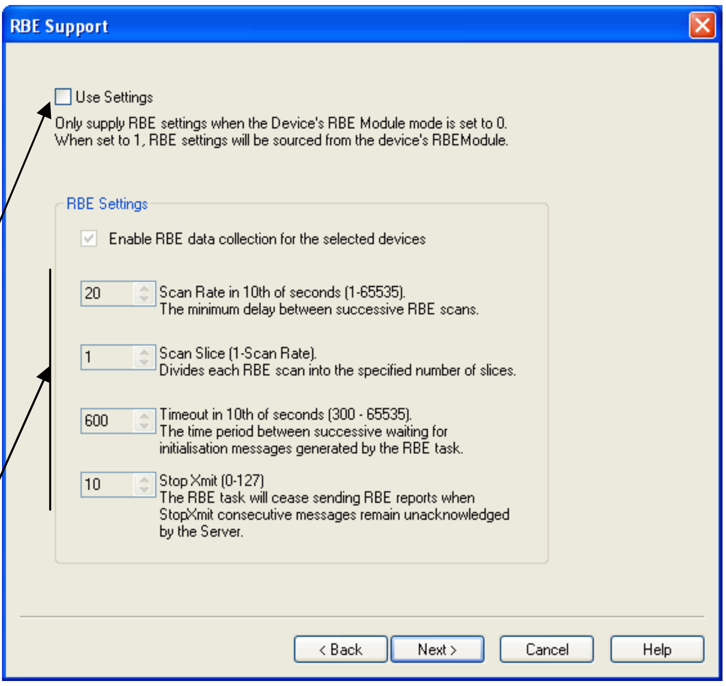
RBE Support Page

Use the RBE Support page to specify RBE settings. This can done **only** if you have set the MODE input within the controllers’ RBE module to zero (0).

Since this option is not checked by default, ObjectServer assumes that you want to apply the RBE settings defined in the control strategy file for your controller.

If you have set the RBE module in your controller(s) to zero (0), you can configure your RBE settings using this group of options.

Note: Although all of these settings work for NW3000 RTUs, **not all** settings work for ControlWave RTUs.


RBE Support

☐ Use Settings

Only supply RBE settings when the Device's RBE Module mode is set to 0.
When set to 1, RBE settings will be sourced from the device's RBEModule.

RBE Settings

✓ Enable RBE data collection for the selected devices

20 Scan Rate in 10th of seconds (1-65535).
The minimum delay between successive RBE scans.

1 Scan Slice (1-Scan Rate).
Divides each RBE scan into the specified number of slices.

600 Timeout in 10th of seconds (300 - 65535).
The time period between successive waiting for initialisation messages generated by the RBE task.

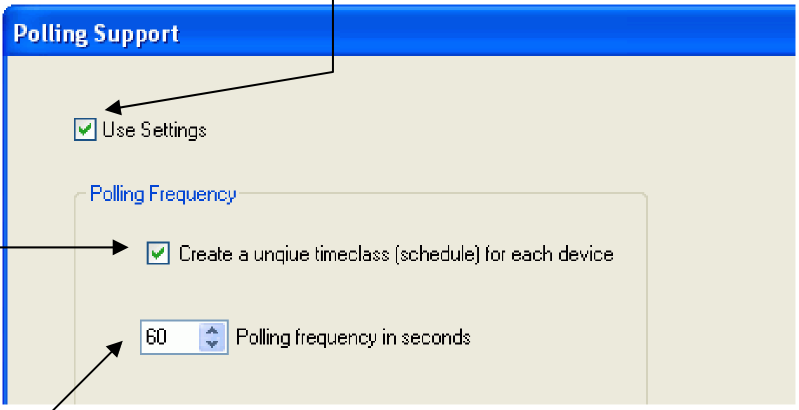
10 Stop Xmit (0-127)
The RBE task will cease sending RBE reports when StopXmit consecutive messages remain unacknowledged by the Server.

Polling Support Page

Use this page to specify a polling rate for the selected controller(s). If you have more than one device and have chosen to import data for all devices, you can specify a collection schedule for each device from here.

ObjectServer assumes that you want to set up a polling schedule for the selected device(s). This collects data for all global signals.

If you have selected more than one device, ObjectServer creates a separate polling schedule for each device when you select this option.


Polling Support

✓ Use Settings

Polling Frequency

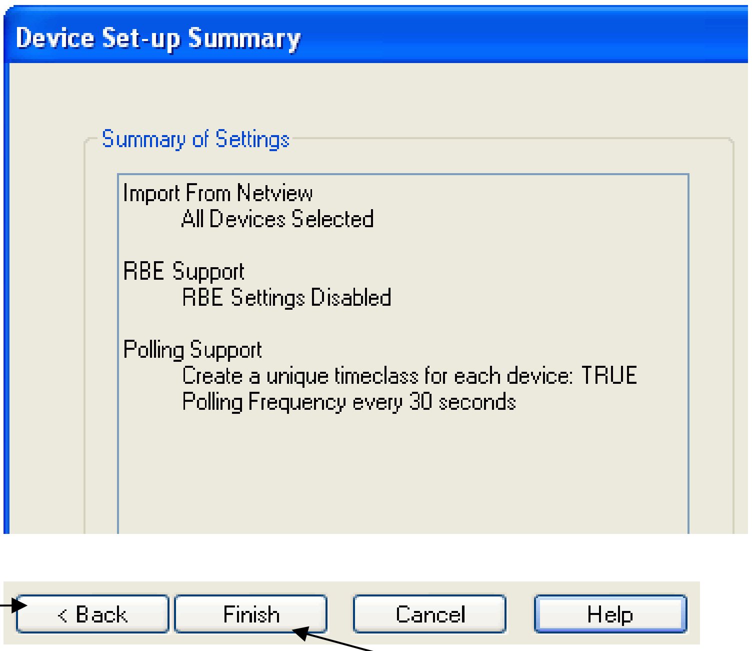
✓ Create a unqiue timeclass (schedule) for each device

60 Polling frequency in seconds

You define the frequency of polling for each device here, but you can change these polling frequencies later by running the NW3000 Set-up Tool and selecting the **Advanced** button.

Device Set-up Summary Page

Use this page to view a summary of the configuration prior to accepting it.



Device Set-up Summary

Summary of Settings

Import From Netview
    All Devices Selected

RBE Support
    RBE Settings Disabled

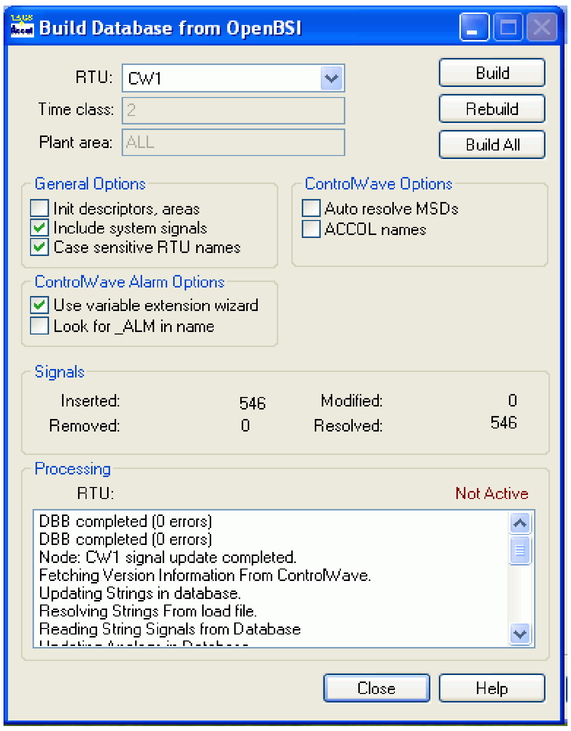
Polling Support
    Create a unique timeclass for each device: TRUE
    Polling Frequency every 30 seconds

If you want to change any of the settings listed here click **<Back** to go back to the relevant page and make the changes.

If you are happy with the configuration, click **Finish**. The wizard then applies these settings using the Database Builder and Template Builder.

Database Builder

The Database Builder runs automatically, importing the devices and signals you have defined during the set-up procedure.



Build Database from OpenBSI

RTU: CW1
Time class: 2
Plant area: ALL

General Options
- [ ] Init descriptors, areas
- [x] Include system signals
- [x] Case sensitive RTU names

ControlWave Options
- [ ] Auto resolve MSDs
- [ ] ACCOL names

ControlWave Alarm Options
- [x] Use variable extension wizard
- [ ] Look for _ALM in name

Signals

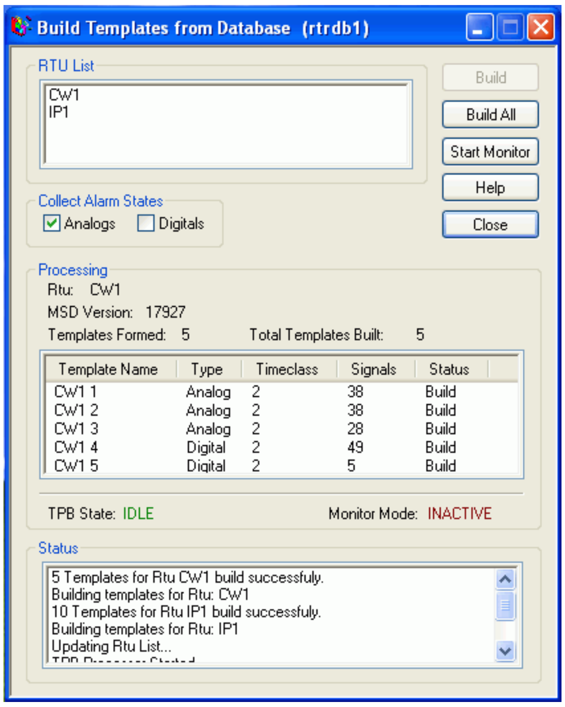
Inserted:	546	Modified:	0
Removed:	0	Resolved:	546

Processing
RTU: Not Active

```
DBB completed (0 errors)
DBB completed (0 errors)
Node: CW1 signal update completed.
Fetching Version Information From ControlWave.
Updating Strings in database.
Resolving Strings From load file.
Reading String Signals from Database
```

Template Builder

After the Database Builder finishes, it closes and the Template Builder opens. The Template Builder creates the necessary templates to ensure the data is collected efficiently. When finished, it closes.



RTU List: CW1, IP1

Collect Alarm States: [✓ Analogs] Digitals

Rtu: CW1
MSD Version: 17927
Templates Formed: 5   Total Templates Built: 5

Template Name	Type	Timeclass	Signals	Status
CW1 1	Analog	2	38	Build
CW1 2	Analog	2	38	Build
CW1 3	Analog	2	28	Build
CW1 4	Digital	2	49	Build
CW1 5	Digital	2	5	Build

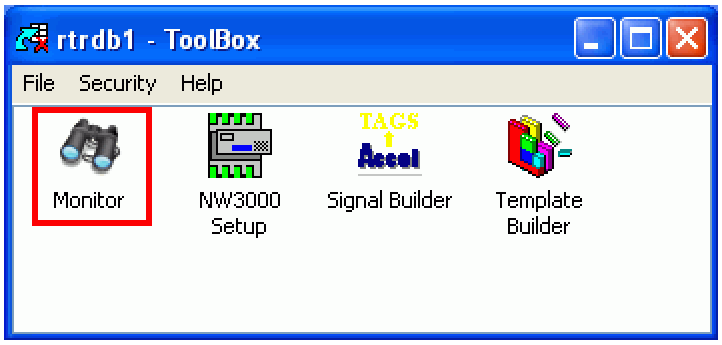
TPB State: IDLE   Monitor Mode: INACTIVE

```
5 Templates for Rtu CW1 build successfuly.
Building templates for Rtu: CW1
10 Templates for Rtu IP1 build successfuly.
Building templates for Rtu: IP1
Updating Rtu List...
```

Congratulations! You have completed the configuration of ObjectServer for data collection from your controllers. ObjectServer should begin collecting data immediately.

Confirming that Data is Being Collected

ObjectServer comes with a suite of tools that enable you to monitor and configure your system. These tools are available from the ObjectServer Toolbox. Open the Toolbox by selecting **Start → Programs → OpenBSI → ObjectServer →** Toolbox. The Monitor tool icon is in the Toolbox:



You can use the Monitor tool to confirm that data is being collected. We cover the Monitor tool in some detail in the next chapter (*5, Monitoring Controllers*).

In this chapter we have shown how you prepare ObjectServer for collecting data from your ControlWave and NW3000 controllers by using the NW3000Setup tool.

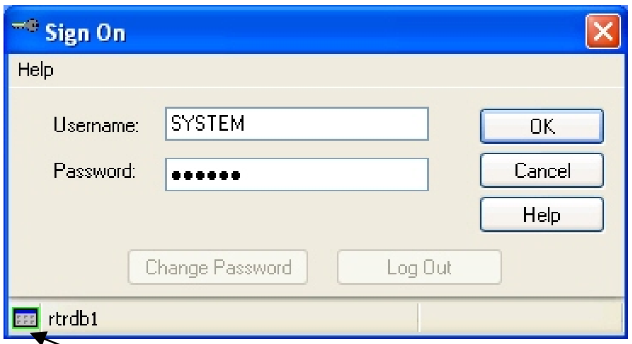
Chapter 5 – Monitoring Bristol Controllers

We need to be able to know the state of our Bristol controllers at all times. Are they connected to ObjectServer? Are they polling successfully? Are they sending RBE signal data? Are they sending alarm data properly? The Monitor tool answers these and other questions.

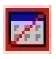
NOTE: This is **not** an exhaustive description of every feature of the Monitor tool. For that, consult the Monitor tool's online help. This is just a discussion of a few of the most important features related to monitoring your controllers.

Starting the Monitor

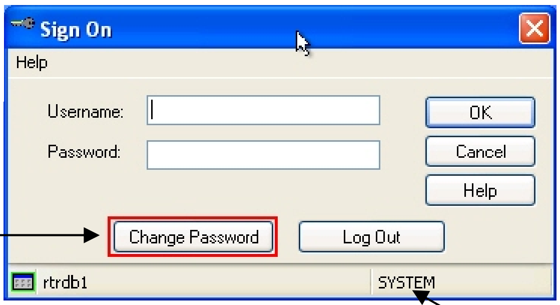
1. Click **Start → All Programs → OpenBSI Tools → ObjectServer → Toolbox**. The Sign On dialog appears.



This green symbol on the left labelled **rtrdb1** indicates that you are connected to the ObjectServer data service. If you were not connected, the symbol would be red.

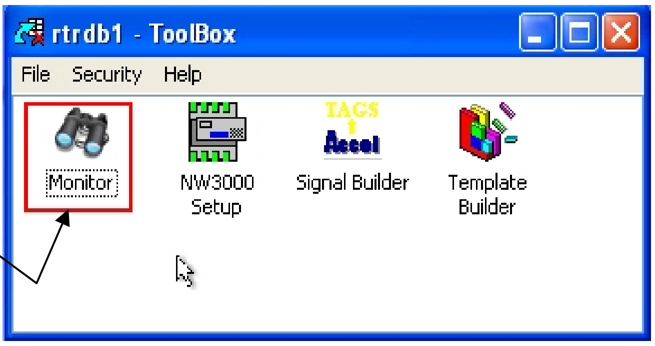


2. You must log in as the SYSTEM user. The password is SYSTEM in capitals, exactly the same as the name. It is advisable once you have logged in the first time to change the password, using the Change Password button.



Note that the SYSTEM user's name displays here once you are logged in.

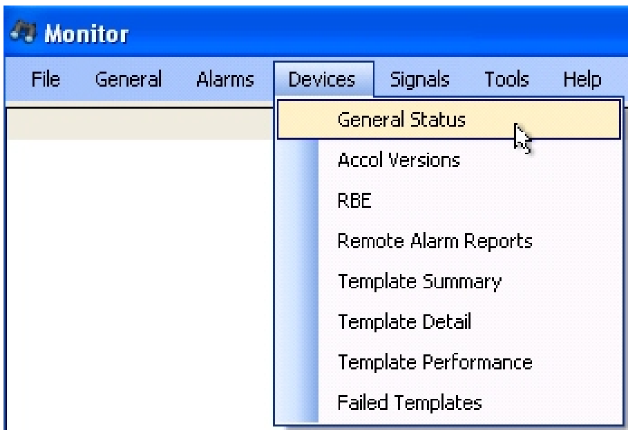
3. Once you are logged in, the tools in the toolbox display. Double-click the Monitor tool icon to start it.



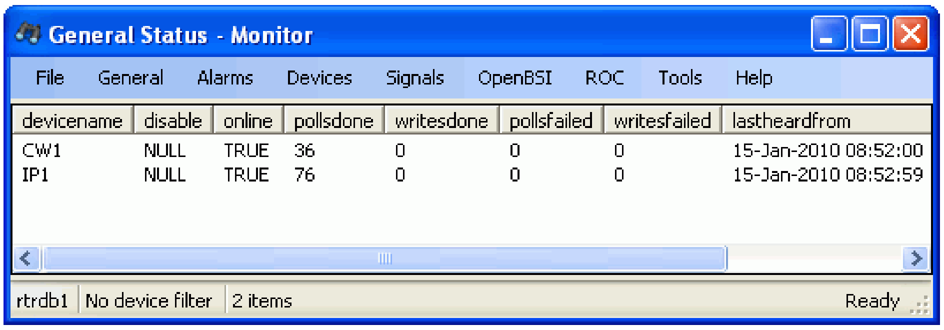
Viewing Device Status

This option allows you to view the general communication status of your devices. When the Monitor first opens, its window is blank until you make a selection from one of the menus.

1. To view the status of your NW3000 or ControlWave devices, select **Devices → General Status** from the menu bar.



2. The devices list with key properties displayed.



devicename	disable	online	pollsdone	writesdone	pollsfailed	writesfailed	lastheardfrom
CW1	NULL	TRUE	36	0	0	0	15-Jan-2010 08:52:00
IP1	NULL	TRUE	76	0	0	0	15-Jan-2010 08:52:59

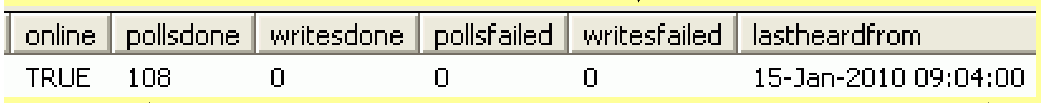
rtrdb1 | No device filter | 2 items | Ready

What do the General Status Values Mean?

Let's look at these properties to see what they mean, and the kinds of values that you should expect when your controllers are working correctly.

Online indicates the general communication status of your controller. TRUE means healthy communications.

Writesdone and *writesfailed* counts indicate how well ObjectServer is doing when writing to the controllers. If *writesfailed* is increasing, you need to



online	pollsdone	writesdone	pollsfailed	writesfailed	lastheardfrom
TRUE	108	0	0	0	15-Jan-2010 09:04:00

Pollsdone and *pollsfailed* indicate how successful template collection has been. If the *pollsfailed* count is increasing, you need to check that control strategy load version numbers match up, since this is the most likely explanation for this kind of behaviour.

The time of the last successful communication with the controller.

Viewing ACCOL Versions

The ACCOL Versions menu item shows discrepancies in the control strategy file version number between the database and the device. If you find that some of the polls are failing on one of your controllers, one of the first things you should check is that the control strategy file version numbers match up for the controller that is affected.

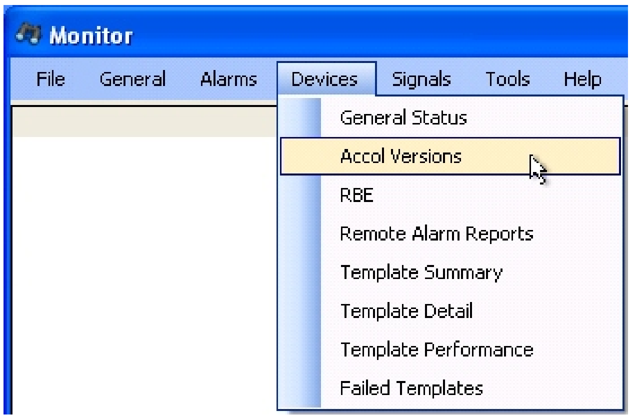
What are Control Strategy (ACCOL or 1131) File Version Numbers?

Whenever you download a modified control strategy file, it is given a new version number in the controller.

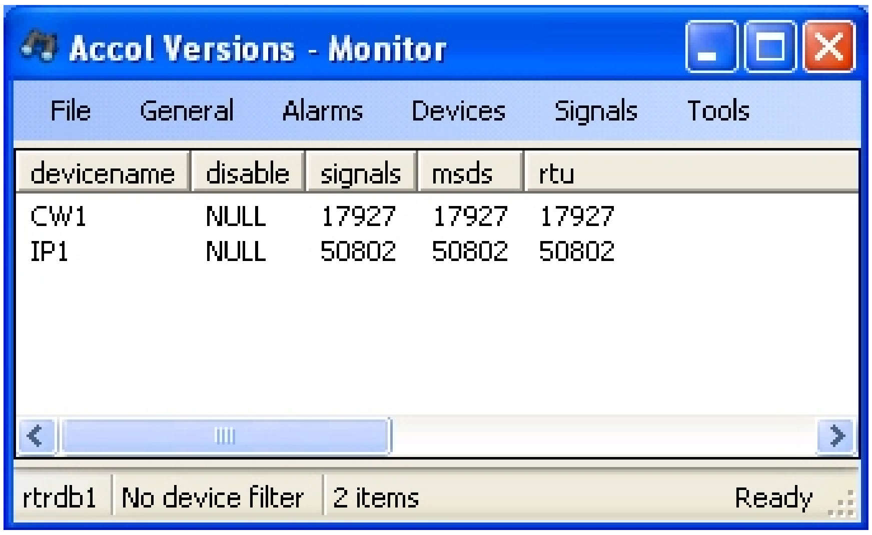
What can happen when a new version is downloaded to a controller?

When you download a new version of the control strategy file to a controller, if you do not update the ObjectServer database the version number in the database conflicts with the version number in the RTU. This could cause polling to fail.

1. To view control strategy file version numbers, select **Devices → Accol Versions** from the Monitor menu bar.



2. The control strategy file version numbers for each device display.



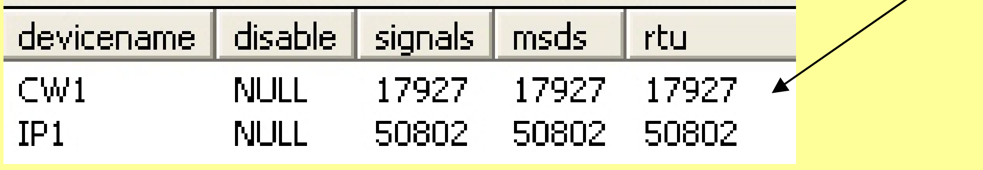
devicename	disable	signals	msds	rtu
CW1	NULL	17927	17927	17927
IP1	NULL	50802	50802	50802

rtrdb1 | No device filter | 2 items | Ready

More about Version Numbers

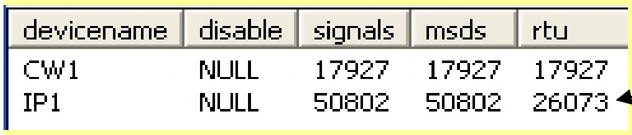
As we said previously, when you download a modified strategy file, it is given a new version number within the controller.

All version numbers (signals, msd, and rtu) should agree for each device. This is correct.



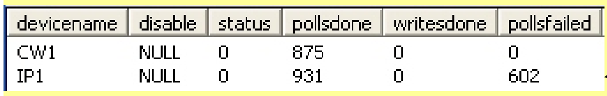
devicename	disable	signals	msds	rtu
CW1	NULL	17927	17927	17927
IP1	NULL	50802	50802	50802

Here, a new version has been downloaded to the IP1 device. The rtu version number is now differs from the signal and msd version numbers for that device.



devicename	disable	signals	msds	rtu
CW1	NULL	17927	17927	17927
IP1	NULL	50802	50802	26073

This increases the *pollsfailed* count for that device. The only way we can stop this is to update the database. Instructions are given on how to do this in the chapter on troubleshooting.

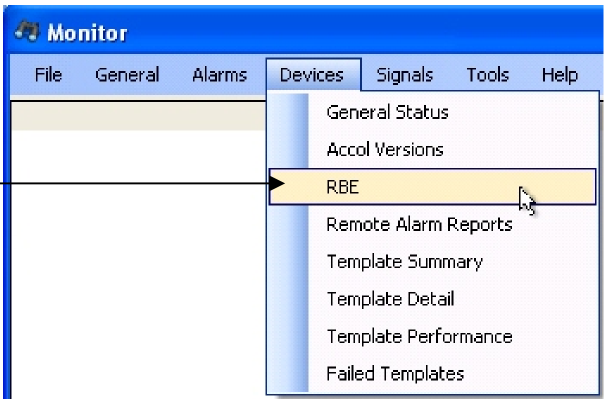


devicename	disable	status	pollsdone	writesdone	pollsfailed
CW1	NULL	0	875	0	0
IP1	NULL	0	931	0	602

Viewing RBE Stats

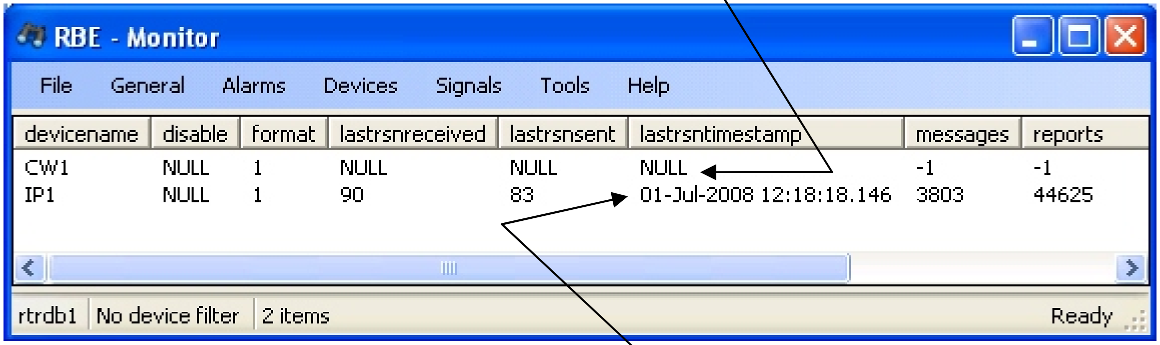
RBE (Report By Exception) statistics confirm that RBE is working correctly for a device.

1. Select **Devices → RBE**.



2. RBE statistics display for each device.

This device has no RBE signals, so no reports are being received.



devicename	disable	format	lastrsnreceived	lastrsnsent	lastrsntimestamp	messages	reports
CW1	NULL	1	NULL	NULL	NULL	-1	-1
IP1	NULL	1	90	83	01-Jul-2008 12:18:18.146	3803	44625

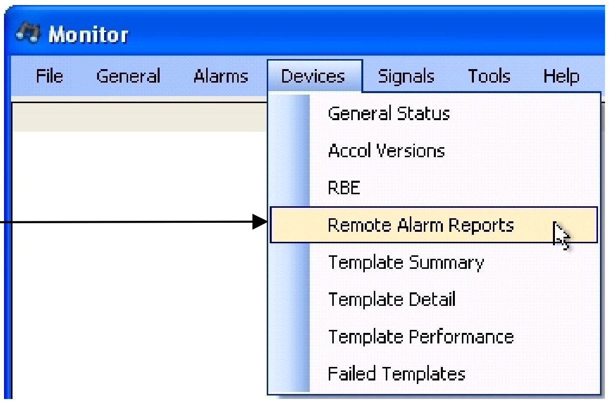
rtrdb1 | No device filter | 2 items | Ready

This device has RBE signals and reports **are** being received.

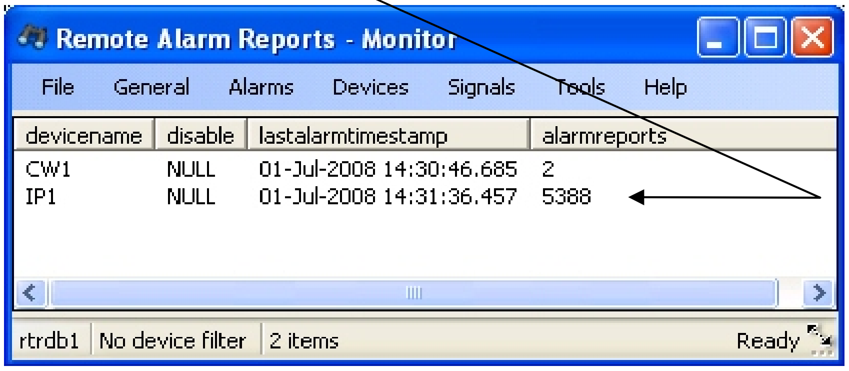
Viewing Alarm Stats

You can also check that alarm signal data is being collected.

1. Select **Devices → Remote Alarm Reports** from the menu bar.



2. Remote alarm report statistics display.



```
devicename  disable  lastalarmtimestamp        alarmreports
CW1         NULL     01-Jul-2008 14:30:46.685  2
IP1         NULL     01-Jul-2008 14:31:36.457  5388
```

rtrdb1 | No device filter | 2 items | Ready

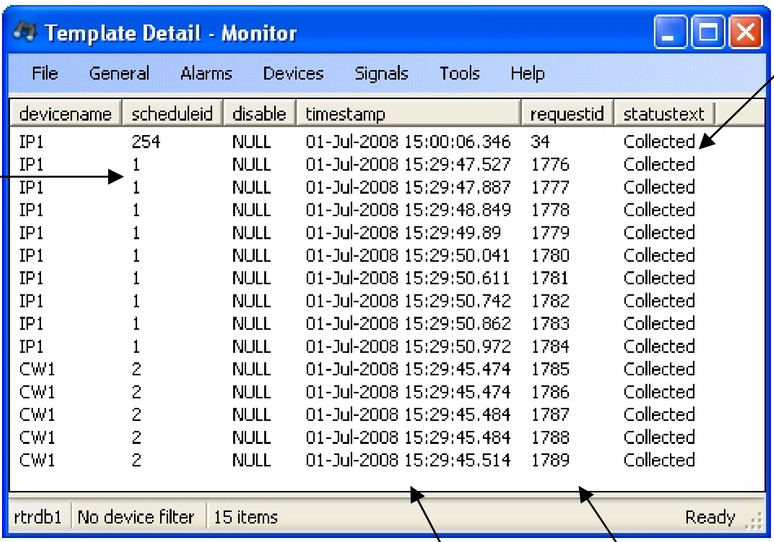
Viewing Template Stats

The final few options on the **Device** menu enable you to review the statistics for template collection from a number of angles.

We have already seen the *pollsdone* and *pollsfailed* statistics displayed in the General Status page, so we can look at the Template Detail view. Select **Devices → Template Detail** from the Monitor menu bar.

The *scheduleid* for each template displays. Each row signifies a single template.

statustext for each template should be “Collected”.


Template Detail - Monitor
File General Alarms Devices Signals Tools Help

devicename	scheduleid	disable	timestamp	requestid	statustext
IP1	254	NULL	01-Jul-2008 15:00:06.346	34	Collected
IP1	1	NULL	01-Jul-2008 15:29:47.527	1776	Collected
IP1	1	NULL	01-Jul-2008 15:29:47.887	1777	Collected
IP1	1	NULL	01-Jul-2008 15:29:48.849	1778	Collected
IP1	1	NULL	01-Jul-2008 15:29:49.89	1779	Collected
IP1	1	NULL	01-Jul-2008 15:29:50.041	1780	Collected
IP1	1	NULL	01-Jul-2008 15:29:50.611	1781	Collected
IP1	1	NULL	01-Jul-2008 15:29:50.742	1782	Collected
IP1	1	NULL	01-Jul-2008 15:29:50.862	1783	Collected
IP1	1	NULL	01-Jul-2008 15:29:50.972	1784	Collected
CW1	2	NULL	01-Jul-2008 15:29:45.474	1785	Collected
CW1	2	NULL	01-Jul-2008 15:29:45.474	1786	Collected
CW1	2	NULL	01-Jul-2008 15:29:45.484	1787	Collected
CW1	2	NULL	01-Jul-2008 15:29:45.484	1788	Collected
CW1	2	NULL	01-Jul-2008 15:29:45.514	1789	Collected

rtrdb1 | No device filter | 15 items | Ready

The timestamps for templates collected from a device should be close together.

Request id numbers should be contiguous.

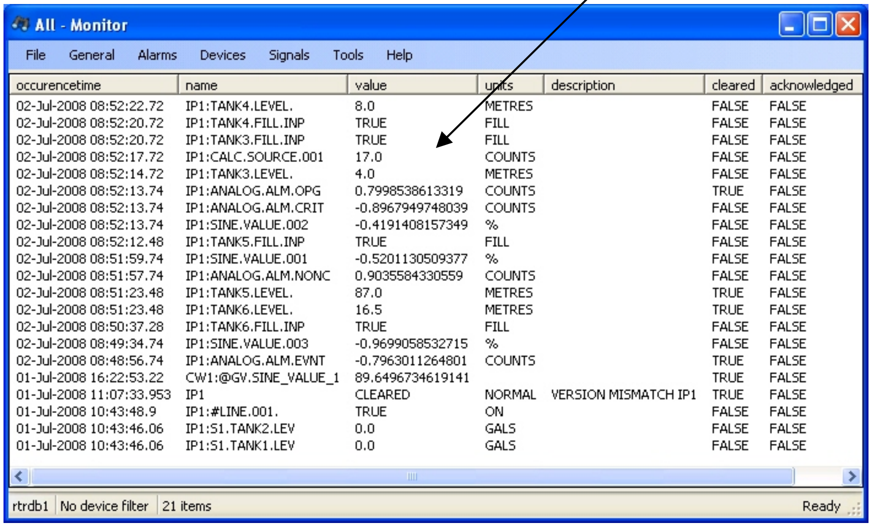
Viewing Alarms

Not only can you view alarm statistics on a per device basis with the Monitor tool, you can also view alarms.

Viewing All Alarms

Select **Alarms → All** from the menu bar.

All current alarms display



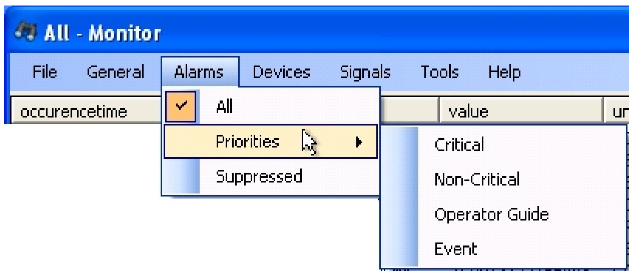
occurencetime	name	value	units	description	cleared	acknowledged
02-Jul-2008 08:52:22.72	IP1:TANK4.LEVEL.	8.0	METRES		FALSE	FALSE
02-Jul-2008 08:52:20.72	IP1:TANK4.FILL.INP	TRUE	FILL		FALSE	FALSE
02-Jul-2008 08:52:20.72	IP1:TANK3.FILL.INP	TRUE	FILL		FALSE	FALSE
02-Jul-2008 08:52:17.72	IP1:CALC.SOURCE.001	17.0	COUNTS		FALSE	FALSE
02-Jul-2008 08:52:14.72	IP1:TANK3.LEVEL.	4.0	METRES		FALSE	FALSE
02-Jul-2008 08:52:13.74	IP1:ANALOG.ALM.OPG	0.7998538613319	COUNTS		TRUE	FALSE
02-Jul-2008 08:52:13.74	IP1:ANALOG.ALM.CRIT	-0.8967949748039	COUNTS		FALSE	FALSE
02-Jul-2008 08:52:13.74	IP1:SINE.VALUE.002	-0.4191408157349	%		FALSE	FALSE
02-Jul-2008 08:52:12.48	IP1:TANK5.FILL.INP	TRUE	FILL		FALSE	FALSE
02-Jul-2008 08:51:59.74	IP1:SINE.VALUE.001	-0.5201130509377	%		FALSE	FALSE
02-Jul-2008 08:51:57.74	IP1:ANALOG.ALM.NONC	0.9035584330559	COUNTS		FALSE	FALSE
02-Jul-2008 08:51:23.48	IP1:TANK5.LEVEL.	87.0	METRES		TRUE	FALSE
02-Jul-2008 08:51:23.48	IP1:TANK6.LEVEL.	16.5	METRES		TRUE	FALSE
02-Jul-2008 08:50:37.28	IP1:TANK6.FILL.INP	TRUE	FILL		FALSE	FALSE
02-Jul-2008 08:49:34.74	IP1:SINE.VALUE.003	-0.9699058532715	%		FALSE	FALSE
02-Jul-2008 08:48:56.74	IP1:ANALOG.ALM.EVNT	-0.7963011264801	COUNTS		TRUE	FALSE
01-Jul-2008 16:22:53.22	CW1:@GV.SINE_VALUE_1	89.6496734619141			TRUE	FALSE
01-Jul-2008 11:07:33.953	IP1	CLEARED	NORMAL	VERSION MISMATCH IP1	TRUE	FALSE
01-Jul-2008 10:43:48.9	IP1:#LINE.001.	TRUE	ON		FALSE	FALSE
01-Jul-2008 10:43:46.06	IP1:S1.TANK2.LEV	0.0	GALS		FALSE	FALSE
01-Jul-2008 10:43:46.06	IP1:S1.TANK1.LEV	0.0	GALS		FALSE	FALSE

rtrdb1 | No device filter | 21 items | Ready

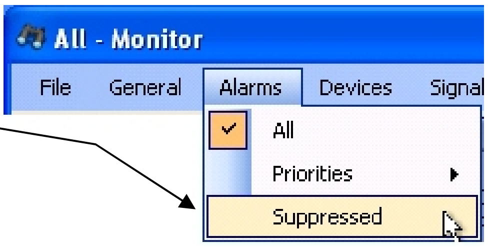
Viewing Filtered Alarms

You can also filter the alarms using options on the **Alarms** menu.

1. You can filter the alarms according to priority by using the appropriate option in the **Priorities** group from the **Alarms** menu.



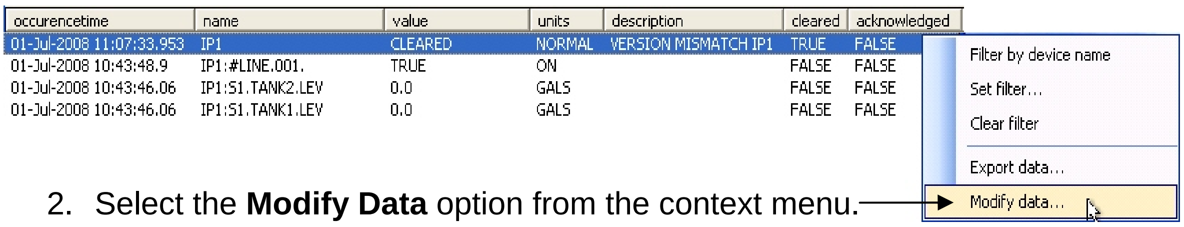
2. Or you can view suppressed alarms by selecting the "Suppressed" option. Suppressed alarms are not signals which have their Alarm Inhibit flag set. They are actually alarms, but the Suppressed property of the alarm is set



Acknowledging Alarms

You can also acknowledge alarms using the Monitor tool.

1. Right-click on an alarm's acknowledged attribute:

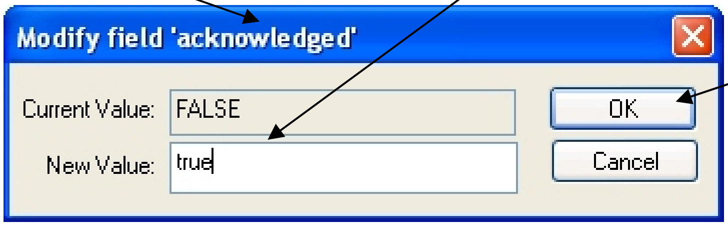


occurencetime	name	value	units	description	cleared	acknowledged
01-Jul-2008 11:07:33.953	IP1	CLEARED	NORMAL	VERSION MISMATCH IP1	TRUE	FALSE
01-Jul-2008 10:43:48.9	IP1:#LINE.001.	TRUE	ON		FALSE	FALSE
01-Jul-2008 10:43:46.06	IP1:S1.TANK2.LEV	0.0	GALS		FALSE	FALSE
01-Jul-2008 10:43:46.06	IP1:S1.TANK1.LEV	0.0	GALS		FALSE	FALSE

2. Select the **Modify Data** option from the context menu.

3. Type **true** into the New Value field.

Check that the field named here is the one you want to modify.



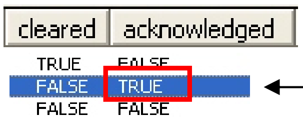
Modify field 'acknowledged'

Current Value: FALSE

New Value: true

Then click OK.

4. The alarm is acknowledged. If was already cleared (as shown in the example in step 1 above) it would disappear from the alarm list. If it was not cleared, the acknowledged property is set to true.



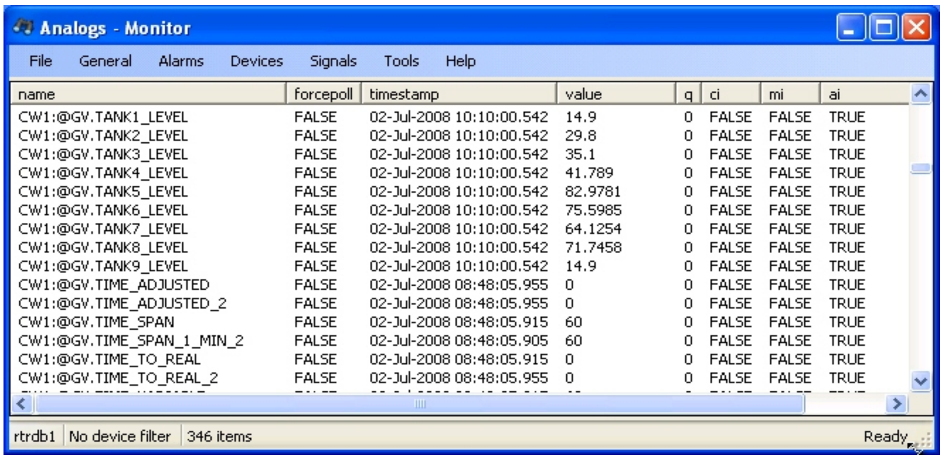
cleared	acknowledged
TRUE	FALSE
FALSE	TRUE
FALSE	FALSE

You can change any value of an alarm except for the *occurencetime*.

Viewing Signals

It is possible to view analog, digital, and string signals with the Monitor tool. Select **Signals → Analogs** from the Monitor menu bar.

The Monitor window displays a list of all analog signals.



Analogs - Monitor

File General Alarms Devices Signals Tools Help

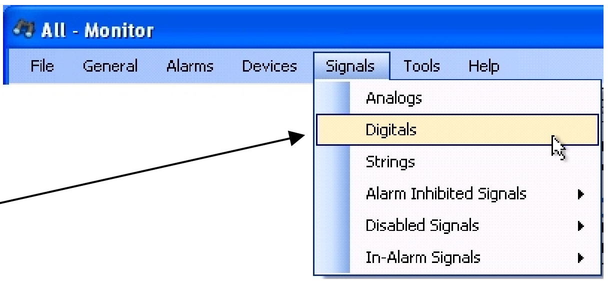
name	forcepoll	timestamp	value	q	ci	mi	ai
CW1:@GV.TANK1_LEVEL	FALSE	02-Jul-2008 10:10:00.542	14.9	0	FALSE	FALSE	TRUE
CW1:@GV.TANK2_LEVEL	FALSE	02-Jul-2008 10:10:00.542	29.8	0	FALSE	FALSE	TRUE
CW1:@GV.TANK3_LEVEL	FALSE	02-Jul-2008 10:10:00.542	35.1	0	FALSE	FALSE	TRUE
CW1:@GV.TANK4_LEVEL	FALSE	02-Jul-2008 10:10:00.542	41.789	0	FALSE	FALSE	TRUE
CW1:@GV.TANK5_LEVEL	FALSE	02-Jul-2008 10:10:00.542	82.9781	0	FALSE	FALSE	TRUE
CW1:@GV.TANK6_LEVEL	FALSE	02-Jul-2008 10:10:00.542	75.5985	0	FALSE	FALSE	TRUE
CW1:@GV.TANK7_LEVEL	FALSE	02-Jul-2008 10:10:00.542	64.1254	0	FALSE	FALSE	TRUE
CW1:@GV.TANK8_LEVEL	FALSE	02-Jul-2008 10:10:00.542	71.7458	0	FALSE	FALSE	TRUE
CW1:@GV.TANK9_LEVEL	FALSE	02-Jul-2008 10:10:00.542	14.9	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_ADJUSTED	FALSE	02-Jul-2008 08:48:05.955	0	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_ADJUSTED_2	FALSE	02-Jul-2008 08:48:05.955	0	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_SPAN	FALSE	02-Jul-2008 08:48:05.915	60	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_SPAN_1_MIN_2	FALSE	02-Jul-2008 08:48:05.905	60	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_TO_REAL	FALSE	02-Jul-2008 08:48:05.915	0	0	FALSE	FALSE	TRUE
CW1:@GV.TIME_TO_REAL_2	FALSE	02-Jul-2008 08:48:05.955	0	0	FALSE	FALSE	TRUE

rtrdb1 | No device filter | 346 items | Ready

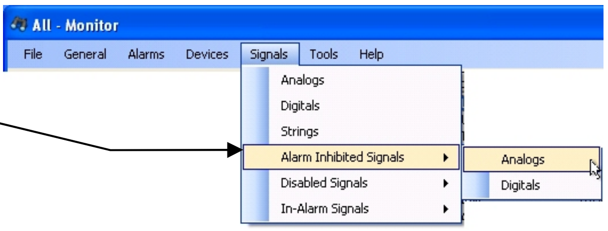
Signal Filters

These are all of the filtering options on the **Signals** menu.

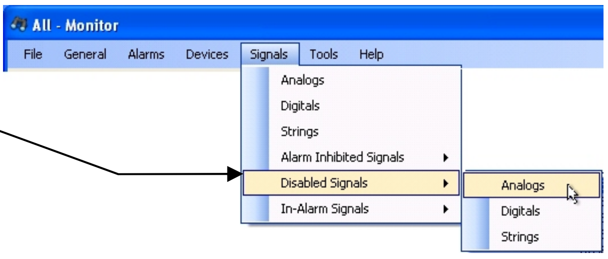
1. You can filter by signal type.We have already mentioned the analogs filter, but you can also view digital or string-only signals by selecting the appropriate option.



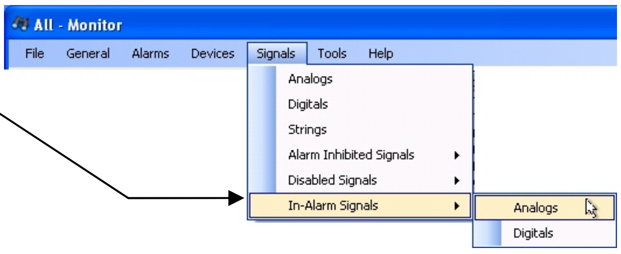
2. You can view only signals which are alarm-inhibited.



3. Disabled signals…



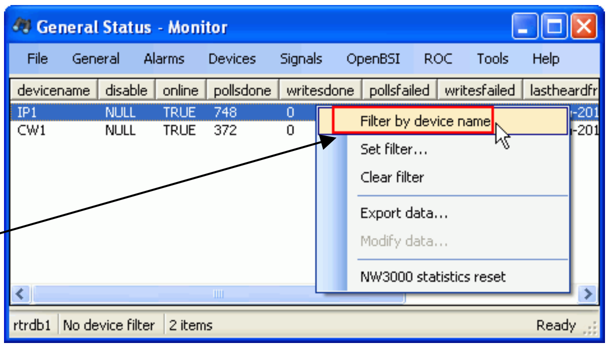
4. Or signals that are in alarm.



Using a Device Filter

You can filter any of the views available from the Monitor tool according to the device name.

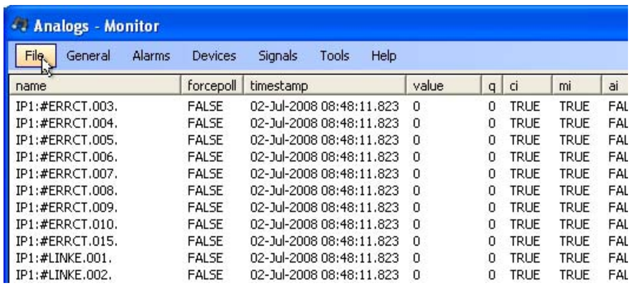
1. Right click on the view and select the **Filter by device name** option from the context menu.



devicename	disable	online	pollsdone	writesdone	pollsfailed	writesfailed	lastheardfr
IP1	NULL	TRUE	748	0			-201
CW1	NULL	TRUE	372	0			-201

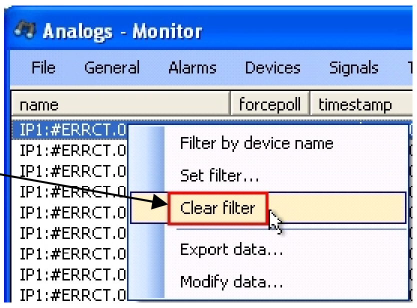
rtdb1 | No device filter | 2 items | Ready

2. If we now select the Signals → Analogs menu, the signal list content filters according to the device filter that we supplied. Here, only objects from the IP1 device display.



name	forcepoll	timestamp	value	q	ci	mi	ai
IP1:#ERRCT.003.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.004.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.005.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.006.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.007.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.008.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.009.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.010.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#ERRCT.015.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#LINKE.001.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL
IP1:#LINKE.002.	FALSE	02-Jul-2008 08:48:11.823	0	0	TRUE	TRUE	FAL

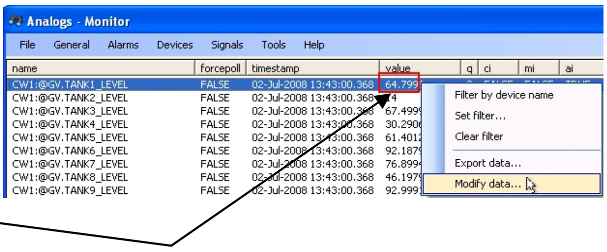
3. We can clear the device filter by right-clicking on any object and selecting the **Clear filter** option from the context menu.



Modifying Signal Properties

The Monitor tool allows you to monitor all signal properties except for the *timestamp*.

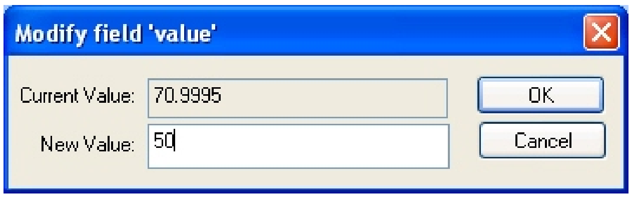
1. Right-click on the field of a signal that you want to modify, and then select the **Modify data** option. Here we have selected the *value* field.



Analogs - Monitor

name	forcepoll	timestamp	value	q	ci	mi	ai
CW1:@GV.TANK1_LEVEL	FALSE	02-Jul-2008 13:43:00.368	64.799				
CW1:@GV.TANK2_LEVEL	FALSE	02-Jul-2008 13:43:00.368	14				
CW1:@GV.TANK3_LEVEL	FALSE	02-Jul-2008 13:43:00.368	67.499				
CW1:@GV.TANK4_LEVEL	FALSE	02-Jul-2008 13:43:00.368	30.290				
CW1:@GV.TANK5_LEVEL	FALSE	02-Jul-2008 13:43:00.368	61.401				
CW1:@GV.TANK6_LEVEL	FALSE	02-Jul-2008 13:43:00.368	92.187				
CW1:@GV.TANK7_LEVEL	FALSE	02-Jul-2008 13:43:00.368	76.899				
CW1:@GV.TANK8_LEVEL	FALSE	02-Jul-2008 13:43:00.368	46.197				
CW1:@GV.TANK9_LEVEL	FALSE	02-Jul-2008 13:43:00.368	92.999				

2. The title bar on the Modify field dialog indicates the specific signal property you are modifying. Type a new value into the New Value field and click **OK**.

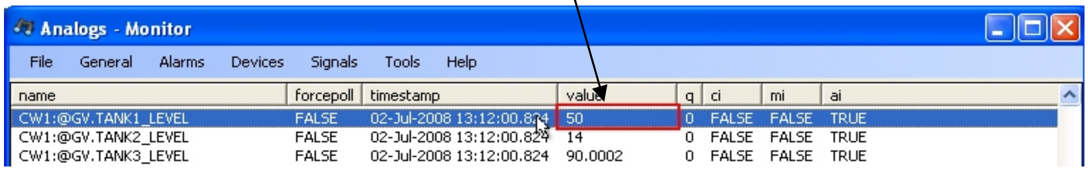


Modify field 'value'

Current Value: 70.9995

New Value: 50

The new value is applied.



Analogs - Monitor

name	forcepoll	timestamp	value	q	ci	mi	ai
CW1:@GV.TANK1_LEVEL	FALSE	02-Jul-2008 13:12:00.824	50	0	FALSE	FALSE	TRUE
CW1:@GV.TANK2_LEVEL	FALSE	02-Jul-2008 13:12:00.824	14	0	FALSE	FALSE	TRUE
CW1:@GV.TANK3_LEVEL	FALSE	02-Jul-2008 13:12:00.824	90.0002	0	FALSE	FALSE	TRUE

That completes this chapter. We have demonstrated how you can use ObjectServer's Monitor tool to:

1. View the status of any controller
2. View and acknowledge alarms
3. View and change signal values

Note: If you encounter problems viewing your controllers or signals in the Monitor tool, please refer to Chapter 8 – Troubleshooting.

[This page is left intentionally blank]

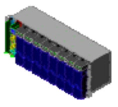
Chapter 6 – Basic Configuration Changes

From time to time, you need to modify most process control/SCADA systems. You may need to add or remove signals or even controllers. This chapter deals with this process of changing things in your system so that your controllers and ObjectServer match up every time you change something.

We see how to:

- Add controllers
- Remove controllers
- Add signals/variables
- Remove signals/variables
- Add signal/variable descriptions
- Change data collection rates
- Change RBE data collection

Remember, for our purposes the terms “controller”, “RTU”, and “device” are interchangeable and all mean the same thing.



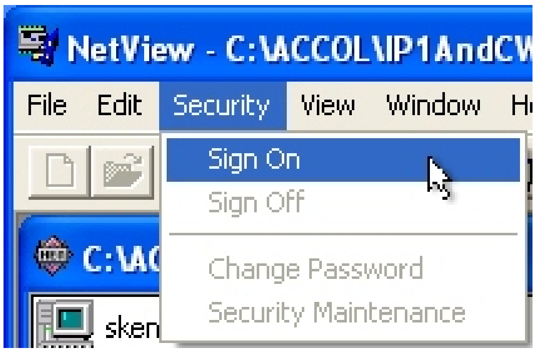
= Controller = Device = RTU

Adding a Controller

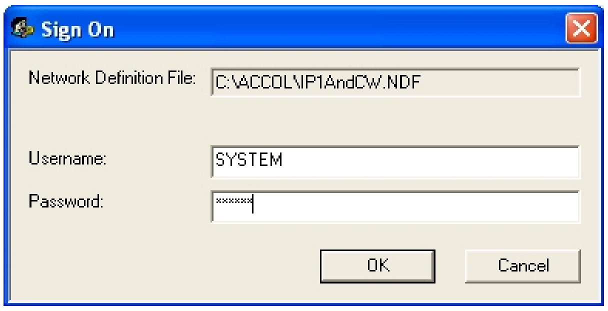
Imagine we have created our control strategy program, downloaded it locally to the new controller, and tested that the program works. We can now add the new controller to our OpenBSI network. Let’s walk through the first few steps.

Adding the New Controller to NetView

1. Log onto NetView as the SYSTEM user. Select **Security → Sign On** from the NetView menu bar.



2. Sign on as the SYSTEM user.


Sign On

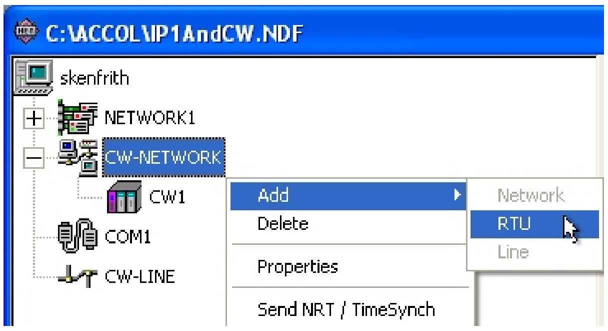
Network Definition File: C:\ACCOL\IP1AndCW.NDF

Username: SYSTEM

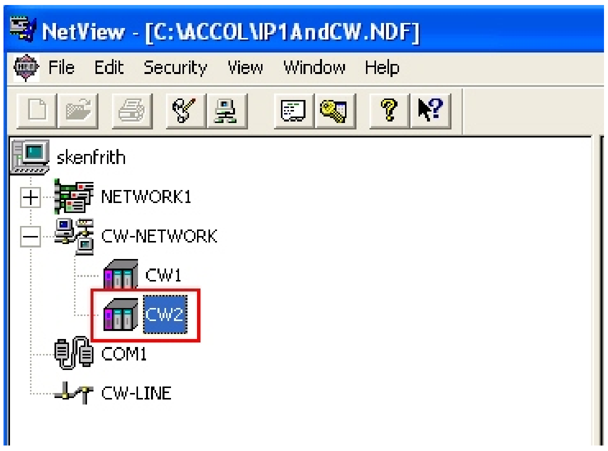
Password: ******

OK Cancel

3. Select the network for the new RTU/controller and select **Add → RTU** as shown here. Configure the RTU as you would normally. (For full instructions on how to do this, see the *Open BSI Utilities Manual,* document# D5081).



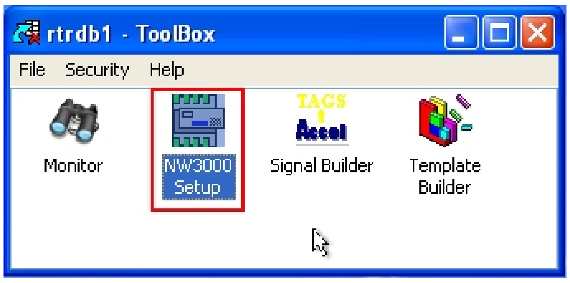
4. Once you've added the new controller to the network and the downloaded the tested control strategy program, it's time to update ObjectServer.



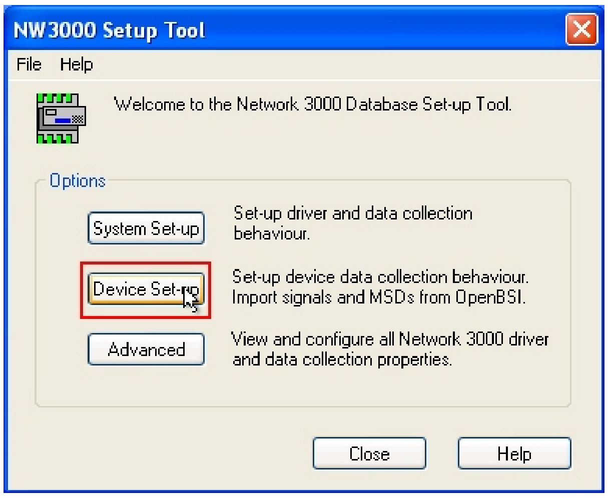
Updating ObjectServer

You now need to update the ObjectServer database with the new RTU and its signals. To do this, use the NW3000 Setup tool. This process also uses the Database Builder and Poll List Builder tools.

1. Open up the Toolbox and log in as the SYSTEM user. (Instructions for doing this are in *Step3: Import Device and Signal Information from OpenBSI* section of *Chapter 4 – ObjectServer Preparation*.) Double-click on the NW3000 Setup icon, highlighted in the image on the right.



2. Click **Device Set-up**.



Welcome to the Network 3000 Database Set-up Tool.

Set-up driver and data collection behaviour.

Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

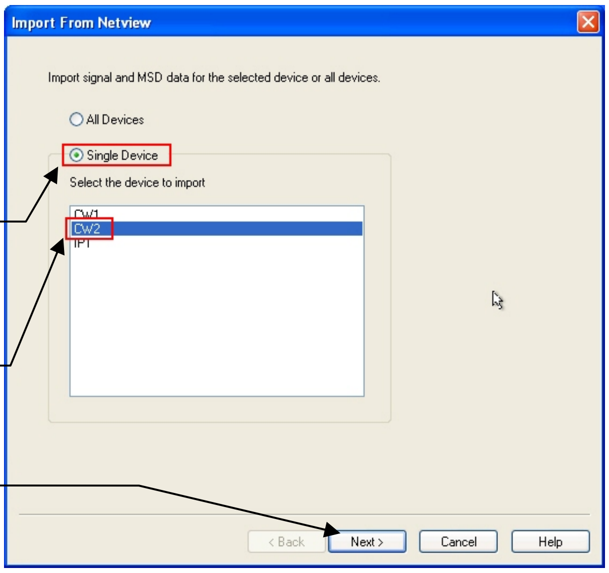
View and configure all Network 3000 driver and data collection properties.

3. When the Import from NetView dialog opens the All Devices option is selected by default.

   Select the **Single Device** option.

   Then select the new controller that you have just added to NetView, as shown.

   Click **Next>**.



Import From Netview

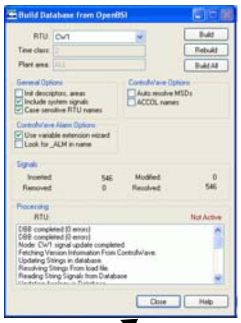
Import signal and MSD data for the selected device or all devices.

All Devices [✓ Single Device]

Select the device to import

CW1
CW2
IP1

4. Follow the instructions in the *NW3000 Device Set-up Wizard* section in *Chapter 4* to complete the import of the new device and its signals into ObjectServer.



Build Database from OpenBSI

RTU: CW1

Time class: 2

Plant area: ALL

General Options: Init descriptors, areas [✓ Include system signals] [✓ Case sensitive RTU names]

ControlWave Options: Auto resolve MSDs; ACCOL names

ControlWave Alarm Options: [✓ Use variable extension wizard] Look for _ALM in name

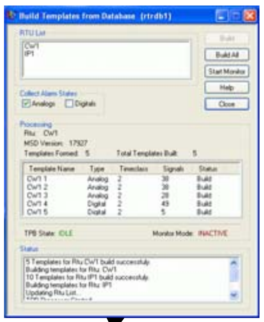
Signals

Inserted	546	Modified	0
Removed	0	Resolved	546

Processing

RTU: Not Active

```
DBB completed (0 errors)
DBB completed (0 errors)
Node: CW1 signal update completed
Fetching Version Information From ControlWave
Updating Strings in database
Resolving Strings From load file
Reading String Signals from Database
```



Build Templates from Database (rtrdb1)

RTU List

CW1
IP1

Collect Alarm States: [✓ Analogs] Digitals

Processing

Rtu: CW1

MSD Version: 17927

Templates Formed: 5 Total Templates Built: 5

Template Name	Type	Timeclass	Signals	Status
CW1 1	Analog	2	30	Build
CW1 2	Analog	2	30	Build
CW1 3	Analog	2	28	Build
CW1 4	Digital	2	49	Build
CW1 5	Digital	2	5	Build

TPB State: IDLE Monitor Mode: INACTIVE

Status

```
5 Templates for Rtu CW1 build successfully
Building templates for Rtu: CW1
10 Templates for Rtu IP1 build successfully
Building templates for Rtu: IP1
Updating Rtu List...
```

When the wizard completes it opens the Database Builder and Poll List Builder tools and imports the new device and its signals into the ObjectServer database.

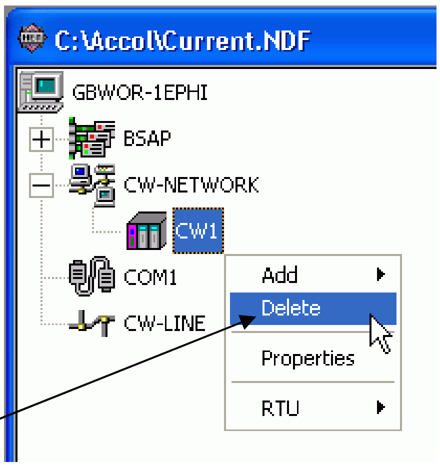
The new controller, its signals, and its polling templates have now been added to the ObjectServer database. ObjectServer can begin collecting data from the new device.

Removing a Controller

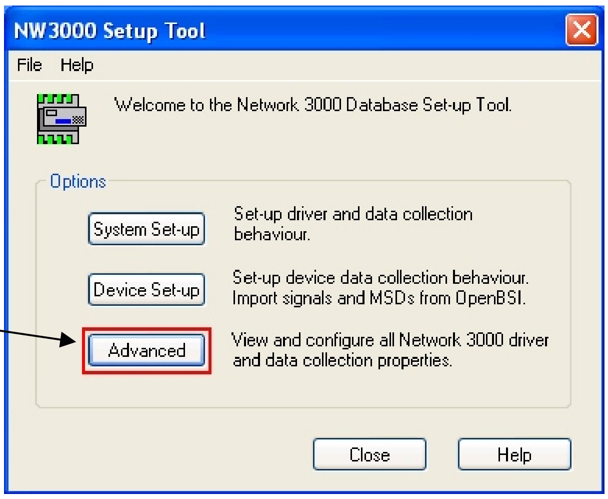
Removing a controller is very similar to adding one and requires the same two activities:

- Removing the controller from NetView.
- Removing the device from ObjectServer.

1. Open NetView and delete the device that is no longer required.



2. Open the NW3000 Set-up tool from the Toolbox and click **Advanced**.



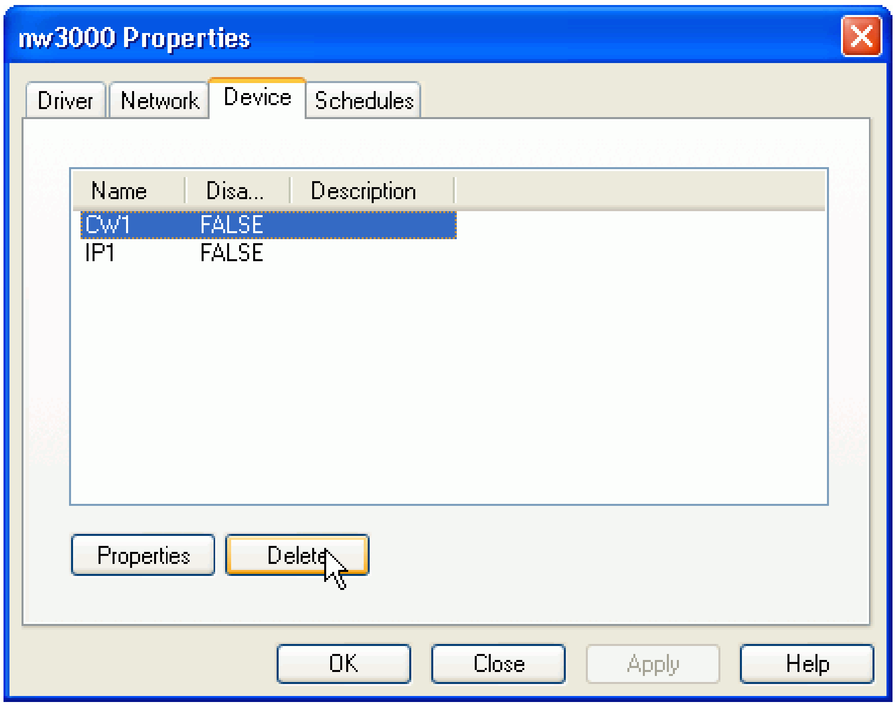
Welcome to the Network 3000 Database Set-up Tool.

Set-up driver and data collection behaviour.

Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

View and configure all Network 3000 driver and data collection properties.

3. When the “nw3000 Properties” dialog displays select the “Device” tab and then select the device that has been deleted from NetView. Then click the [Delete] button, followed by the [OK] button.



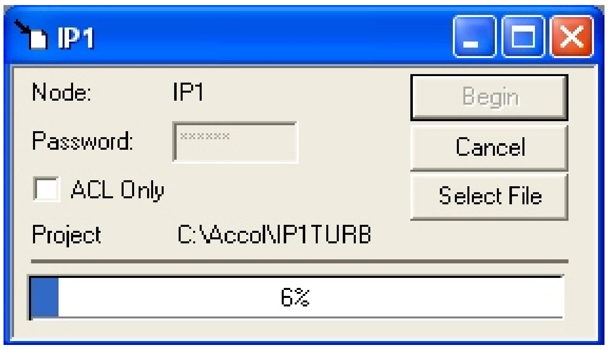
Name	Disa...	Description
CW1	FALSE	
IP1	FALSE	

Adding Signals/Variables

Occasionally you may need to add or remove a complete device from the controller network. More often than not, though, you just need to add or remove signals from existing controllers.

Note: “Signals” and “variables” are the same thing. They represent our real world process control objects. However, this guide uses ”signal” in reference with NW3000 controllers and “variable” in reference to ControlWave devices.

1. Create a signal using either Workbench or ControlWave Designer, depending on the device. Bear in mind the rules outlined in the *Identifying and Preparing Signals for Collection by ObjectServer* (from *Chapter 3 – Controller Preparation*), and download the updated strategy file to the controller.


IP1

Node: IP1

Password: ******

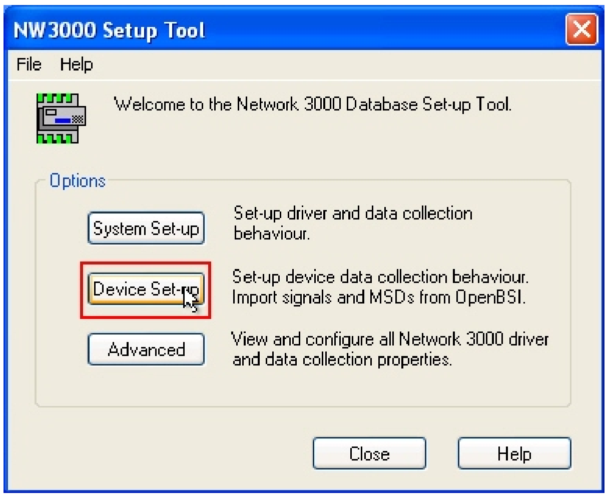
☐ ACL Only

Project C:\Accol\IP1TURB

6%

Note: For details of how to download a new control strategy file to a NW3000 or ControlWave device see the OpenBSI Utilities Manual (Document #D5081). You can find this on the OpenBSI installation disk.

2. Open the NW3000 Set-up tool from the Toolbox and click **Device Set-up**.


NW3000 Setup Tool

Welcome to the Network 3000 Database Set-up Tool.

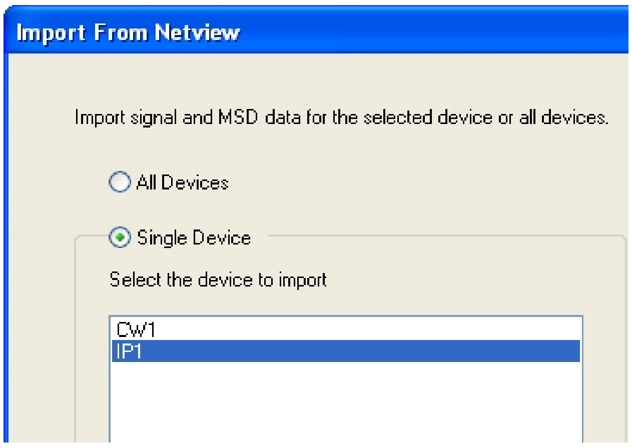
Options

System Set-up – Set-up driver and data collection behaviour.

Device Set-up – Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

Advanced – View and configure all Network 3000 driver and data collection properties.

3. When the “Import From NetView” dialog displays, select the **Single Device** option and click **Next>**. The wizard completes as described in Chapter 3 and adds the new signal to the ObjectServer database.


Import From Netview

Import signal and MSD data for the selected device or all devices.

○ All Devices

● Single Device

Select the device to import

CW1
IP1

Removing Signals/Variables

To delete signals/variables, follow the same instructions as for adding a signal, except that you need to **delete** the signal/variable from the control strategy program (rather than adding it) and then download the control strategy program back to the controller.

Use the NW3000 Set-up tool exactly as described above to remove the signal from the ObjectServer database.

Changing Signal Descriptions from within the Load File

When you originally created the control strategy file for a controller, you may not have included signal descriptors. Later, you can decide to provide descriptors for the signals in the load. This process describes how to change signal descriptions and update the ObjectServer database.

Adding Signal Descriptors

Adding descriptors to NW3000 or ControlWave signals involves two things:

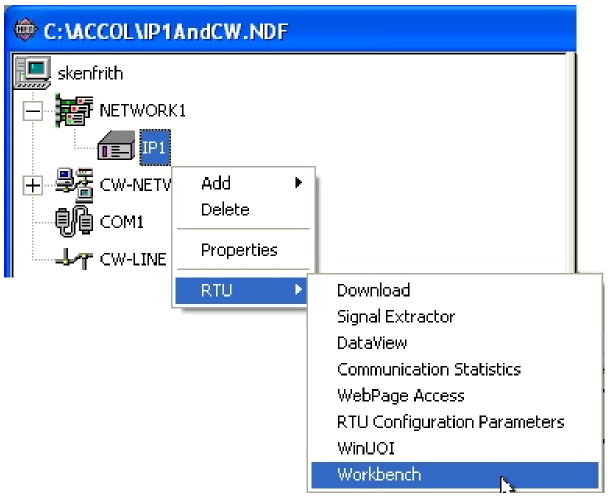
1. Update the signal descriptions in the control strategy file. The exact procedure differs for the NW3000 or ControlWave controller.

2. Run the Database Builder and Template Builder for the controller to update the ObjectServer database.

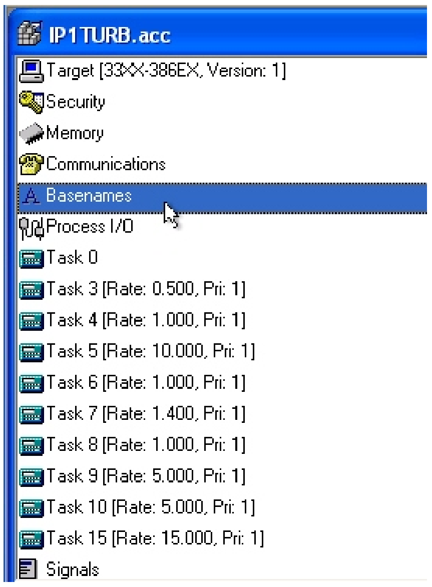
Here are the details.

Updating Basename Descriptors in ACCOL (NW3000 Controllers)

1. Open Workbench from NetView by right-clicking the controller that you want to change and selecting **RTU** and **Workbench** from the menus (as shown on the right). Alternately, you can start Workbench from the Windows **Start** button.

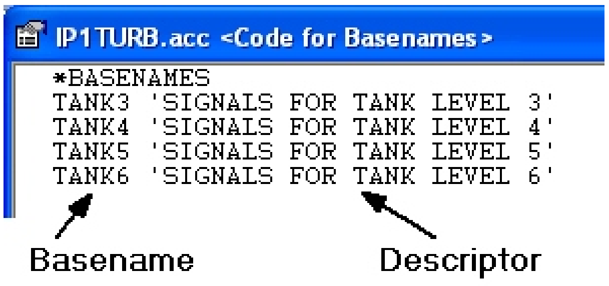


2. Double-click **Basenames** from the list of program entities Workbench displays.


IP1TURB.acc
Target [33XX-386EX, Version: 1]
Security
Memory
Communications
Basenames
Process I/O
Task 0
Task 3 [Rate: 0.500, Pri: 1]
Task 4 [Rate: 1.000, Pri: 1]
Task 5 [Rate: 10.000, Pri: 1]
Task 6 [Rate: 1.000, Pri: 1]
Task 7 [Rate: 1.400, Pri: 1]
Task 8 [Rate: 1.000, Pri: 1]
Task 9 [Rate: 5.000, Pri: 1]
Task 10 [Rate: 5.000, Pri: 1]
Task 15 [Rate: 15.000, Pri: 1]
Signals

3. Type the Basenames and Descriptors under the Basename section. Then build and download the file to the controller.

   For an explanation of "basename" see *An Introduction to ACCOL* (Document #D4056).



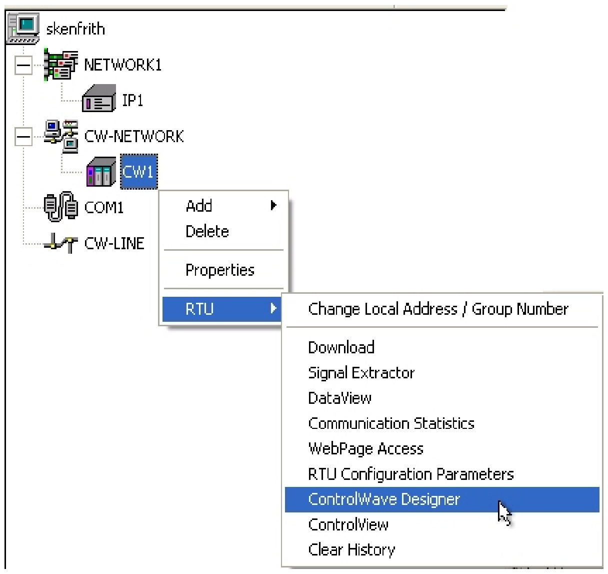
```
*BASENAMES
TANK3 'SIGNALS FOR TANK LEVEL 3'
TANK4 'SIGNALS FOR TANK LEVEL 4'
TANK5 'SIGNALS FOR TANK LEVEL 5'
TANK6 'SIGNALS FOR TANK LEVEL 6'
```

Now follow the instructions in the section *Getting Descriptor Changes into the ObjectServer Database* later in this chapter to get the new descriptors into the database.

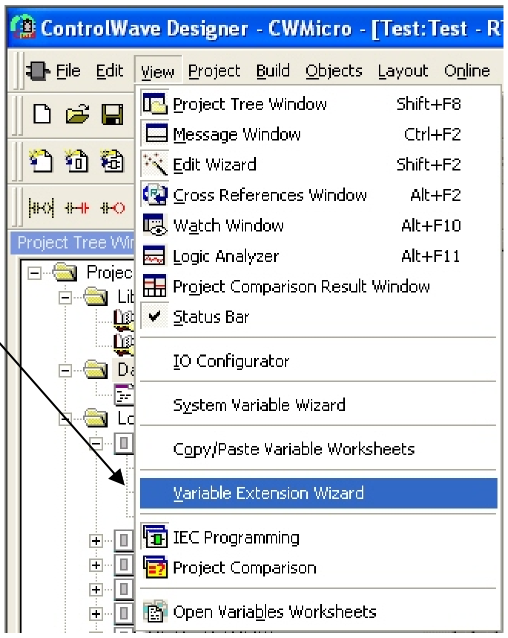
Updating Variable Descriptions with ControlWave Designer (ControlWave Controllers)

This procedure outlines the process to update the signal description in ControlWave Designer.

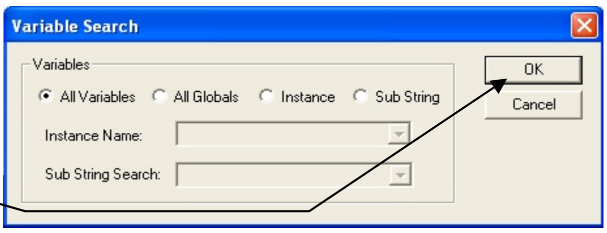
1. In NetView, right-click on the controller you want to change. Select **RTU → ControlWave Designer** (as show on the right) to open ControlWave Desiger. Alternatively, you can also open ControlWave Designer by selecting **Start → All Programs → OpenBSI Tools → ControlWave Tools → ControlWave Designer**.



2. Select **View → Variable Extension Wizard** from the ControlWave Designer menu bar.


ControlWave Designer - CWMicro - [Test:Test - R
File Edit View Project Build Objects Layout Online
Project Tree Wi
Projec
Lil
Da
Lo
Project Tree Window Shift+F8
Message Window Ctrl+F2
Edit Wizard Shift+F2
Cross References Window Alt+F2
Watch Window Alt+F10
Logic Analyzer Alt+F11
Project Comparison Result Window
Status Bar
IO Configurator
System Variable Wizard
Copy/Paste Variable Worksheets
Variable Extension Wizard
IEC Programming
Project Comparison
Open Variables Worksheets

3. Leave the **All Variables** option selected and click **OK**.



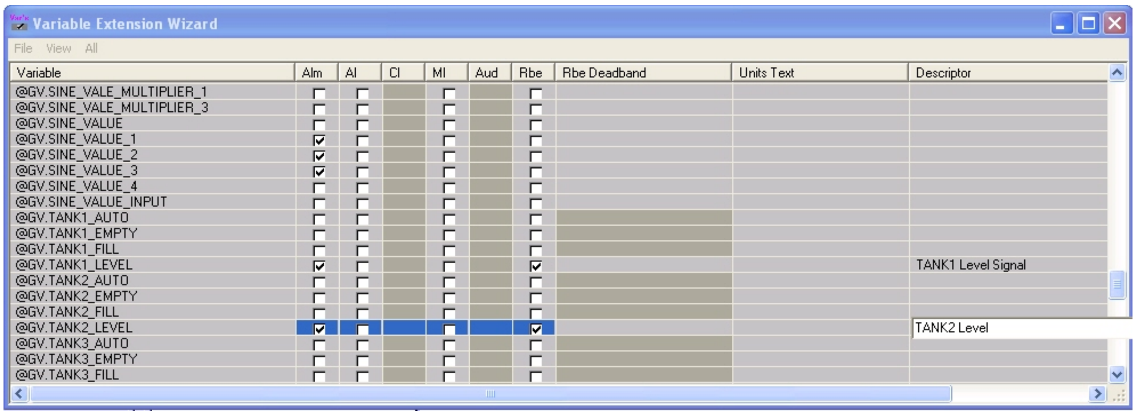
Variable Search

Variables: [✓ All Variables] All Globals Instance Sub String

Instance Name:

Sub String Search:

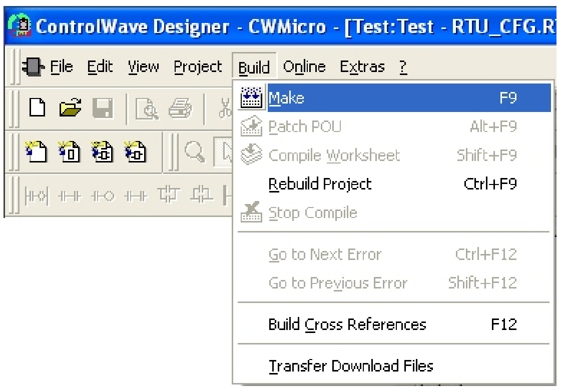
4. For each signal requiring a descriptor, select that signal's Descriptor field and enter the descriptor:



Variable Extension Wizard

Variable	Alm	AI	CI	MI	Aud	Rbe	Rbe Deadband	Units Text	Descriptor
@GV.SINE_VALE_MULTIPLIER_1	☐	☐		☐		☐			
@GV.SINE_VALE_MULTIPLIER_3	☐	☐		☐		☐			
@GV.SINE_VALUE	☐	☐		☐		☐			
@GV.SINE_VALUE_1	☑	☐		☐		☐			
@GV.SINE_VALUE_2	☑	☐		☐		☐			
@GV.SINE_VALUE_3	☑	☐		☐		☐			
@GV.SINE_VALUE_4	☐	☐		☐		☐			
@GV.SINE_VALUE_INPUT	☐	☐		☐		☐			
@GV.TANK1_AUTO	☐	☐		☐		☐			
@GV.TANK1_EMPTY	☐	☐		☐		☐			
@GV.TANK1_FILL	☐	☐		☐		☐			
@GV.TANK1_LEVEL	☑	☐		☐		☑			TANK1 Level Signal
@GV.TANK2_AUTO	☐	☐		☐		☐			
@GV.TANK2_EMPTY	☐	☐		☐		☐			
@GV.TANK2_FILL	☐	☐		☐		☐			
@GV.TANK2_LEVEL	☑	☐		☐		☑			TANK2 Level
@GV.TANK3_AUTO	☐	☐		☐		☐			
@GV.TANK3_EMPTY	☐	☐		☐		☐			
@GV.TANK3_FILL	☐	☐		☐		☐			

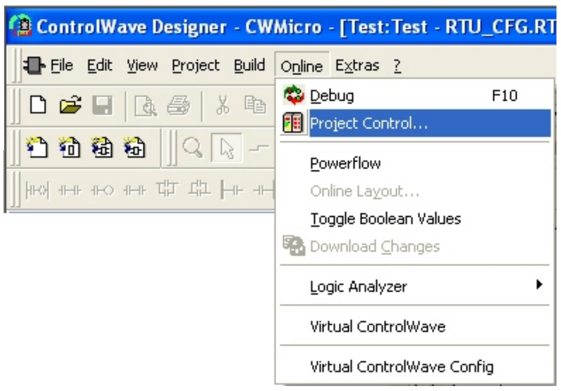
5. Select **File → Save** then **File → Exit** on the Variable Extension Wizard dialog. The ControlWave Designer screen displays. Select **Build → Make**. Ensure that the project compiles successfully without errors.



ControlWave Designer - CWMicro - [Test:Test - RTU_CFG.R

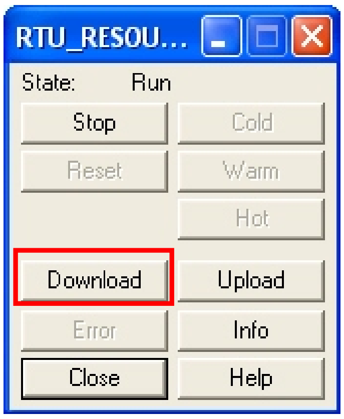
File Edit View Project Build Online Extras ?
Make F9
Patch POU Alt+F9
Compile Worksheet Shift+F9
Rebuild Project Ctrl+F9
Stop Compile
Go to Next Error Ctrl+F12
Go to Previous Error Shift+F12
Build Cross References F12
Transfer Download Files

6. Select **Online → Project Control…** from the ControlWave Designer menu bar.

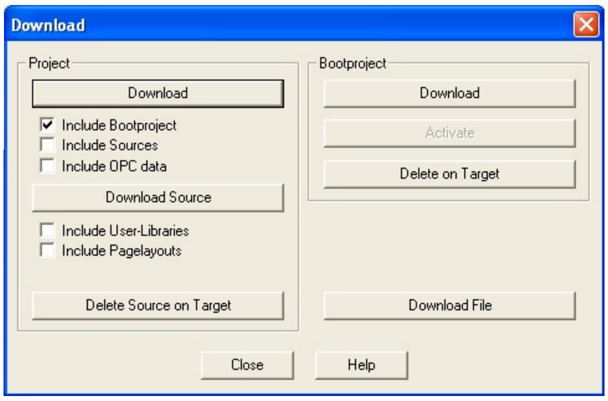


ControlWave Designer - CWMicro - [Test:Test - RTU_CFG.RT

7. Sign in, if required. Then click **Download** on the Project Control dialog. The Download dialog displays.



8. Click **Download** to download the control strategy file to the controller.



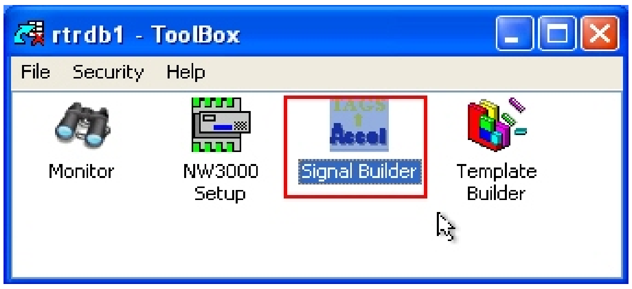
Now follow the instructions in the section *Getting Descriptor changes into the ObjectServer Database* to get the new descriptors into the database.

Getting Descriptor Changes into the ObjectServer Database

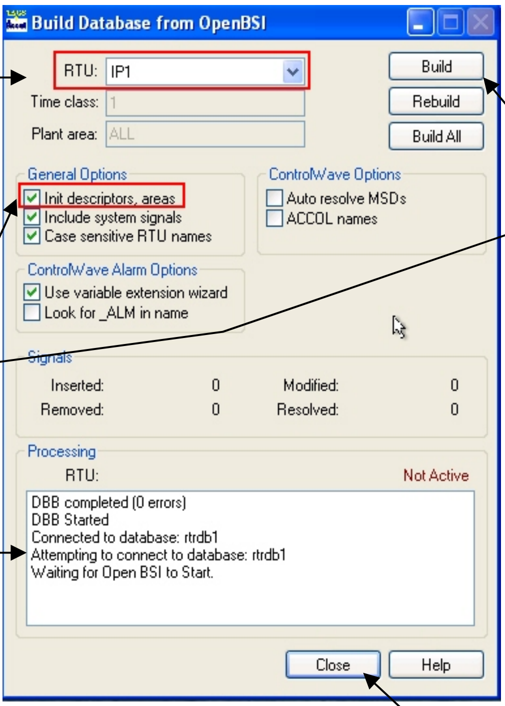
Until now you have updated the ObjectServer database using the Device Set-up wizard in the NW3000 Set-up tool. This wizard uses the Signal Builder and the Template Builder tools to update the database. This section explains how to use these tools directly to update the ObjectServer database.

Note: This explanation assumes you have already made changes to your control strategy file, and tested and downloaded it the controller.

1. Open the ObjectServer Toolbox and double-click the Signal Builder tool icon. The Build Database from OpenBSI dialog displays.



2. Select the RTU that you changed.
3. Select the **Init descriptors, areas** check box. **(This is critical!)**
4. Click **Build**.



```
DBB completed (0 errors)
DBB Started
Connected to database: rtrdb1
Attempting to connect to database: rtrdb1
Waiting for Open BSI to Start.
```

As the Signal Builder updates the database, various progress messages display here. When the process successfully completes, the message *DBB completed (0 errors)* displays at the top of the list.

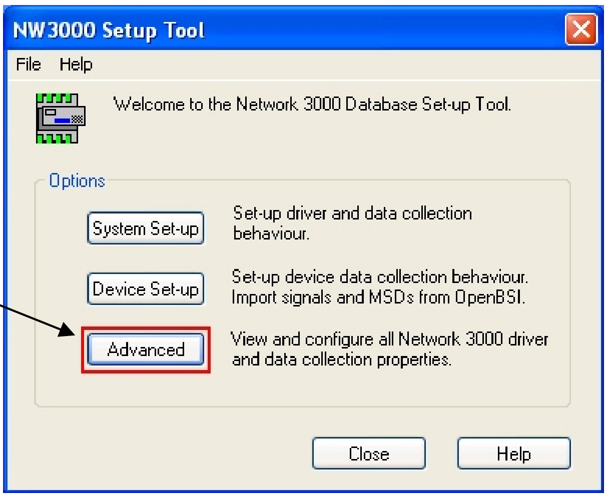
When the build process completes, click **Close**.

This adds the new descriptors from the control strategy file to the ObjectServer database.

Changing Data Collection Rates

If you used the default settings when you initially built the ObjectServer database, you may want to change the rate at which ObjectServer collects data from controllers. Do this using the NW3000 Setup tool.

1. Open the NW3000 Setup Tool from the ObjectServer Toolbox and click **Advanced** (highlighted in the image on the right).



Welcome to the Network 3000 Database Set-up Tool.

System Set-up: Set-up driver and data collection behaviour.

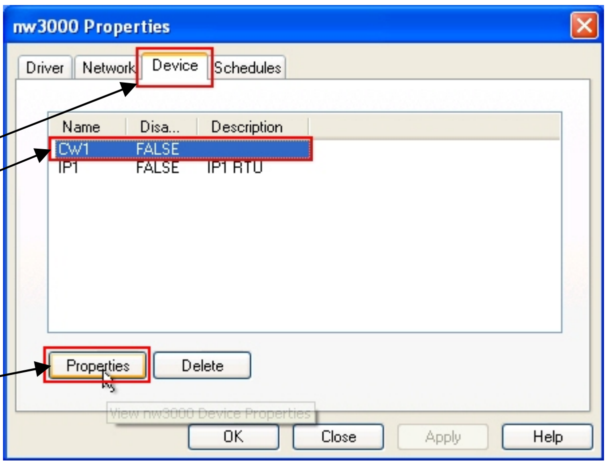
Device Set-up: Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.

Advanced: View and configure all Network 3000 driver and data collection properties.

2. When the nw3000 Properties dialog displays, select the Device tab (to view the collection schedule for each device).

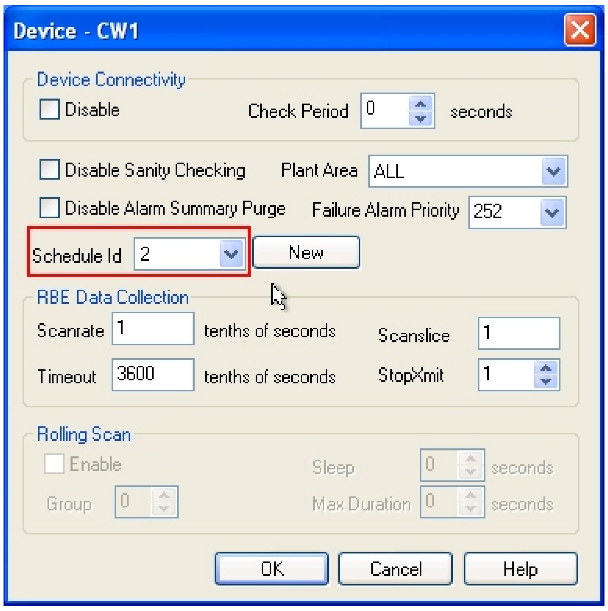
3. Select the controller that you want to change from the list of controllers.

4. Click **Properties**.

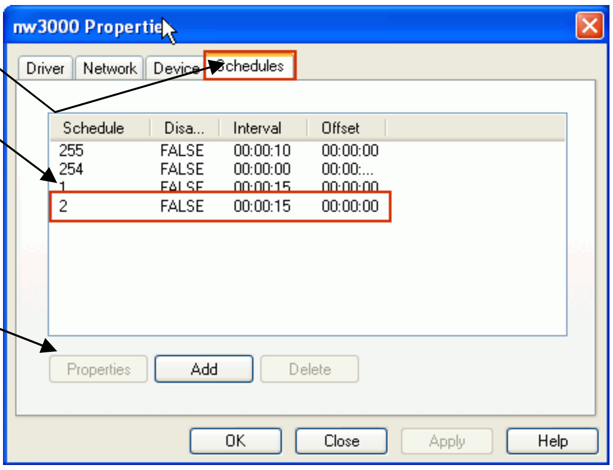


Name	Disa...	Description
CW1	FALSE	
IP1	FALSE	IP1 RTU

5. Note the value in the Schedule Id field for the device (highlighted to the right). In this example, the Schedule Id for the device is **2**. Click **OK** to return to the nw3000 Properties dialog.

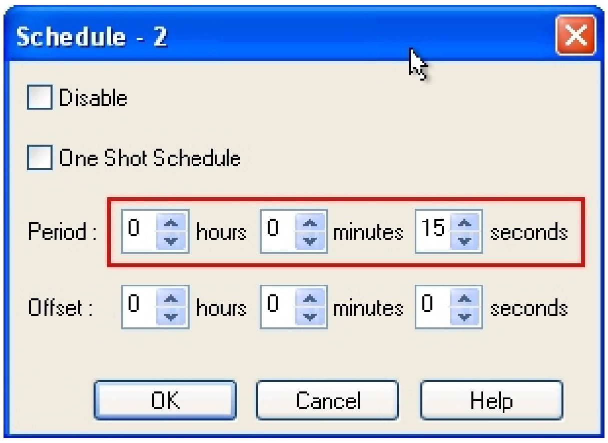


6. Now select the Schedules tab. Then select the schedule for the device that you are changing. Click **Properties**.

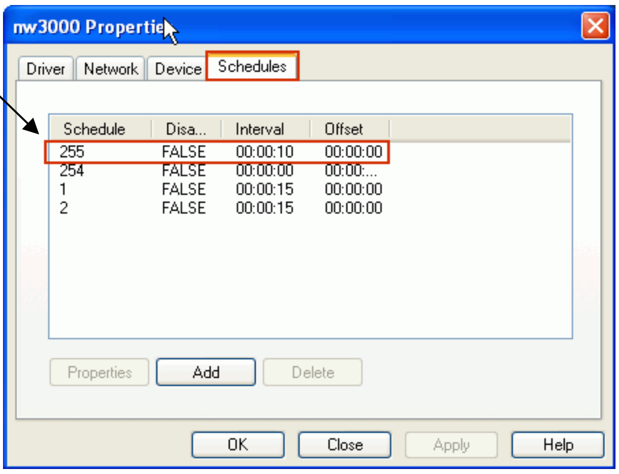


Schedule	Disa...	Interval	Offset
255	FALSE	00:00:10	00:00:00
254	FALSE	00:00:00	00:00:...
1	FALSE	00:00:15	00:00:00
2	FALSE	00:00:15	00:00:00

7. Now change the polling period for the schedule by changing the values in the hours, minutes, or seconds fields. You can either type values directly into these boxes or click ▼ and ▲. In this example, we have changed the schedule period from one minute to 15 seconds. When finished, click **OK** on this dialog, click **OK** on the nw3000 Properties dialog, and finally click **Close** on the NW3000 Setup Tool dialog. Job done!


Schedule - 2
Disable
One Shot Schedule
Period : 0 hours 0 minutes 15 seconds
Offset : 0 hours 0 minutes 0 seconds
OK
Cancel
Help

8. Note: Schedule number 255 is the Active Polling schedule. If an OPC client requests a tag for a signal, the system automatically (but temporarily) assigns that signal to this schedule as well as the one to which it normally belongs. This ensures faster updates for that signal. By default, the Active Polling Schedule polls every 10 seconds.



Schedule	Disa...	Interval	Offset
255	FALSE	00:00:10	00:00:00
254	FALSE	00:00:00	00:00:...
1	FALSE	00:00:15	00:00:00
2	FALSE	00:00:15	00:00:00

Making Changes to RBE Settings

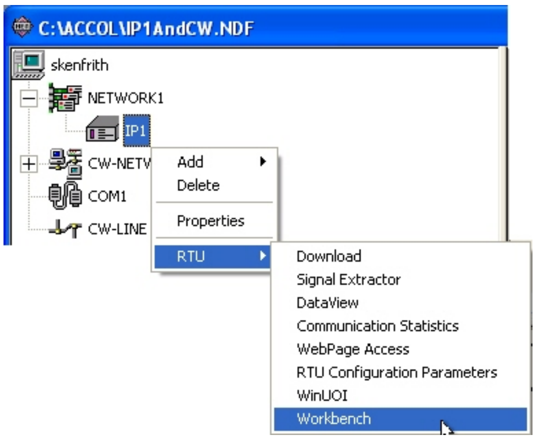
Sometimes you need to change the default RBE settings for a system. The network may be struggling, and you may need to experimentally alter RBE settings until you get them right.

Accessing and changing these values from ObjectServer prevents you from having to keep changing and downloading the RBE settings in the control strategy file.

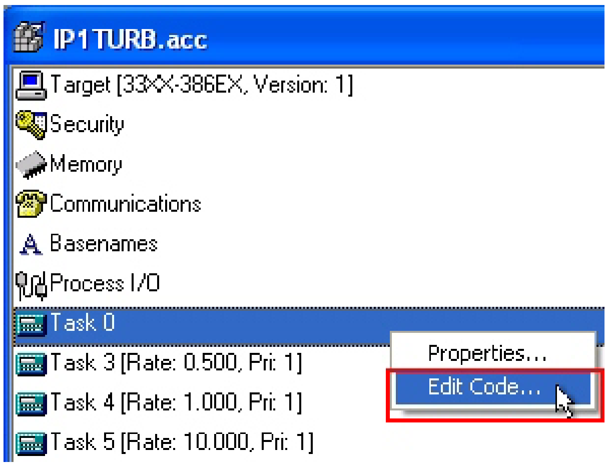
Setting the RBE Mode in Workbench (ACCOL)

Set the RBE mode in your control strategy file to zero (0) in order to change the rest of the RBE settings on the fly from ObjectServer.

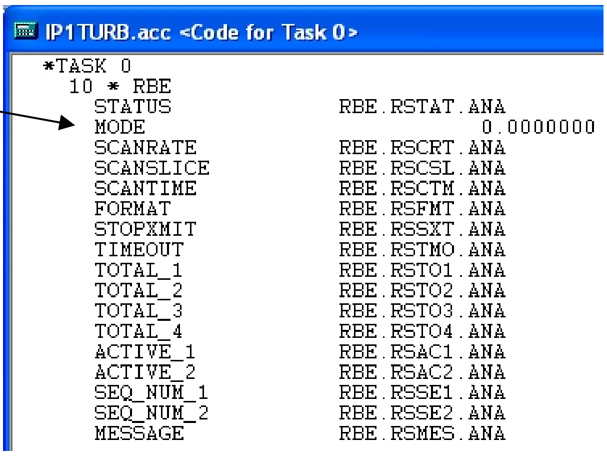
1. Select the controller you want to change in NetView, right-click and select **Workbench**.



2. Select **Task 0** from the Workbench interface, then right-click and select **Edit Code**. Task 0 has no rate, and is always used for the RBE module.

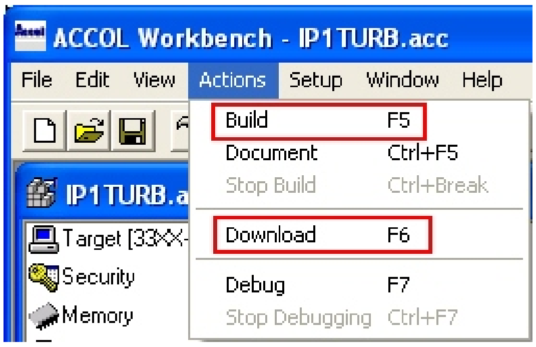

IP1TURB.acc
Target [33XX-386EX, Version: 1]
Security
Memory
Communications
Basenames
Process I/O
Task 0
Task 3 [Rate: 0.500, Pri: 1]
Task 4 [Rate: 1.000, Pri: 1]
Task 5 [Rate: 10.000, Pri: 1]
Properties...
Edit Code...

3. Change the MODE value to zero (0.0) as shown here.



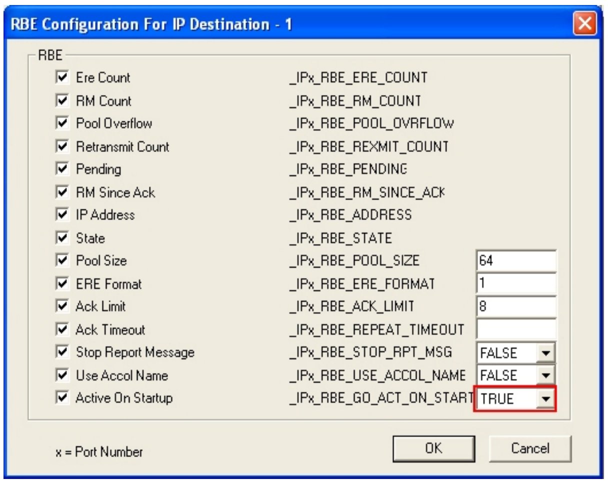
```
*TASK 0
  10 * RBE
    STATUS                    RBE.RSTAT.ANA
    MODE                              0.0000000
    SCANRATE                  RBE.RSCRT.ANA
    SCANSLICE                 RBE.RSCSL.ANA
    SCANTIME                  RBE.RSCTM.ANA
    FORMAT                    RBE.RSFMT.ANA
    STOPXMIT                  RBE.RSSXT.ANA
    TIMEOUT                   RBE.RSTMO.ANA
    TOTAL_1                   RBE.RSTO1.ANA
    TOTAL_2                   RBE.RSTO2.ANA
    TOTAL_3                   RBE.RSTO3.ANA
    TOTAL_4                   RBE.RSTO4.ANA
    ACTIVE_1                  RBE.RSAC1.ANA
    ACTIVE_2                  RBE.RSAC2.ANA
    SEQ_NUM_1                 RBE.RSSE1.ANA
    SEQ_NUM_2                 RBE.RSSE2.ANA
    MESSAGE                   RBE.RSMES.ANA
```

4. Build the load and download it to the controller.



Setting the RBE Module in ControlWave Designer

1. Set the Active On Startup value to **TRUE**.



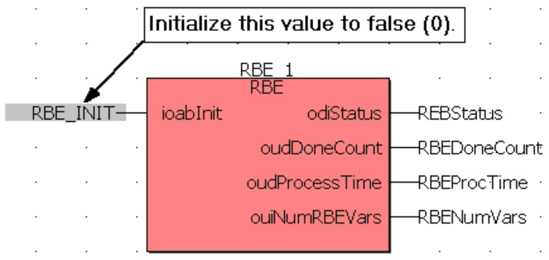
RBE Configuration For IP Destination - 1

RBE

☑ Ere Count	_IPx_RBE_ERE_COUNT	
☑ RM Count	_IPx_RBE_RM_COUNT	
☑ Pool Overflow	_IPx_RBE_POOL_OVRFLOW	
☑ Retransmit Count	_IPx_RBE_REXMIT_COUNT	
☑ Pending	_IPx_RBE_PENDING	
☑ RM Since Ack	_IPx_RBE_RM_SINCE_ACK	
☑ IP Address	_IPx_RBE_ADDRESS	
☑ State	_IPx_RBE_STATE	
☑ Pool Size	_IPx_RBE_POOL_SIZE	64
☑ ERE Format	_IPx_RBE_ERE_FORMAT	1
☑ Ack Limit	_IPx_RBE_ACK_LIMIT	8
☑ Ack Timeout	_IPx_RBE_REPEAT_TIMEOUT	
☑ Stop Report Message	_IPx_RBE_STOP_RPT_MSG	FALSE
☑ Use Accol Name	_IPx_RBE_USE_ACCOL_NAME	FALSE
☑ Active On Startup	_IPx_RBE_GO_ACT_ON_START	TRUE

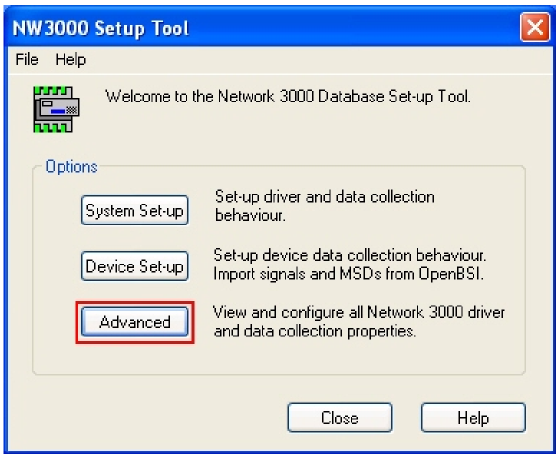
x = Port Number

2. Initialize the ioabInit input of the RBE module to zero.



Updating RBE Settings in ObjectServer

1. Open the NW3000 Setup Tool from the ObjectServer Toolbox and select **Advanced**.



NW3000 Setup Tool

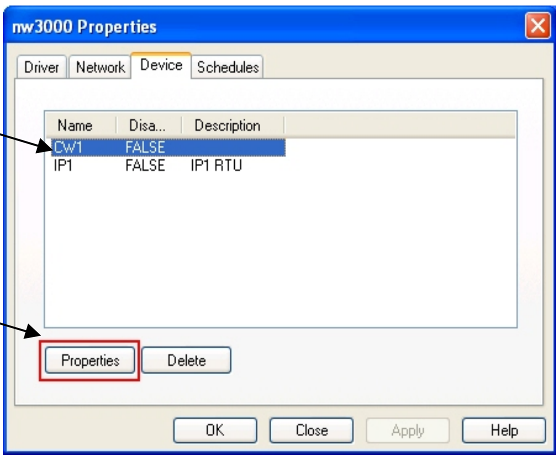
Welcome to the Network 3000 Database Set-up Tool.

Options

System Set-up	Set-up driver and data collection behaviour.
Device Set-up	Set-up device data collection behaviour. Import signals and MSDs from OpenBSI.
Advanced	View and configure all Network 3000 driver and data collection properties.

2. Select the Device tab and then select the controller that has the modified RBE module.

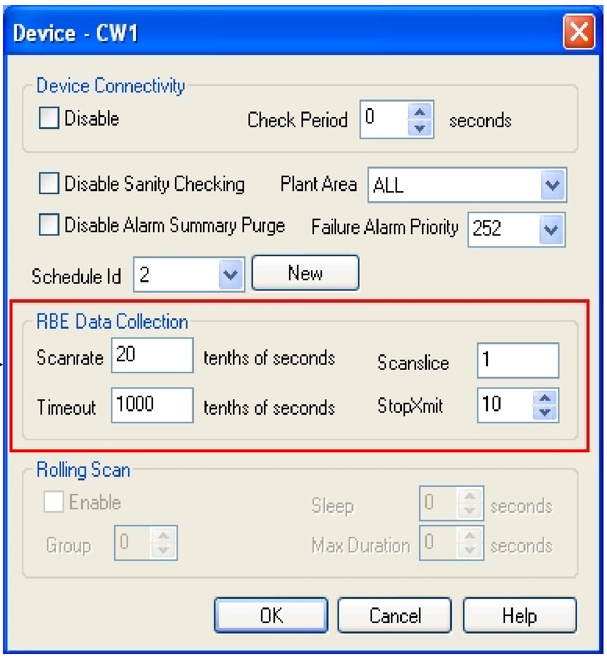
   Then click **Properties**.



nw3000 Properties

Name	Disa...	Description
CW1	FALSE	
IP1	FALSE	IP1 RTU

3. On the Device dialog, set the RBE options in the RBE Data Collection section. Click the [Help] button the dialog for an explanation of each option.



This chapter reviewed the processes you can use to modify your system as it grows and changes:

- Adding controllers
- Removing controllers
- Adding signals/variables
- Removing signals/variables
- Adding signal/variable descriptions
- Changing data collection rates
- Changing RBE data collection

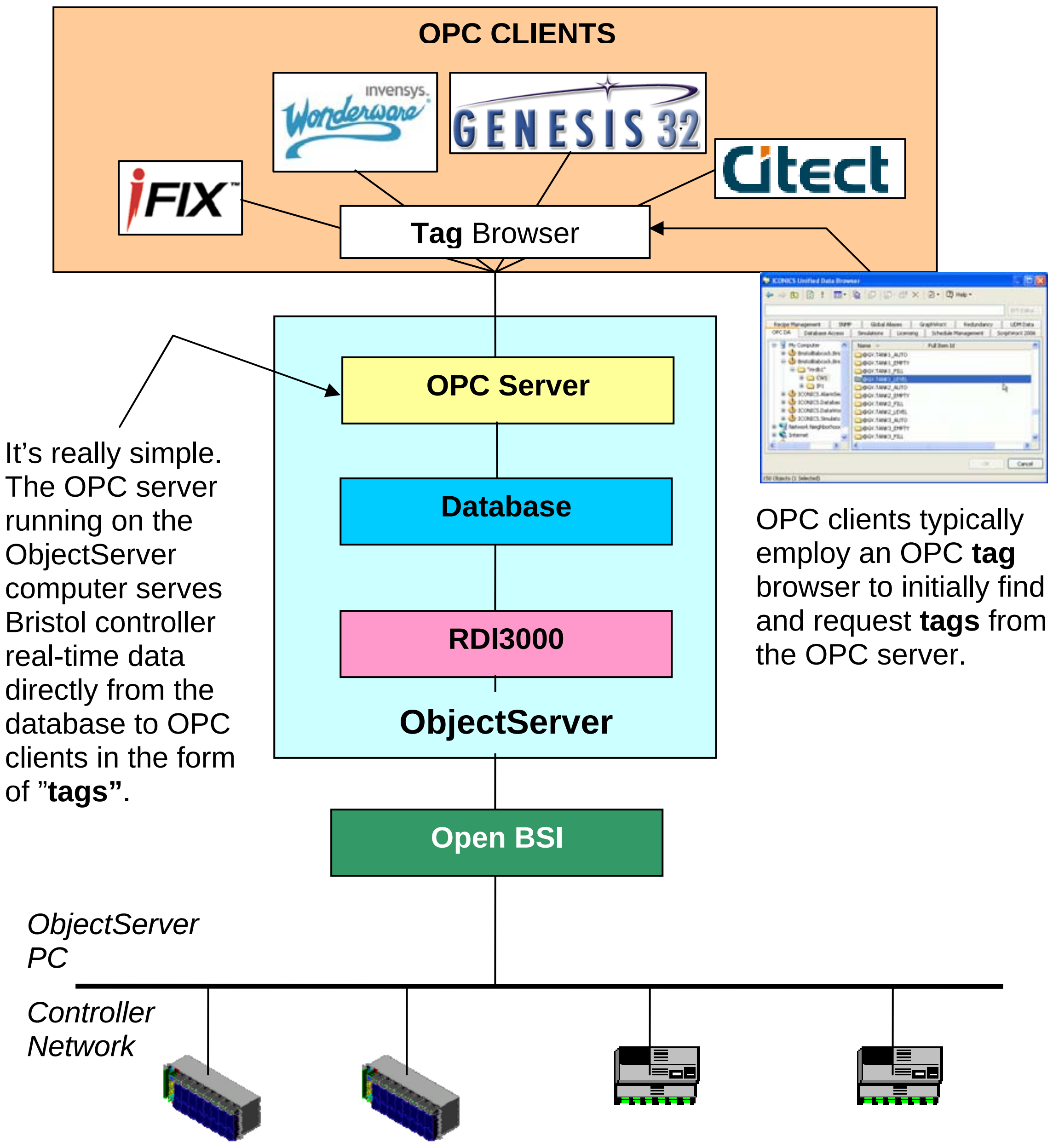
[This page is intentionally left blank.]

Chapter 7 – Displaying ObjectServer Data in OPC Clients

This chapter explains how to configure a number of OPC clients to display ObjectServer data. OPC clients are also called HMIs, because they provide a ”human-machine interface” onto process control data.

Getting ObjectServer Real-time Data into OPC Clients on the Same Computer

In Chapter 1 of this manual we saw how the data got from the Bristol controllers into the ObjectServer database. We can now look at how the data gets from the ObjectServer database to the OPC client. Firstly, we see how this happens when the client is running on the **same** PC as ObjectServer.



It’s really simple. The OPC server running on the ObjectServer computer serves Bristol controller real-time data directly from the database to OPC clients in the form of ”**tags”**.

OPC clients typically employ an OPC **tag** browser to initially find and request **tags** from the OPC server.

Help – What's a Tag?

At the most basic level, a "tag" is a label that identifies an object. For instance, we tie labels to our luggage when we go on holiday so that if the luggage gets lost, the airport people can identify it as ours.

According to the dictionary, a tag can also be described as:-

a. A label assigned to identify data in a computer's memory.
b. A sequence of characters in a markup language used to provide information, such as formatting specifications, about a document.

These definitions belong to the disciplines of computer programming and web development respectively.

However, these definitions are also useful in describing what an OPC tag is.

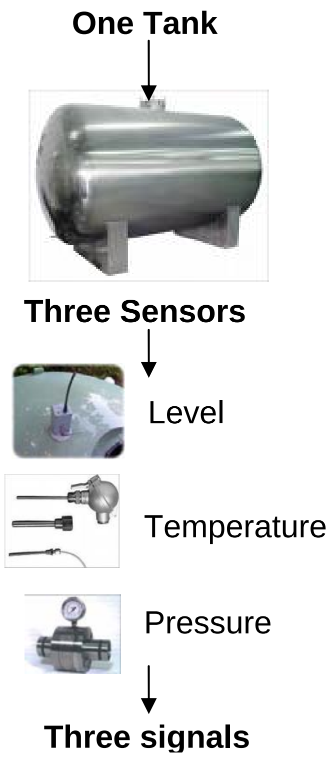
DEFINITION: An OPC tag can be seen as a coded label that describes a single property of an object (= signal = variable) within an OPC server's object store.

Objects and ObjectServer

As we said in Chapter 1 of this manual, on a very basic level, the objects stored in ObjectServer are pumps, tanks, valves, and so on.

However, process control systems are more interested in the **state** of each object in the system. We obtain this data by attaching sensors to the objects, which send signals to our RTU. A simple valve only requires one sensor to indicate that it is open or closed. However, we may need more than one sensor to understand the state of other objects.

For instance, we may need to know the level, temperature, and pressure of fluid inside a tank. We attach sensors to the tank that signal these measurements to the RTU. In effect, each of these three sensors is represented in the database as a separate signal object, but they reference the same physical object



How do OPC Servers Define Signal Data?

We have established that the objects in our ObjectServer process control system are more correctly understood as sensors that are attached to actual physical objects. Because sensors communicate via electronic signals, they are known as “signals” to ObjectServer.

Each signal has a **value**. Our **TANK1.LEVEL.** signal is the height of fluid in our tank, the **TANK1.PRES.** signal is the pressure in the tank, and **TANK1.TEMP.** signal is the temperature in the tank.

But besides value, each signal also has a number of other properties. These can be its **name** (TANK1.LEVEL.) which distinguishes it from other signals and its **units** property (such as metres) which tells us the units in which its **value** is measured.

In fact, non-alarm Bristol controller signals have a total of **seven** properties the ObjectServer OPC server can expose: **name, value, description, units, questionable, controlinhibit, and manualinhibit**. Bristol alarm signals have 11 additional properties, relating to their alarm status. The OPC Server makes each property of each signal available to our OPC client as an OPC tag. So what is an OPC tag again?

An OPC Tag is a Query

OPC clients display OPC data by requesting the data from OPC servers in the form of an OPC tag. So, an OPC tag is actually a type of query.

Databases enable us to get at the data they store by permitting us to query that data. Query methods have evolved with the needs of applications. Following are three examples of how query methods are employed for different purposes.

The first of the three example query methods applies to a flat file database in the form of an address book. This query method returns data as collections of records starting with the same letter.

The second example method queries a relational database as the rows and columns of a table. The third example method shows how OPC tags return the value of a single attribute to the OPC client, for display as a shape or text object.

The objects ObjectServed serves are actually single properties of objects that are visualized as drawn objects on a display. Each OPC tag on the client requests a single property or attribute of a signal in our database.

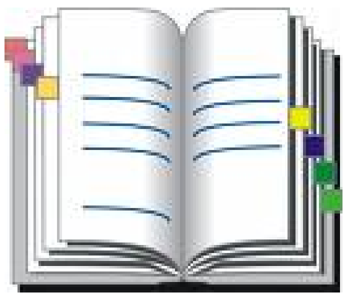
Alphabetical Query

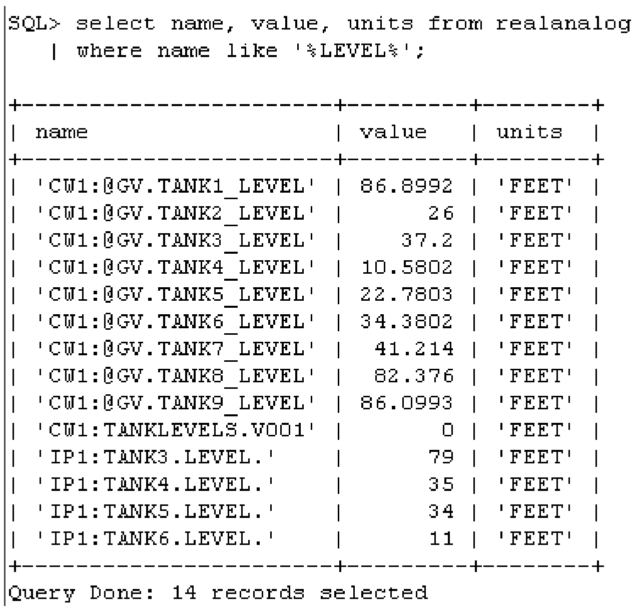
The simplest of databases is an address book. This type of database is known as a flat file database. It has only one table.

We query this kind of database by grouping the entries alphabetically. This kind of query enables us to find what we want in two steps:-

1. We find a group of records which begins with the same letter
2. We search down these records to find the exact one we want.

This kind of query returns a record object with all of its attributes (such as a name, an address, or a phone number).





```
SQL> select name, value, units from realanalog
   | where name like '%LEVEL%';

+-----------------------+---------+--------+
| name                  | value   | units  |
+-----------------------+---------+--------+
| 'CW1:@GV.TANK1_LEVEL' | 86.8992 | 'FEET' |
| 'CW1:@GV.TANK2_LEVEL' |      26 | 'FEET' |
| 'CW1:@GV.TANK3_LEVEL' |    37.2 | 'FEET' |
| 'CW1:@GV.TANK4_LEVEL' | 10.5802 | 'FEET' |
| 'CW1:@GV.TANK5_LEVEL' | 22.7803 | 'FEET' |
| 'CW1:@GV.TANK6_LEVEL' | 34.3802 | 'FEET' |
| 'CW1:@GV.TANK7_LEVEL' |  41.214 | 'FEET' |
| 'CW1:@GV.TANK8_LEVEL' |  82.376 | 'FEET' |
| 'CW1:@GV.TANK9_LEVEL' | 86.0993 | 'FEET' |
| 'CW1:TANKLEVELS.V001' |       0 | 'FEET' |
| 'IP1:TANK3.LEVEL.'    |      79 | 'FEET' |
| 'IP1:TANK4.LEVEL.'    |      35 | 'FEET' |
| 'IP1:TANK5.LEVEL.'    |      34 | 'FEET' |
| 'IP1:TANK6.LEVEL.'    |      11 | 'FEET' |
+-----------------------+---------+--------+
Query Done: 14 records selected
```

SQL Query

Relational databases make queries a much more complex task, so complex in fact that it requires a language to formulate the query. This language is called Structured Query Language (SQL).

SQL enables us to search for one or more attributes of one or more objects in one or more tables.

The example on the left shows an SQL query sent to the ObjectServer database using the SQL client. The results of an SQL query are always shown as a table with rows and columns.

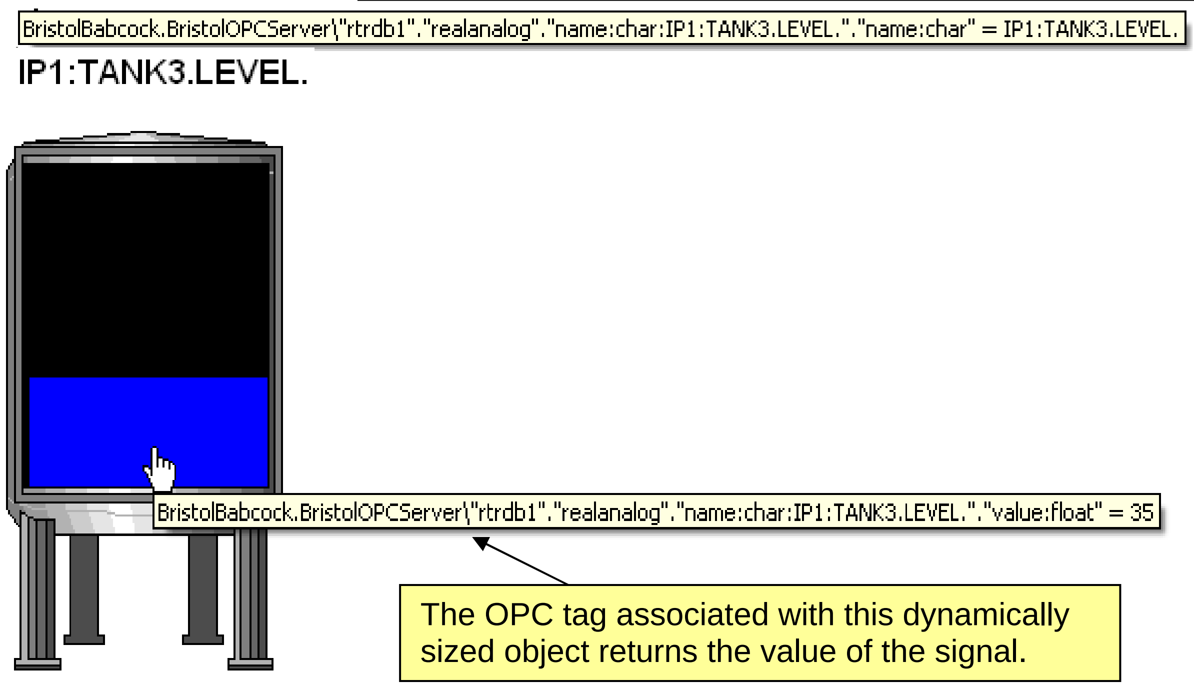
In this type of query, an " object" is understood to be a single row from the returned table. Each row in our example displays three properties for each object (signal).

Object Query

OPC Data Access clients display objects visually as shapes or text. Display objects can be static or dynamic. Each dynamic visual object is linked to the value of single attribute in the database. The “object: that ObjectServer serves up here is actually a single attribute (not a row) from the ObjectServer database.

For each OPC tag registered by the client display, the OPC server sends an Object Query (which requests a single attribute) from the ObjectServer database. The value requested is updated on the Client display every few seconds.

The OPC tag associated with this text object returns the name of the signal.



```
BristolBabcock.BristolOPCServer\"rtrdb1"."realanalog"."name:char:IP1:TANK3.LEVEL."."name:char" = IP1:TANK3.LEVEL.
```

IP1:TANK3.LEVEL.

```
BristolBabcock.BristolOPCServer\"rtrdb1"."realanalog"."name:char:IP1:TANK3.LEVEL."."value:float" = 35
```

The OPC tag associated with this dynamically sized object returns the value of the signal.

Preparation

Before you attempt to configure any OPC client HMI to display ObjectServer data, you must make sure that ObjectServer is running and collecting data from your controllers. You can also start the OPC server manually, but the OPC client should do this for you if you don’t. Select **Start → Run,** then type **BristolOPCServer.exe** into the Open field, and click **OK**..

Getting ObjectServer Real-time Data into Genesis32™

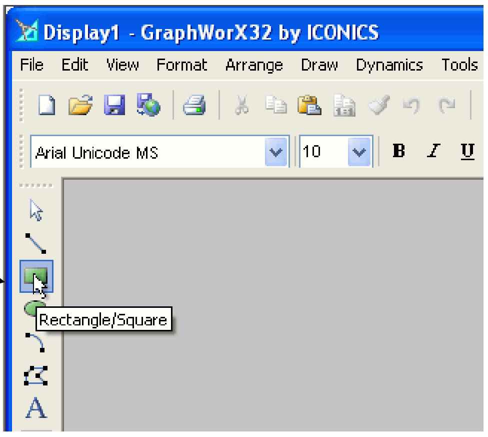
Genesis32 is an OPC data access client application developed by ICONICS, Inc. Here's how to get ObjectServer data into a GraphWorX™32 display (GraphWorX32 is the display tool included within Genesis32).

NOTE

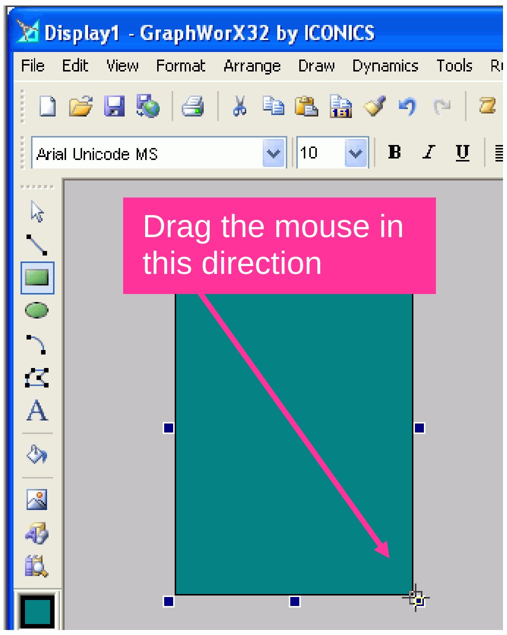
What follows is not a detailed description of how to use every feature of GraphWorX32. It is intended only as an example to show how to get tags from the ObjectServer OPC server into GraphWorX32. Please use the online help provided with GraphWorX32 for information on any other feature.

We are going to display a tank level on our display as a rectangle, so that the height of the rectangle represents the value of the signal. You can use any real analog signal having a value that fluctuates between a defined maximum and minimum value.

1. First, we open GraphWorX32 and select the rectangle tool

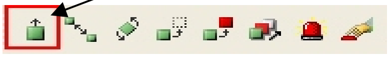

Display1 - GraphWorX32 by ICONICS
File Edit View Format Arrange Draw Dynamics Tools
Arial Unicode MS
10
B I U
Rectangle/Square

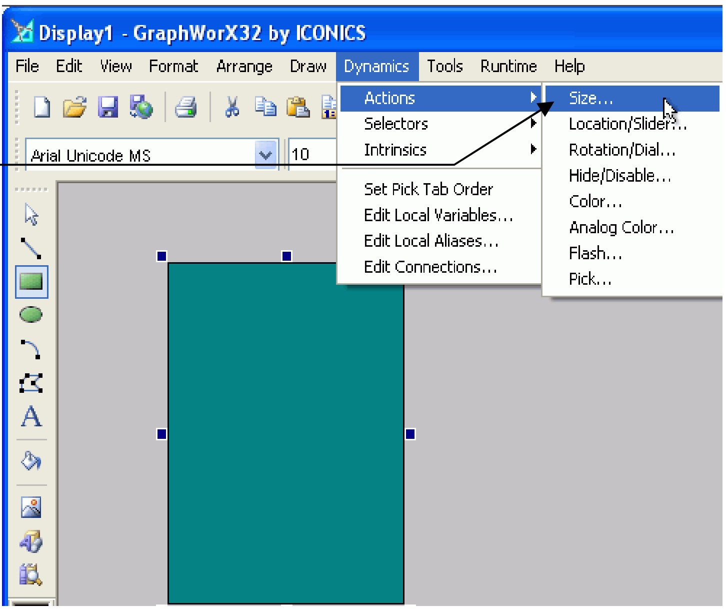
2. Then we click on the display area, and create the rectangle. We now have a visible rectangle object, but it is not yet connected through OPC with our signal value. To do this, we need to make this into a dynamic object.


Display1 - GraphWorX32 by ICONICS
File Edit View Format Arrange Draw Dynamics Tools R
Arial Unicode MS
10
B I U
Drag the mouse in this direction

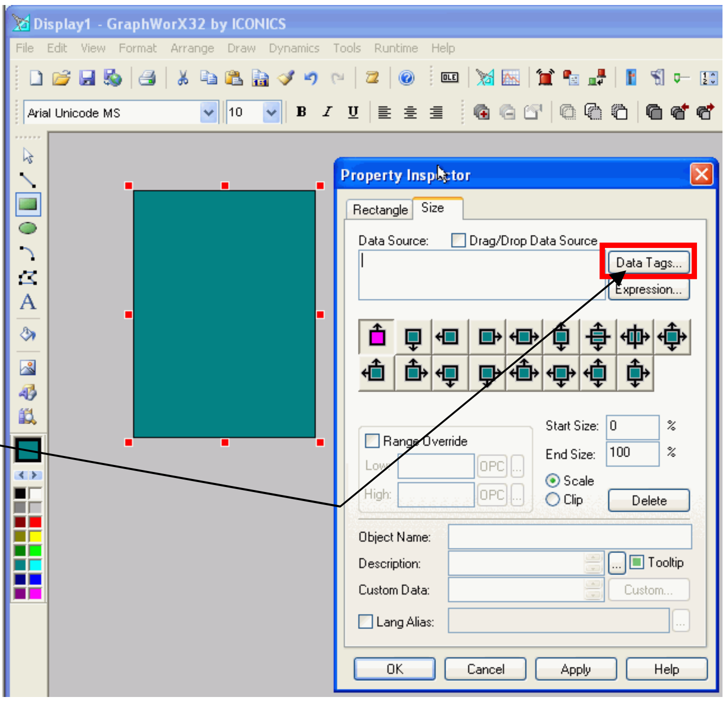
3. With the rectangle still selected, select **Dynamics → Actions → Size** from the menu bar.

   You could also click on the Size dynamic button on the dynamic toolbar.





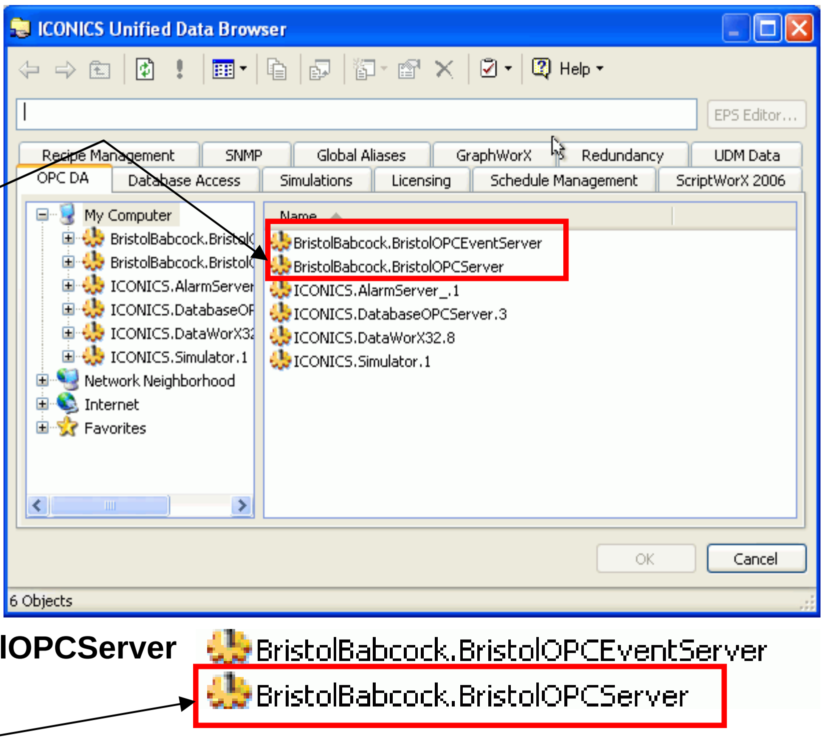
4. The rectangle's Property Inspector dialog displays with the new Size tab added, and the cursor inside the Data Source box. We can now browse for Tags by clicking **Data Tags**.



5. The ICONICS Unified Data Browser searches for the available OPC servers. We can see two ObjectServer OPC servers outlined in red.

   Make sure you select the **right** OPC Server. You need real-time data, not alarm/event data.

   Double-click on the **BristolBabcock.BristolOPCServer**

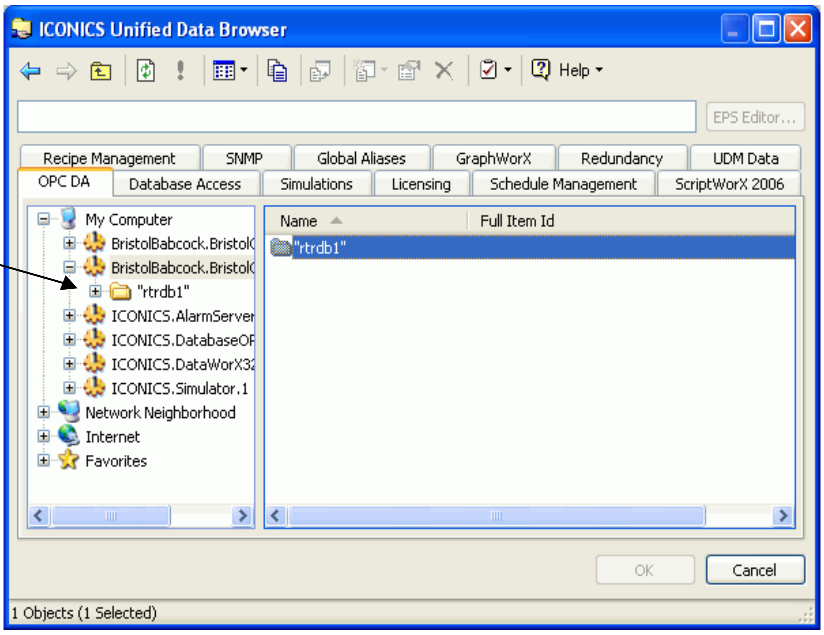


BristolBabcock.BristolOPCEventServer
BristolBabcock.BristolOPCServer
ICONICS.AlarmServer_.1
ICONICS.DatabaseOPCServer.3
ICONICS.DataWorX32.8
ICONICS.Simulator.1

6 Objects

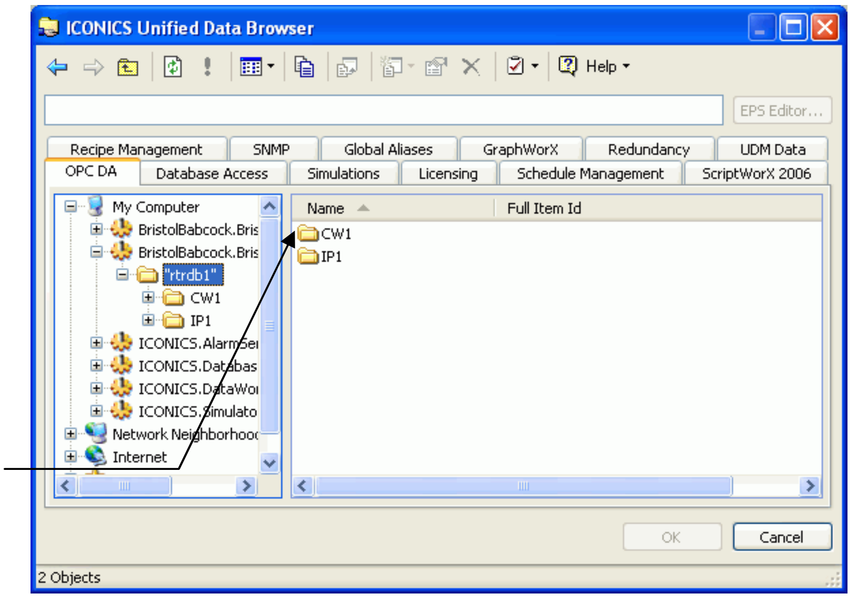
6. Now, in the left pane, click the plus sign to the left of the “rtrdb1” folder. This folder represents the whole ObjectServer Database

   We are drilling down to get at the tag we want.

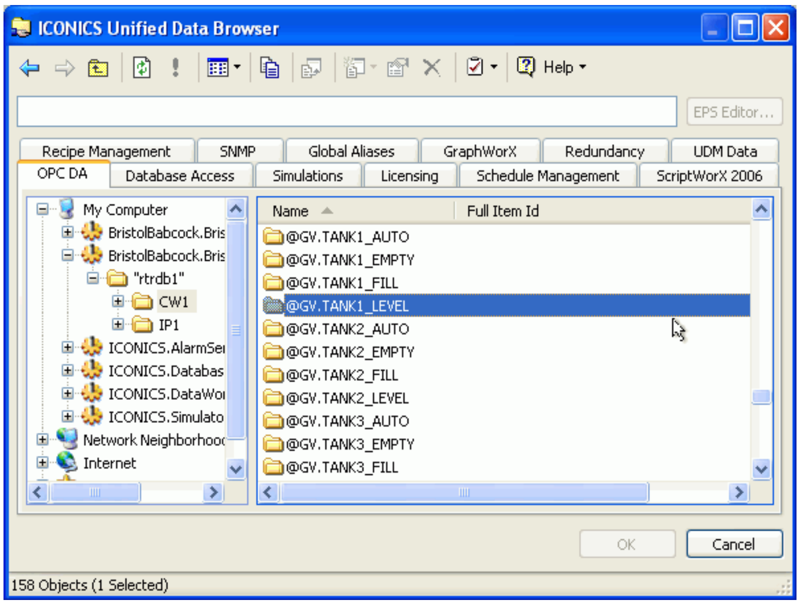


1 Objects (1 Selected)

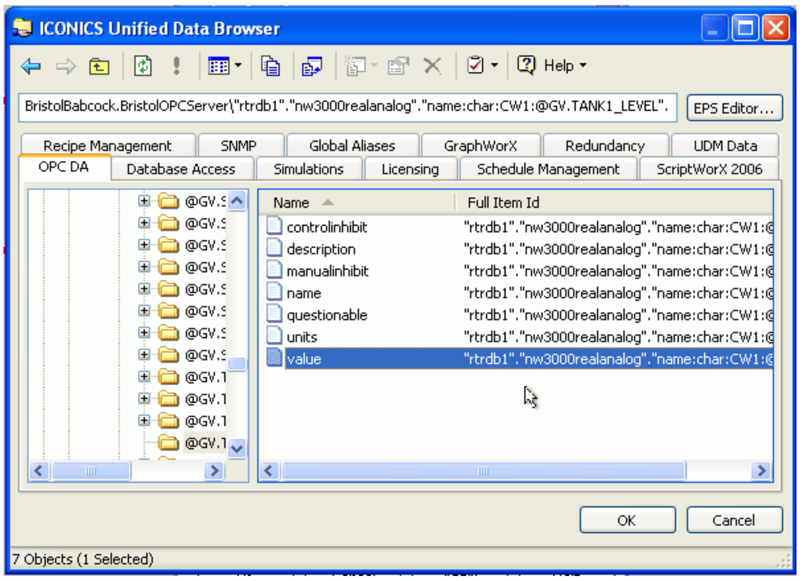
7. Now we can see the available controllers . In the right pane double-click controller that contains the signal you want. In this example we double-click controller 'CW1'.



8. We can now see lots of signals that belong to the controller. We need to find the signal we are after, and then double click on it. For this example, we use TANK1_LEVEL.



9. This exposes the individual tags for the signal we want. Select the 'value' tag, then click OK. This re-displays the Property Inspector dialog.



BristolBabcock.BristolOPCServer\"rtdb1"."nw3000realanalog"."name:char:CW1:@GV.TANK1_LEVEL".

Name	Full Item Id
controlinhibit	"rtdb1"."nw3000realanalog"."name:char:CW1:@
description	"rtdb1"."nw3000realanalog"."name:char:CW1:@
manualinhibit	"rtdb1"."nw3000realanalog"."name:char:CW1:@
name	"rtdb1"."nw3000realanalog"."name:char:CW1:@
questionable	"rtdb1"."nw3000realanalog"."name:char:CW1:@
units	"rtdb1"."nw3000realanalog"."name:char:CW1:@
value	"rtdb1"."nw3000realanalog"."name:char:CW1:@

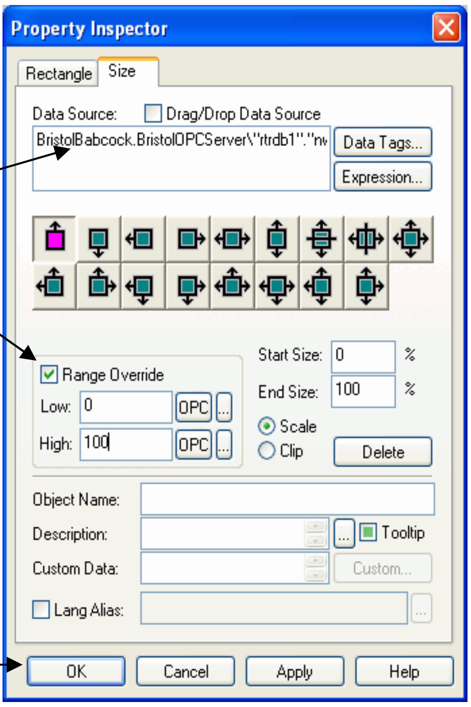
7 Objects (1 Selected)

Why do Different Signals Have Different Tags?

The OPC server serves different OPC tags depending on the signal type. Tags for Bristol controller **non-alarm** realanalog signals include controlinhibit, manualinhibit, description, name, questionable, units, and value. Realanalog **alarm** signals use all of these plus **highlimit** and **lowlimit** tags.

Note: The tag is automatically written into the Data Source field.

10. Select the Range Override check box and enter in the Low and High values for the signal value. Then click **OK**.



Data Source: ☐ Drag/Drop Data Source

BristolBabcock.BristolOPCServer\"rtdb1"."nw

☑ Range Override

Low: 0

High: 100

Start Size: 0 %

End Size: 100 %

● Scale ○ Clip

Object Name:

Description:

Custom Data:

☐ Lang Alias:

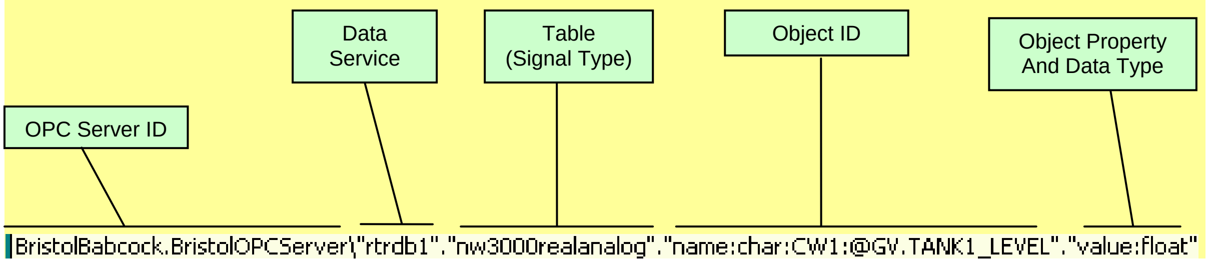
More about OPC Tags

An OPC tag is a string that enables the OPC server to serve object values that OPC clients request.

The actual format of the OPC tag used differs between OPC servers, but the format is typically <*OPCServerProgID*>\<*OPCItem*>. The <*OPCServerProgID*> portion of the tag is always present, followed by a backslash delimiter (\). The OPC Item part of the tag defines an object property. The number of elements in the part of the tag that defines the OPC Item may differ widely for each OPC server. For instance, an ObjectServer OPC Item includes the data-service, table name, signal name, and signal property. You delimit each part of this section of the tag with a period.

An Example ObjectServer OPC Tag

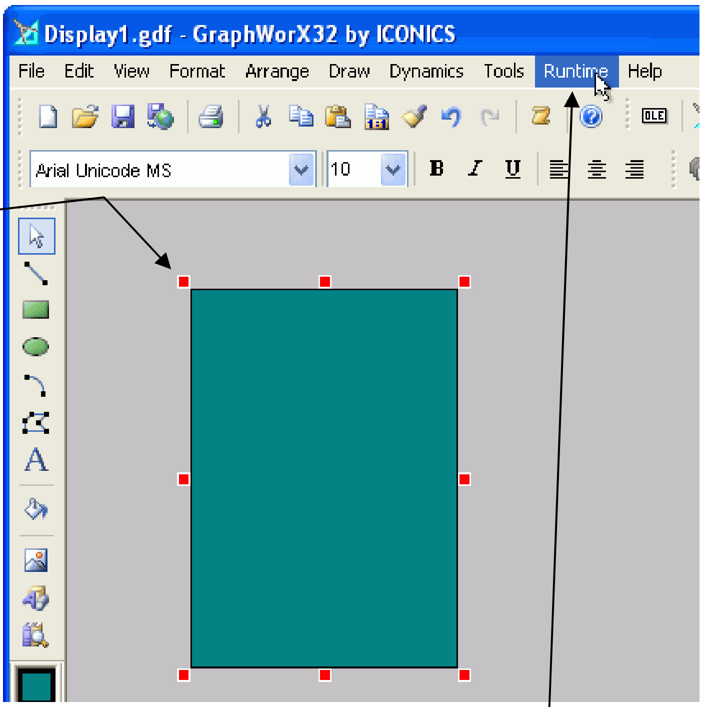
Here's an example ObjectServer OPC tag. Labels on each part of the tag explain what that part identifies. Once you know the tag structure, you can use this information to copy and modify tags textually rather than having to use the tag browser every time.



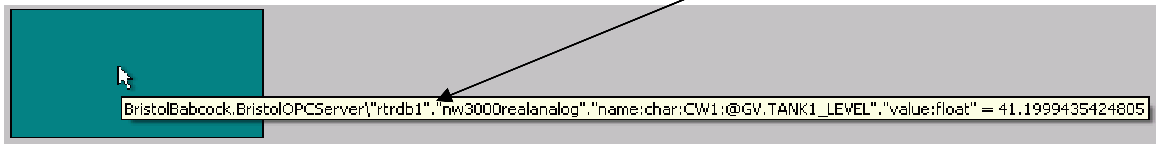
BristolBabcock.BristolOPCServer\"rtrdb1"."nw3000realanalog"."name:char:CW1:@GV.TANK1_LEVEL"."value:float"

11. Back on the display page, we can see that the selected rectangle ,has red dots around it. These, signify that it is a dynamic object, now associated with an OPC tag value.

    We need to put the display into Runtime mode to see the dynamic object in action. Select the Runtime menu option, highlighted here.



12. In Runtime mode, the rectangle's height adjusts to represent the value of the tag that we have associated it with. If we hold the mouse over the rectangle a tooltip displays the tag and its value.



```
BristolBabcock.BristolOPCServer\"rtrdb1"."nw3000realanalog"."name:char:CW1:@GV.TANK1_LEVEL"."value:float" = 41.1999435424805
```

That's all there is to it. The OPC server is now serving ObjectServer data to the GraphWorX OPC client and displaying it on the page.

TIP – Copy, Paste and Change the Tag

Once you have retrieved one OPC tag in this way, there is no need to go through this whole process again. You can copy, paste and change the original OPC tag into the Data Source field for any dynamic object to display any tag you like on the HMI display.

Using ObjectServer with Legacy GraphWorX™ HMIs

HMI systems created to view NW3000 and ControlWave controller data before the introduction of ObjectServer used the ICONICS OPC Server (also known as the OpenBSI OPC Server).

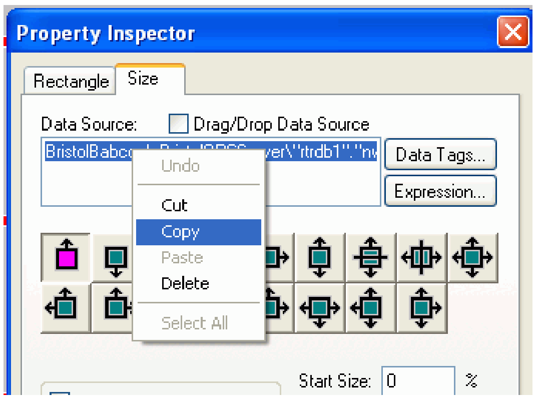
The ICONICS OPC server used a different tag format to the one ObjectServer uses, but the ObjectServer OPC Server can be set to read the old tag format simply by running the **'IconicsOPCServerReplace.cmd'** file found in the ObjectServer bin folder (the default location is 'C:\OpenBSI\ObjectServer\bin').

Now the ObjectServer OPC server can register and serve OpenBSI OPC tags. This action can be reversed at any time by running the 'IconicsOPCServerRestore.cmd' file from the same directory.

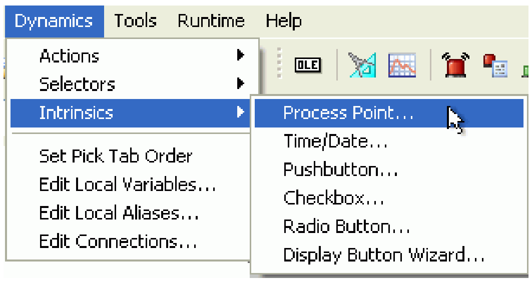
Modifying a Tag in GraphWorX

If we want to display the name of the signal above the Tank Level, using a Process Point, this is what we can do:-

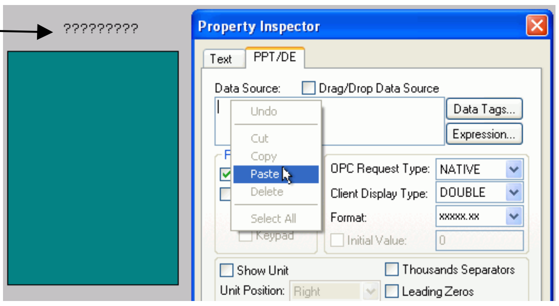
1. Copy the tag from the Size tab's Data Source field.



2. Create a Process Point by selecting the **Dynamics → Intrinsics → Process Point** option from the menu bar, as shown here. A Process Point displays a tag as text.

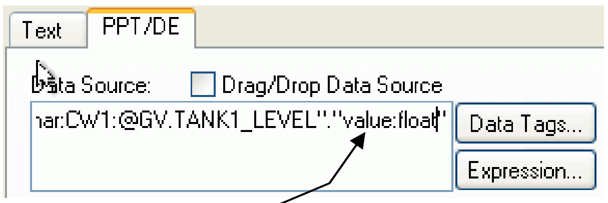


3. Click on the display area where you want the Process Point to appear and then paste the copied tag into the Process Point's Data Source field.



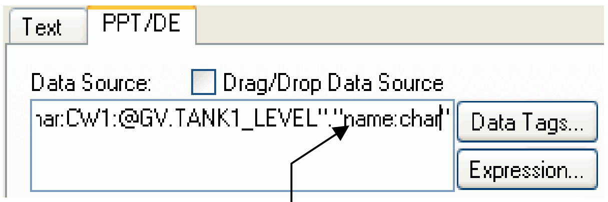
OPC Request Type: NATIVE
Client Display Type: DOUBLE
Format: xxxxx.xx

4. Change the object property of the tag to be the name. You have to specify the data type of the attribute, so change “value:float” to “name:char”. Retain the paired quotation marks! The name is a string or character data type (“char” for short).



```
nar:CW1:@GV.TANK1_LEVEL"."value:float"
```

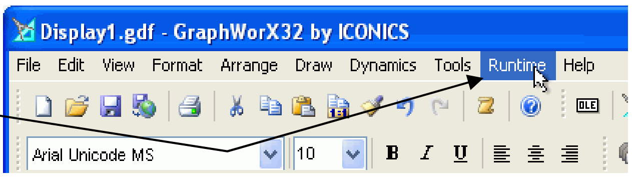
Delete the “value:float” property.



```
nar:CW1:@GV.TANK1_LEVEL"."name:char"
```

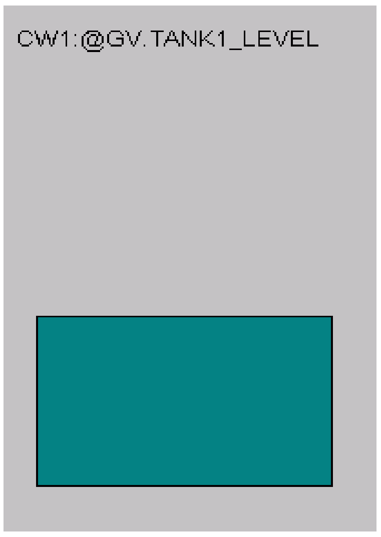
Type in the new “name:char” property. **Keep the double quotes**.

5. Close the Property Inspector dialog and put the display into Runtime mode.



6. The signal name is now displays above the rectangular level display.

   We have used a Process Point to display the tag as text. We have modified the text of the original “value” tag to display the “name” tag instead.



You should now be able to get ObjectServer data into the Genesis32 HMI application via the OPC Server.

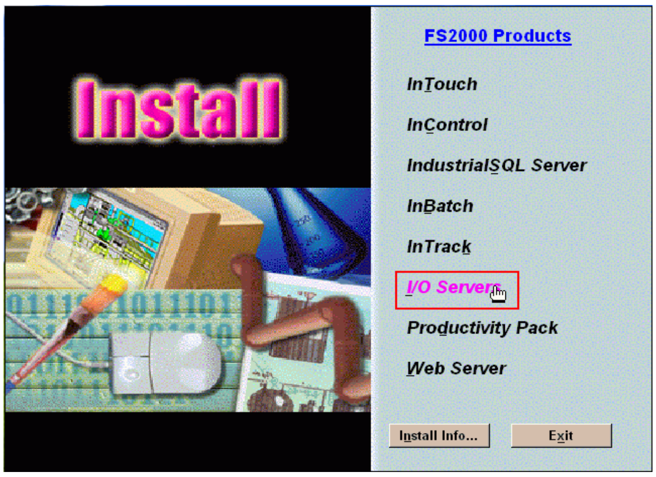
Getting ObjectServer Real-time Data into InTouch®

Sold by Wonderware®, InTouch is an HMI software package that includes an OPC client, so it can display data from OPC servers. This section shows how to get ObjectServer data into an InTouch window that is running on the same computer as ObjectServer.

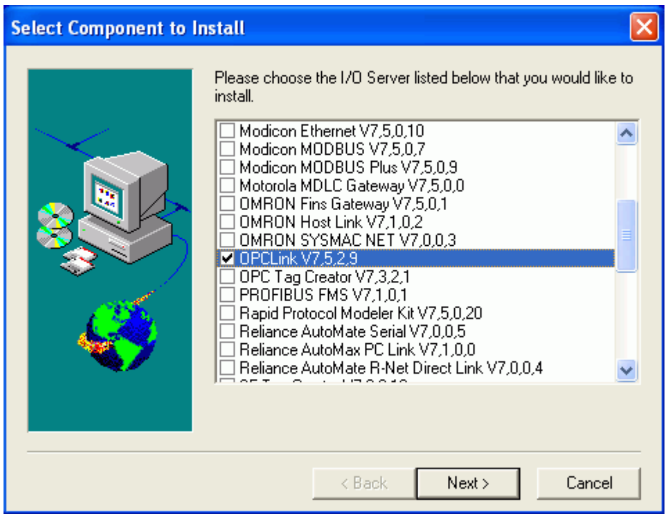
NOTE
What follows is not a detailed description of how to use every feature of InTouch. It is intended only as an example to show how to get tags from the ObjectServer OPC server into InTouch. Please use the online help provided with InTouch for information on any other feature.

1. You must first install the OPCLink I/O Server in order to view data from the ObjectServer OPC Server with InTouch.

   After installing InTouch, place the I/O Servers disk into the CD/DVD Drive and click the 'IO Servers option.



2. Select the OPCLink option from the list of I/O Servers. Then click **Next>** and finish the installation wizard.

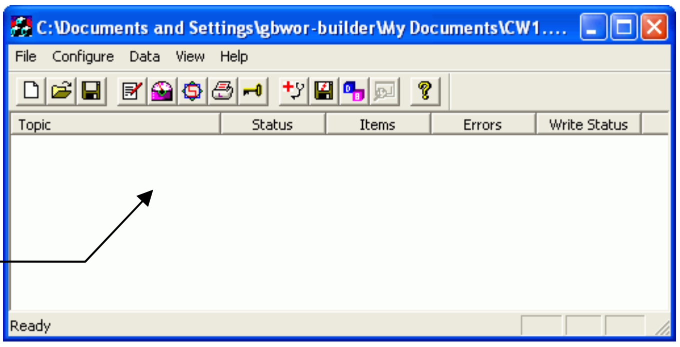

Select Component to Install

Please choose the I/O Server listed below that you would like to install.

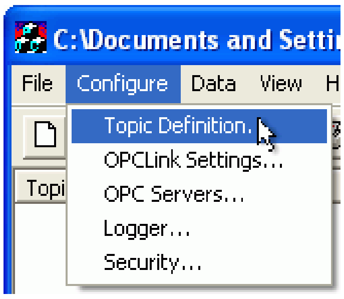
- [ ] Modicon Ethernet V7.5.0.10
- [ ] Modicon MODBUS V7.5.0.7
- [ ] Modicon MODBUS Plus V7.5.0.9
- [ ] Motorola MDLC Gateway V7.5.0.0
- [ ] OMRON Fins Gateway V7.5.0.1
- [ ] OMRON Host Link V7.1.0.2
- [ ] OMRON SYSMAC NET V7.0.0.3
- [✓] OPCLink V7.5.2.9
- [ ] OPC Tag Creator V7.3.2.1
- [ ] PROFIBUS FMS V7.1.0.1
- [ ] Rapid Protocol Modeler Kit V7.5.0.20
- [ ] Reliance AutoMate Serial V7.0.0.5
- [ ] Reliance AutoMax PC Link V7.1.0.0
- [ ] Reliance AutoMate R-Net Direct Link V7.0.0.4

We now need to configure OPCLink to connect to the ObjectServer OPC server. We can then set up an OPC Path for the OPC Server, so the client can request tags based on that path.

3. Start the OPC Link application. Click **Start → All Programs → Wonderware FactorySuite → IO Servers → OPCLink**. The user interface displays.

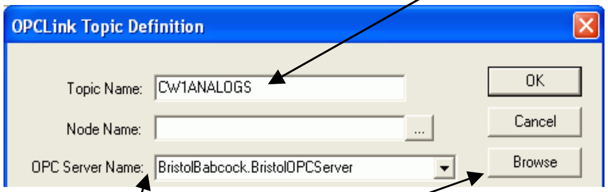

C:\Documents and Settings\gbwor-builder\My Documents\CW1....
File Configure Data View Help
Topic Status Items Errors Write Status
Ready

4. In OPCLink, a topic contains the necessary configuration for a group of tags used in an InTouch window (display). Select **Configure → Topic Definition**.



5. Click **New** on the Topic Definition dialog, and then configure a new Topic.

   Type a name for the topic here. The name can be any text you want, so be descriptive to remind yourself of what the topic does. In this example we want to be able find all analog tags from the CW1 device, so the topic name (CW1ANALOGS) should reflect this.

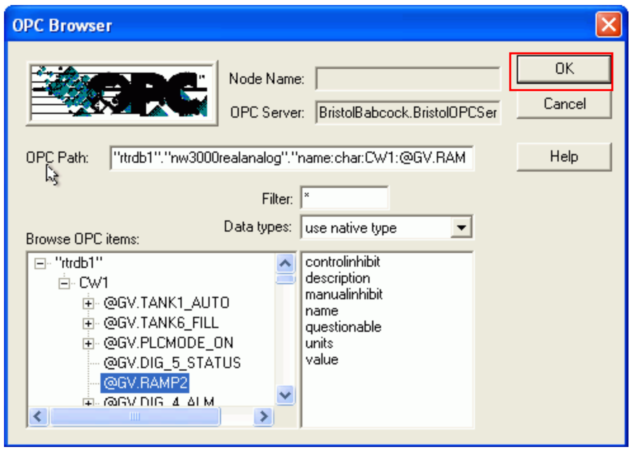


Topic Name: CW1ANALOGS
Node Name:
OPC Server Name: BristolBabcock.BristolOPCServer

Select the BristolBabcock.BristolOPCServer object from the list of OPC Servers.

Click **Browse** next to the OPC Server Name field to browse for tags using the selected OPC server.

6. From the OPC Browser interface, go through the tag hierarchy and select **any** realanalog signal. Then click **OK**.



OPC Browser

Node Name:

OPC Server: BristolBabcock.BristolOPCSer

OPC Path: "rtrdb1"."nw3000realanalog"."name:char:CW1:@GV.RAM

Filter: *

Data types: use native type

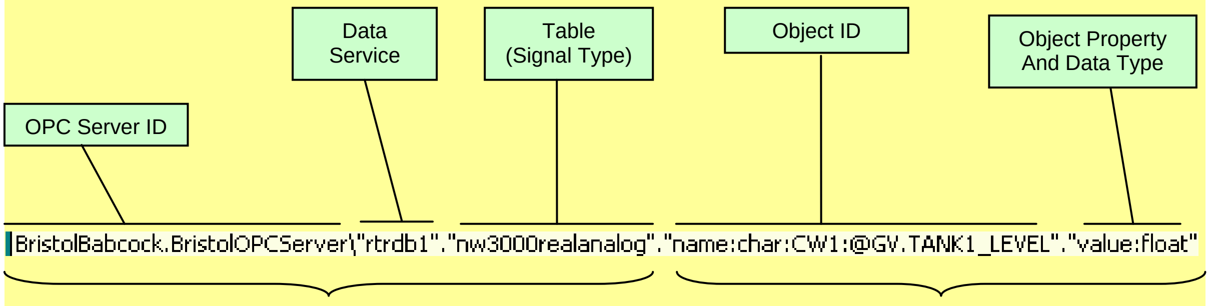
Browse OPC items:

- "rtrdb1"
  - CW1
    - @GV.TANK1_AUTO
    - @GV.TANK6_FILL
    - @GV.PLCMODE_ON
    - @GV.DIG_5_STATUS
    - @GV.RAMP2
    - @GV.DIG_4_ALM

controlinhibit
description
manualinhibit
name
questionable
units
value

Why Don’t You Select a Specific Tag?

We don’t select a specific tag because a Topic Definition is more useful if we can use it to select **multiple** tags. We are going to use the first part of the OPC tag string to define a path that exposes all the analog signals that are within a selected Bristol controller. Below is a fully qualified ObjectServer OPC tag.

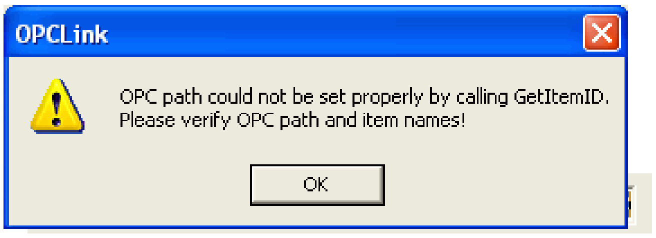


BristolBabcock.BristolOPCServer\"rtrdb1"."nw3000realanalog"."name:char:CW1:@GV.TANK1_LEVEL"."value:float"

We are going to use this part of the tag as an OPC path in InTouch.

Then we can define this part of the tag ourselves when creating InTouch windows (displays). We can use the OPC Browser interface to search for the exact tag we want, and then copy this last part of the tag into the Item field on the Tagname Dictionary dialog (see step 17 below).

7. If you see this warning dialog, just click **OK**. We will be adjusting the OPC path manually in the next step.



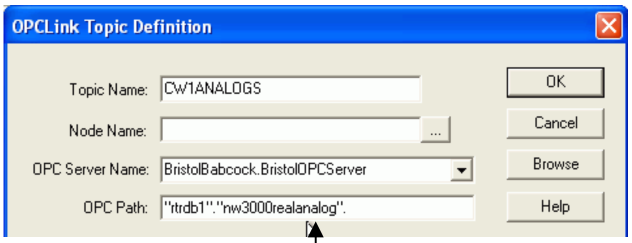
OPCLink

OPC path could not be set properly by calling GetItemID.
Please verify OPC path and item names!

8. Back on the OPC Link Topic Definition dialog, we need to modify the OPC Path. The modified path name ensures that we can specify any realanalog tag within the InTouch Tag Dictionary.

   When you have made this modification click **OK**.

Delete the part of the tag from after the second period to the end of the tag, as shown here. Do not include the second period in the selection



OPCLink Topic Definition

Topic Name: CW1ANALOGS

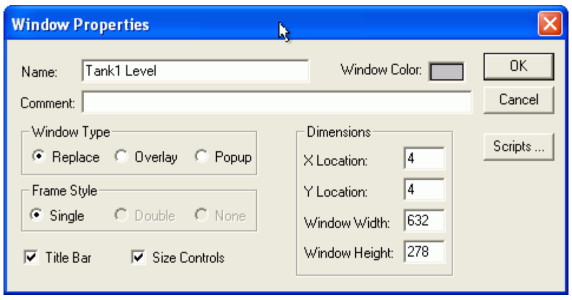
Node Name:

OPC Server Name: BristolBabcock.BristolOPCServer

OPC Path: "rtrdb1"."nw3000realanalog".

The OPC Path should now read *"rtrdb1"."nw3000realanalog".*

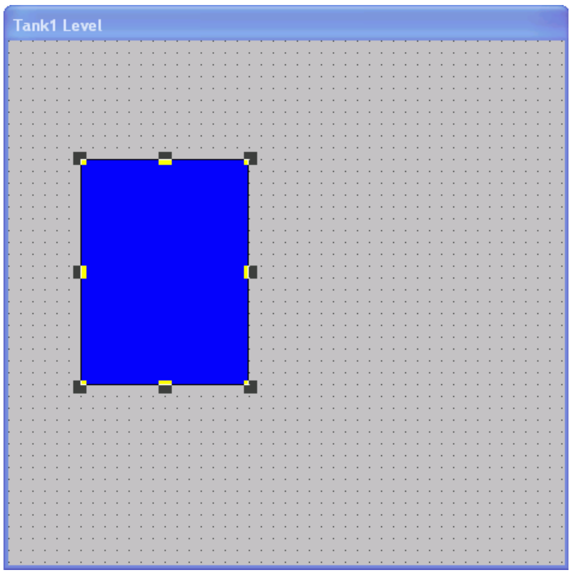
9. Now start InTouch WindowMaker by selecting **Start → All Programs → Wonderware FactorySuite → InTouch WindowMaker**. Then select **File → New Window…** to create a new window. In this example, we will name this window "Tank1 Level".


Window Properties

Name: Tank1 Level
Comment:
Window Type: [✓ Replace] Overlay Popup
Frame Style: [✓ Single] Double None
[✓ Title Bar] [✓ Size Controls]
Window Color:
Dimensions
X Location: 4
Y Location: 4
Window Width: 632
Window Height: 278

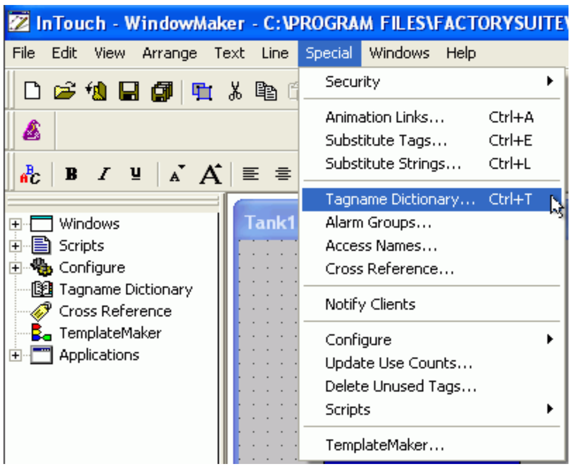
10. We have re-sized the window and created a rectangle with a blue background To represent the level in Tank1. We now need to animate this visual object and control its height by the value of the Tank1.Level tag in the ObjectServer OPC server.

    To do this, we need to know the ObjectServer OPC tag format and the names of the signals in the Bristol Controller load.

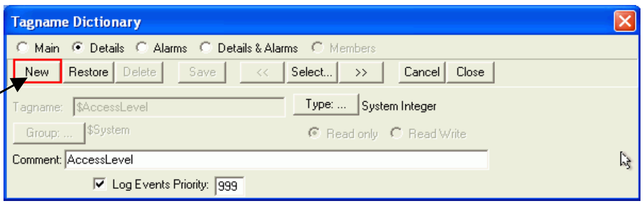


11. We now need to create a new tag in the InTouch 'Tag Dictionary' that links to the topic we created in OPCLink and uses its path to form the tag for the Tank1 Level value.

    Select **Special → Tagname Dictionary** from the InTouch WindowMaker menu bar.



12. Click **New** on the ‘Tagname Dictionary’ dialog to create a new tag definition.


Tagname Dictionary

( ) Main [✓ Details] ( ) Alarms ( ) Details & Alarms ( ) Members

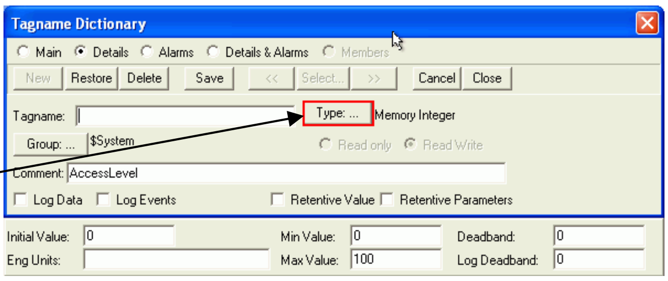
Tagname: $AccessLevel Type: ... System Integer

Group: ... $System [✓ Read only] ( ) Read Write

Comment: AccessLevel

☑ Log Events Priority: 999

13. Click **Type**.


Tagname Dictionary

( ) Main [✓ Details] ( ) Alarms ( ) Details & Alarms ( ) Members

Tagname: Type: ... Memory Integer

Group: ... $System ( ) Read only [✓ Read Write]

Comment: AccessLevel

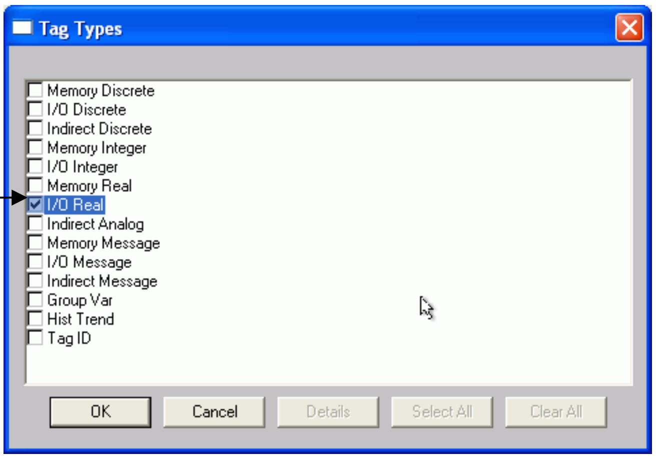
☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

Initial Value: 0 Min Value: 0 Deadband: 0

Eng Units: Max Value: 100 Log Deadband: 0

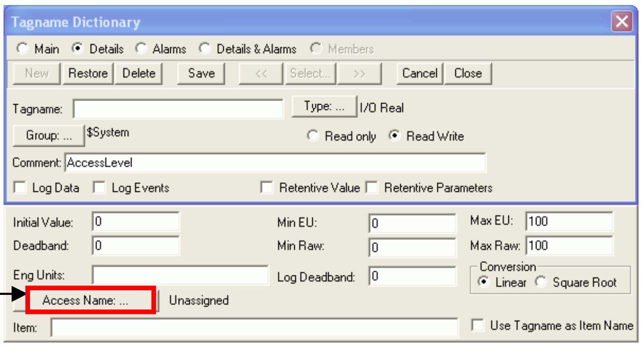
14. Select IO Real check box from the Tag Types dialog.

Click **OK**.


Tag Types

☐ Memory Discrete
☐ I/O Discrete
☐ Indirect Discrete
☐ Memory Integer
☐ I/O Integer
☐ Memory Real
☑ I/O Real
☐ Indirect Analog
☐ Memory Message
☐ I/O Message
☐ Indirect Message
☐ Group Var
☐ Hist Trend
☐ Tag ID

15. A new section appears at the bottom of the Tagname Dictionary dialog. We need to create an Access Name, so click **Access Name**.


Tagname Dictionary

( ) Main [✓ Details] ( ) Alarms ( ) Details & Alarms ( ) Members

Tagname: Type: ... I/O Real

Group: ... $System ( ) Read only [✓ Read Write]

Comment: AccessLevel

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

Initial Value: 0 Min EU: 0 Max EU: 100

Deadband: 0 Min Raw: 0 Max Raw: 100

Eng Units: Log Deadband: 0 Conversion [✓ Linear] ( ) Square Root

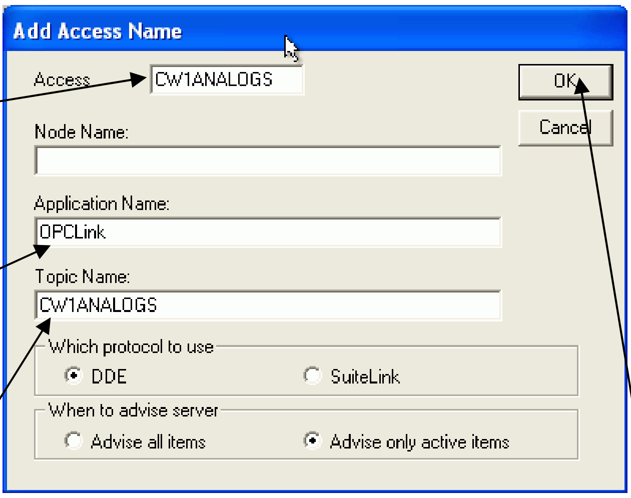
Access Name: ... Unassigned

Item: ☐ Use Tagname as Item Name

16. On the Add Access Name dialog, define a name for the Access.

Define the Application Name from which we can access our tag (OPCLink).

Define the topic which we created in OPCLink (CW1ANALOGS).


Add Access Name

Access: CW1ANALOGS

Node Name:

Application Name: OPCLink

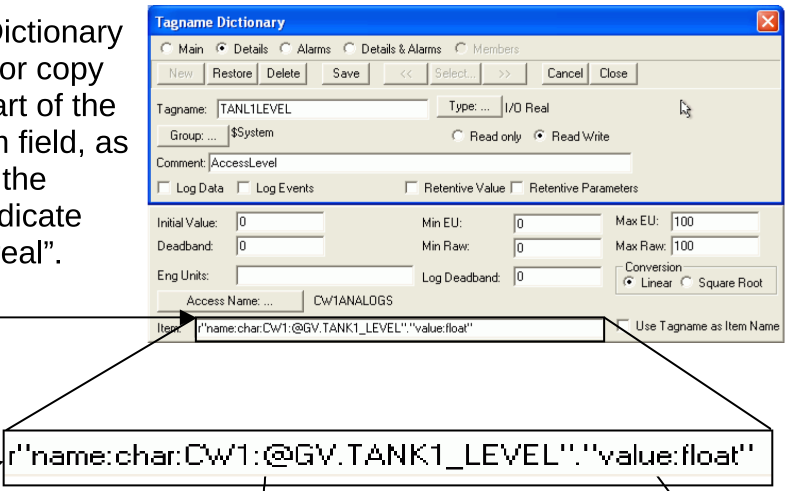
Topic Name: CW1ANALOGS

Which protocol to use: [✓ DDE] SuiteLink

When to advise server: Advise all items [✓ Advise only active items]

When you have finished filling in these fields, click **OK**.

17. When the Tagname Dictionary dialog displays, type (or copy and paste) the final part of the tag name into the Item field, as shown here. Precede the partial tag with **r**, to indicate that the data type is ”real”.


Tagname Dictionary

Main [✓ Details] Alarms Details & Alarms Members

Tagname: TANL1LEVEL

Type: I/O Real

Group: $System

Read only [✓ Read Write]

Comment: AccessLevel

Log Data Log Events Retentive Value Retentive Parameters

Initial Value: 0

Deadband: 0

Eng Units:

Min EU: 0

Min Raw: 0

Log Deadband: 0

Max EU: 100

Max Raw: 100

Conversion: [✓ Linear] Square Root

Access Name: CW1ANALOGS

Item: r"name:char:CW1:@GV.TANK1_LEVEL"."value:float"

Use Tagname as Item Name

r"name:char:CW1:@GV.TANK1_LEVEL"."value:float"

The **r** signifies a real value.

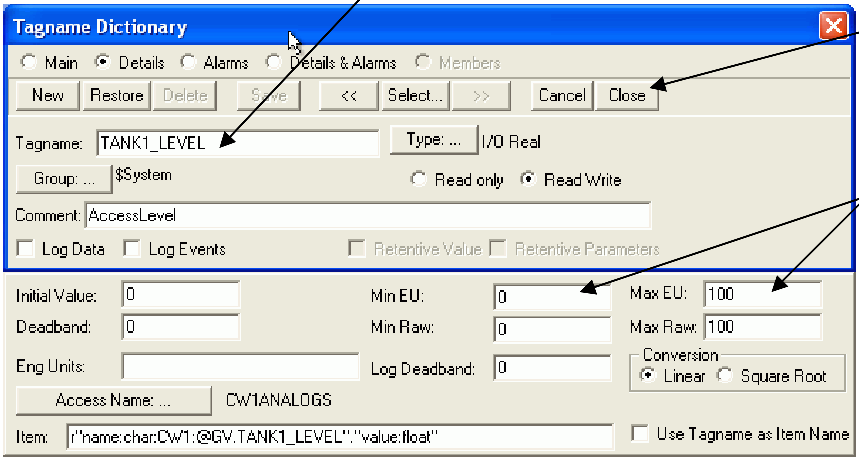
This part of the tag identifies the object.

This is part of the tag specifies the object property that we want to display.

In the Tagname field we have entered the tag name as **TANK1_LEVEL** (spaces are not allowed in tag names).

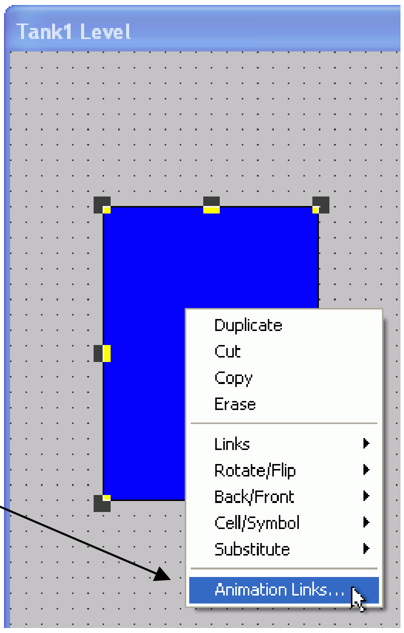
When everything has been configured, click **Close**. This saves your configuration and returns you to the main window.

Set the Min and Max fields before clicking **Close**.

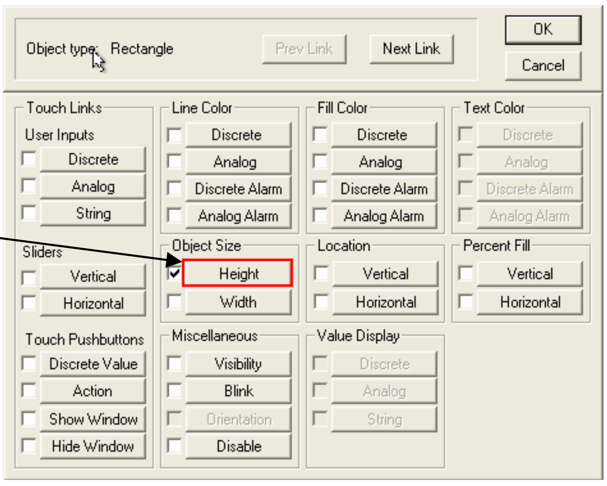


Tagname:	TANK1_LEVEL	I/O Real			
Group:	$System	Read only	[✓ Read Write]		
Comment:	AccessLevel				
Log Data	Log Events	Retentive Value	Retentive Parameters		
Initial Value:	0	Min EU:	0	Max EU:	100
Deadband:	0	Min Raw:	0	Max Raw:	100
Eng Units:		Log Deadband:	0	Conversion:	[✓ Linear] Square Root
Access Name:	CW1ANALOGS				
Item:	r"name:char:CW1:@GV.TANK1_LEVEL"."value:float"	Use Tagname as Item Name			

18. Now we animate our graphic by associating it with the new OPC tag we have created (that the ObjectServer OPC server updates). Right-click on the rectangle and select **Animation Links** from the shortcut menu.

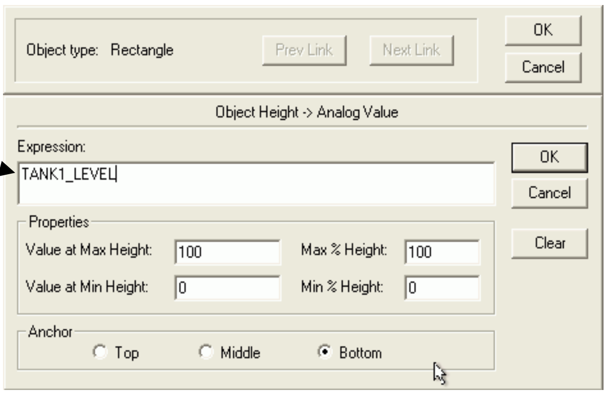


19. Select the check box next to the Height button under the ‘Object Size’ section of the dialog. Then click **Height**.



20. Type the name of the tag that you have just created in the Expression box for the object animation.

Then click OK on this and the previous dialog. This should return you to the display window.


Object type: Rectangle

Object Height -> Analog Value

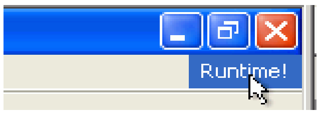
Expression: TANK1_LEVEL

Properties

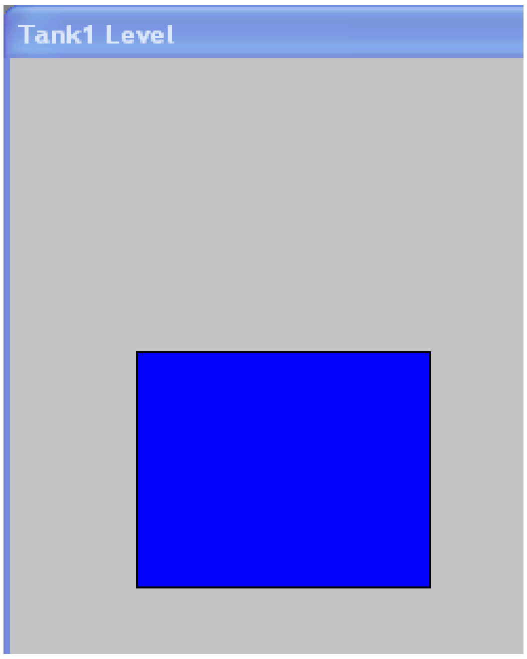
Value at Max Height:	100	Max % Height:	100
Value at Min Height:	0	Min % Height:	0

Anchor: Top Middle [✓ Bottom]

6. From the upper right-hand corner of the display window, click **Runtime!** This places the display window into Runtime mode.



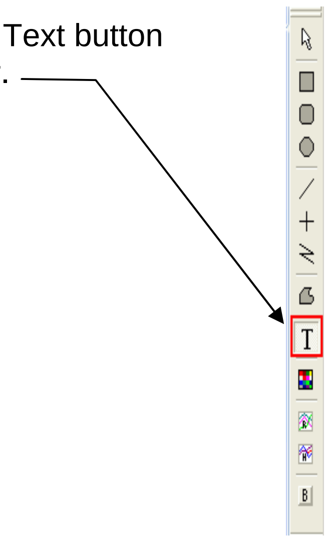
22. In Runtime mode, the rectangle’s height should update according to the corresponding value of the TANK1_LEVEL tag.



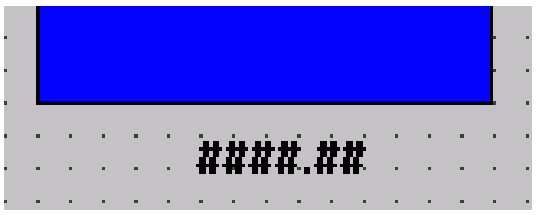
Using the Text Tool in InTouch to Show ObjectServer Tags

To show the signal's value as text (rather than a size animation), we need to create a new text object with wildcard symbols and then associate this object with a Display animation.

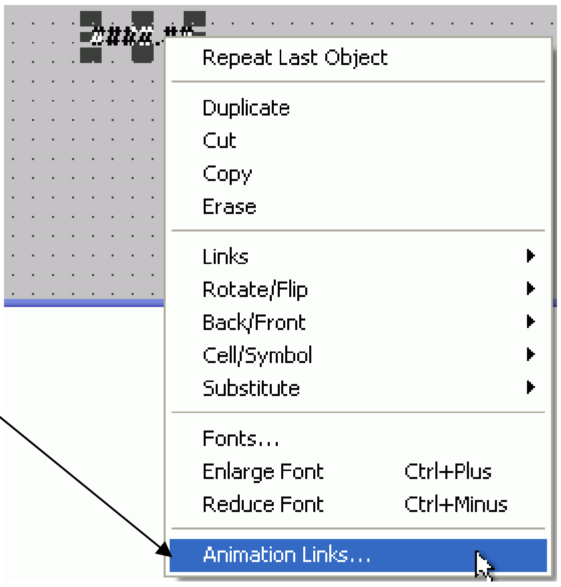
1. First, select the Text button from the toolbar.



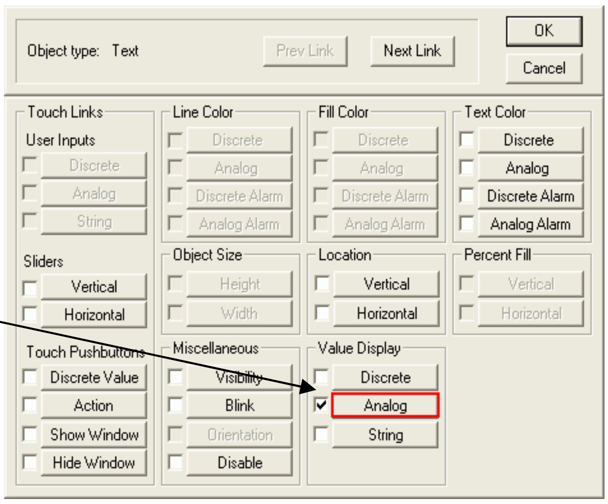
2. Then type ####.## onto the display area as shown here.



3. Right-click on the text object and select **Animation Links** from the shortcut menu.

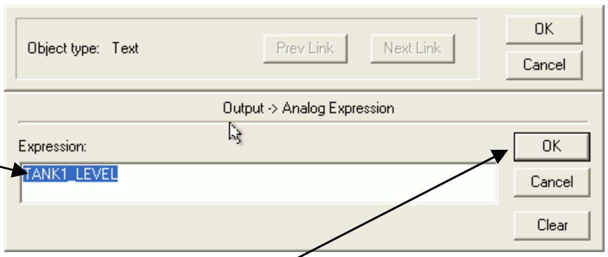


4. Select the check box to the left side of the Analog button under the Value Display section of the object animation dialog. Then click **Analog** button.



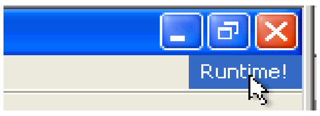
5. Type the name of the tag you created into the Expression box (in our example the tag name was **TANK1_LEVEL**).

   Then click **OK** on this and the previous dialog to return to the display window.

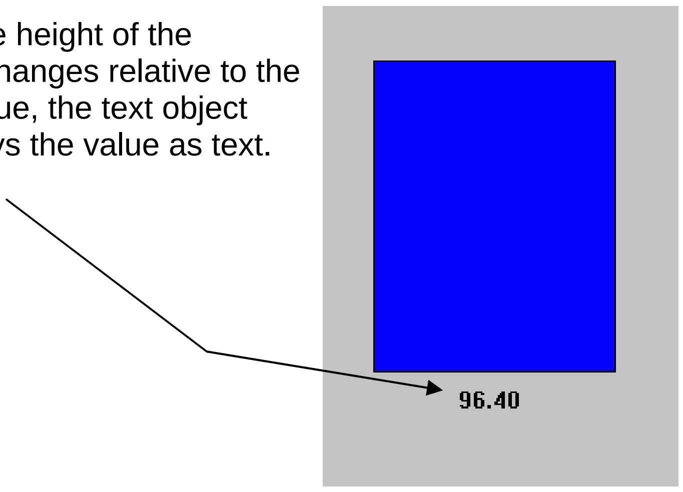


Expression: TANK1_LEVEL

6. From the upper right hand corner of the display window, click **Runtime!** again.



7. Now, as the height of the rectangle changes relative to the signal's value, the text object also displays the value as text.



Getting ObjectServer Real-time Data into iFix™

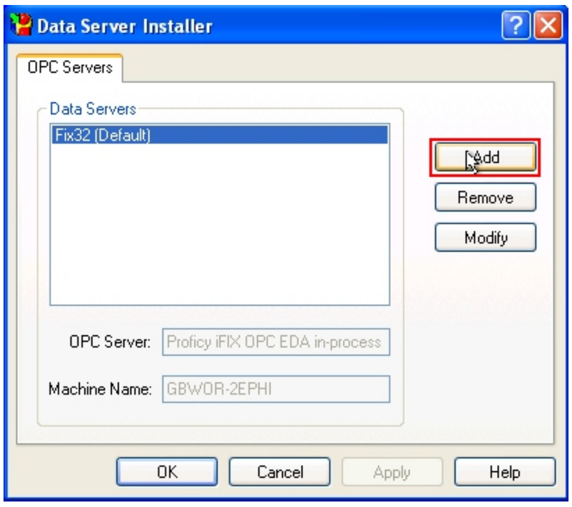
This is an HMI OPC client application formerly developed by Intellution, but now owned by GE Fanuc. This section describes how to get ObjectServer data into an iFix window that is running on the same computer as ObjectServer.

NOTE

What follows is not a detailed description of how to use every feature of iFix. It is intended only as an example to show how to get tags from the ObjectServer OPC Server into iFix. Please use the online help provided with iFix for information on any other feature.

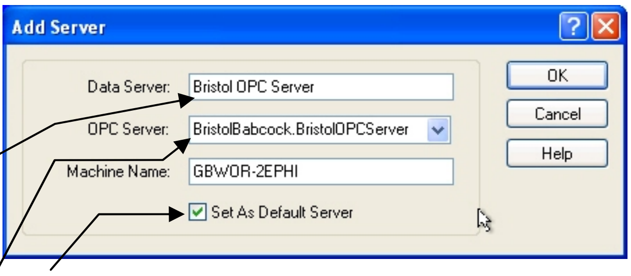
1. Set up the ObjectServer OPC server as the default data server for iFix. Click **Start → Programs → Proficy HMI SCADA iFix → Tools → Data Server Installer**.

2. From the Data Server Installer dialog click **Add**.



OPC Server: Proficy iFIX OPC EDA in-process

Machine Name: GBWOR-2EPHI



Data Server: Bristol OPC Server

OPC Server: BristolBabcock.BristolOPCServer

Machine Name: GBWOR-2EPHI

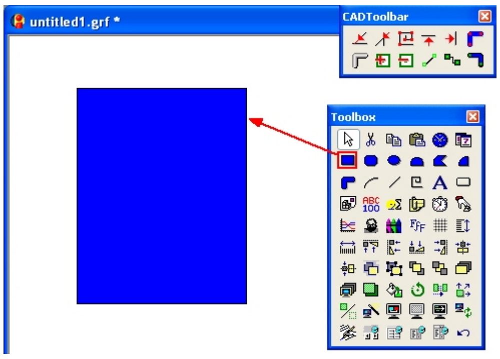
☑ Set As Default Server

2. On the Add Server dialog type the name of the data server as you want it to appear in iFix into the Data Server field.

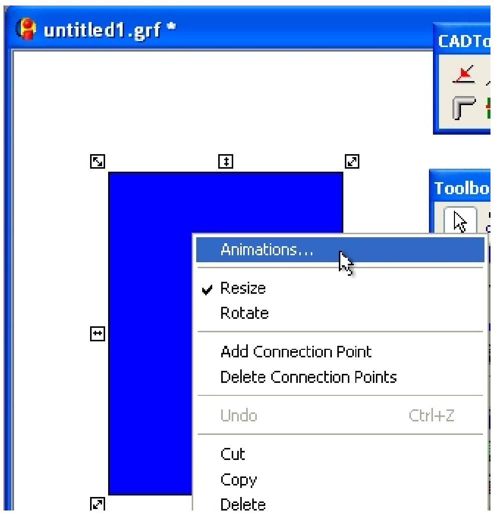
   Then select the ObjectServer OPC Server from the OPC Server's drop-down list of available OPC servers.

Finally, select the **Set As Default Server** check box. Then click **OK** to close the dialog/

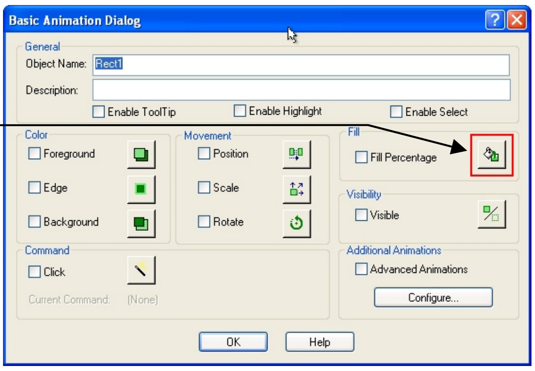
3. Next, in the Proficy iFix workspace click the rectangle object on the Toolbox and draw a rectangle on the drawing area of the untitled picture as shown.



4. Right-click on the rectangle and select the **Animations** option.



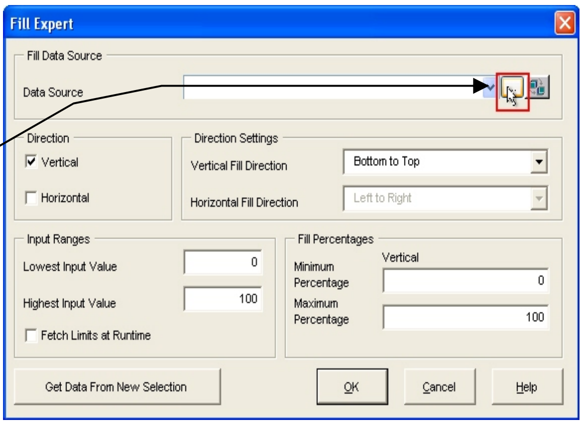
5. Next, click the button to the right of the Fill Percentage animation option, highlighted in this image.


Basic Animation Dialog

General
Object Name: Rect1
Description:
☐ Enable ToolTip ☐ Enable Highlight ☐ Enable Select

Color: ☐ Foreground ☐ Edge ☐ Background
Movement: ☐ Position ☐ Scale ☐ Rotate
Fill: ☐ Fill Percentage
Visibility: ☐ Visible
Command: ☐ Click
Current Command: (None)
Additional Animations: ☐ Advanced Animations

6. Select the button marked with an ellipsis (…) to the right of the Data Source field to browse for the ObjectServer OPC server data source.


Fill Expert

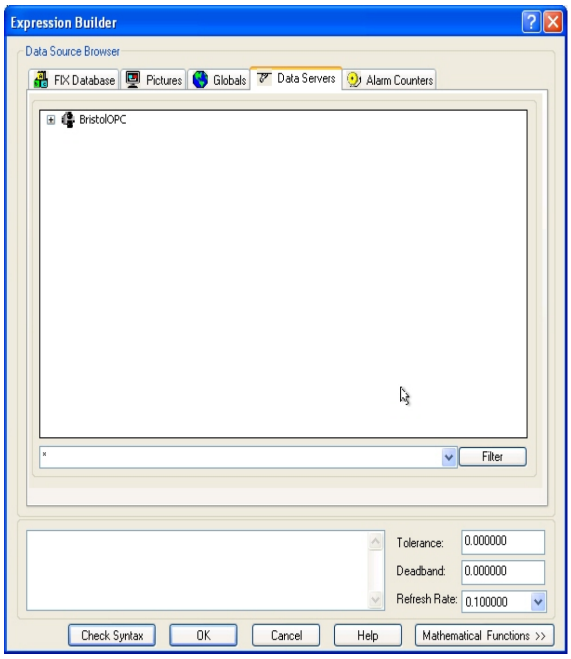
Fill Data Source
Data Source:

Direction: ☑ Vertical ☐ Horizontal
Direction Settings
Vertical Fill Direction: Bottom to Top
Horizontal Fill Direction: Left to Right

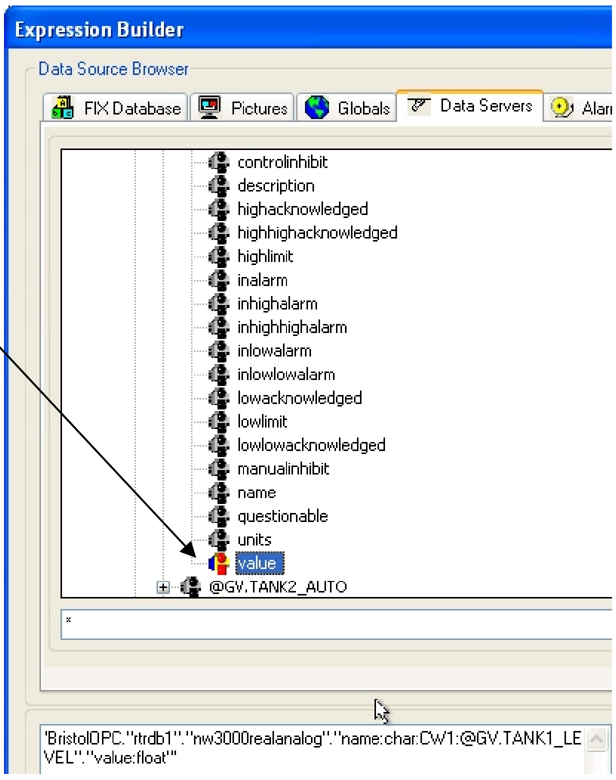
Input Ranges
Lowest Input Value: 0
Highest Input Value: 100
☐ Fetch Limits at Runtime

Fill Percentages (Vertical)
Minimum Percentage: 0
Maximum Percentage: 100

7. Select the Data Servers tab from the Expression Builder dialog. The BristolOPC server should display because we configured it in steps 1 and 2.

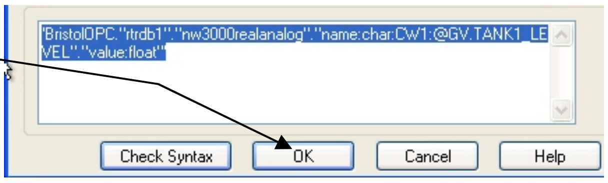


8. Now select the tag that you want. In this example, we have chosen the Value property of the TANK1_LEVEL signal in the CW! device.



'BristolOPC.''rtrdb1''.''nw3000realanalog''.''name:char:CW1:@GV.TANK1_LEVEL''.''value:float'''

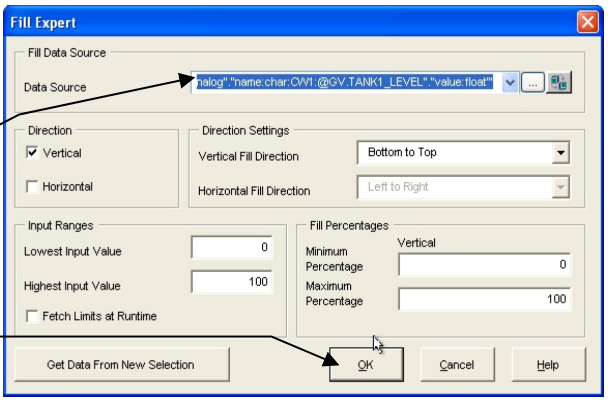
Click **OK** to return to the Fill Expert dialog.



'BristolOPC.''rtrdb1''.''nw3000realanalog''.''name:char:CW1:@GV.TANK1_LEVEL''.''value:float'''

9. The OPC tag that you selected completes the Data Source field on the Fill Expert dialog.

   Click **OK** to return to the Basic Animation dialog.



Fill Data Source
Data Source: nalog","name:char:CW1:@GV.TANK1_LEVEL","value:float""

Direction
Vertical [✓]
Horizontal [ ]

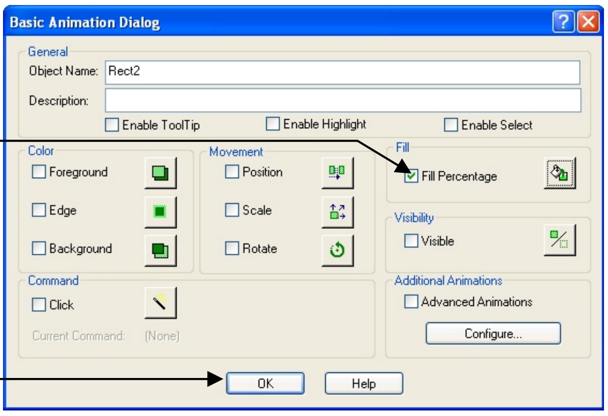
Direction Settings
Vertical Fill Direction: Bottom to Top
Horizontal Fill Direction: Left to Right

Input Ranges
Lowest Input Value: 0
Highest Input Value: 100
Fetch Limits at Runtime [ ]

Fill Percentages
Vertical
Minimum Percentage: 0
Maximum Percentage: 100

On the Basic Animation dialog, note that the Fill Percentage check box has been selected..

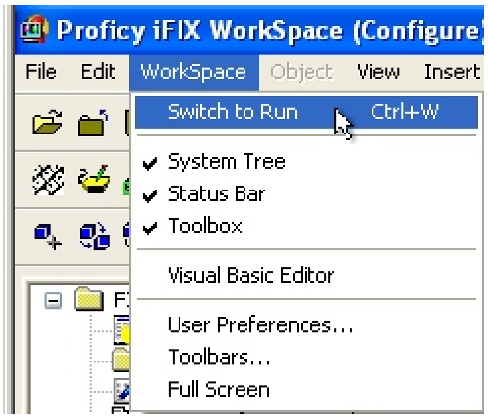
Click **OK** to return to the picture window.



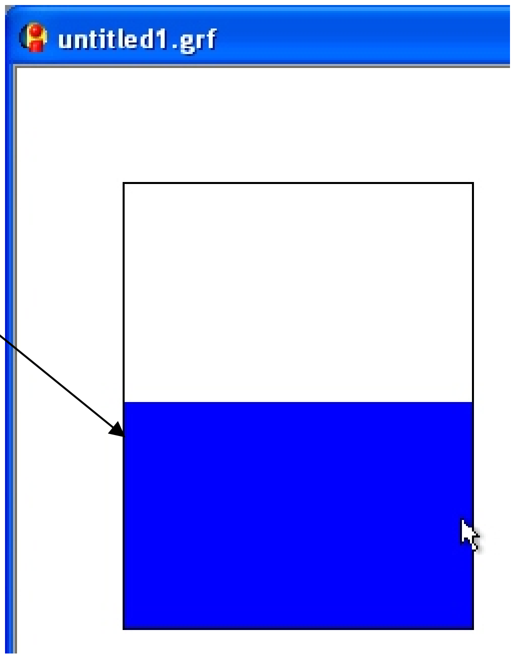
General
Object Name: Rect2
Description:
Enable ToolTip [ ] Enable Highlight [ ] Enable Select [ ]

Color: Foreground [ ] Edge [ ] Background [ ]
Movement: Position [ ] Scale [ ] Rotate [ ]
Fill: Fill Percentage [✓]
Visibility: Visible [ ]
Command: Click [ ]
Current Command: (None)
Additional Animations: Advanced Animations [ ]

10. Switch the workspace into Run mode by selecting the **Workspace → Switch to Run** option from the menu bar, or by pressing the **Ctrl** and **W** keys simultaneous.



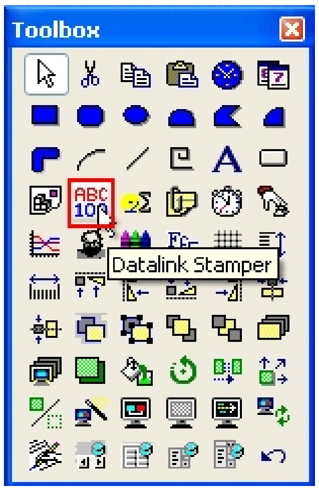
11. In Run mode, the rectangle adjusts its size to reflect the value of the selected OPC tag.



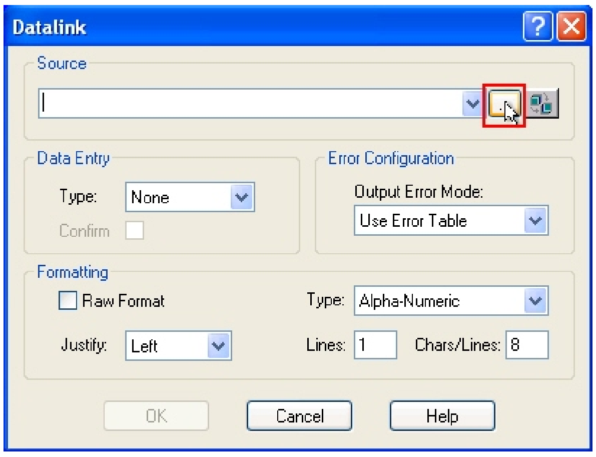
Using the iFix Datalink Stamper to Display ObjectServer Tags

Use this option to display tags as text in iFix by adding a 'Datalink Stamper' object to our picture. This displays the signal's name above the rectangular object.
.

1. Select the Datalink Stamper tool (highlighted in red in the image on the right) from the iFix Toolbox. The Datalink dialog displays,.



2. On the Datalink dialog, select the ellipsis (…) button to the right of the Source field. The Expression Builder dialog displays.


Datalink

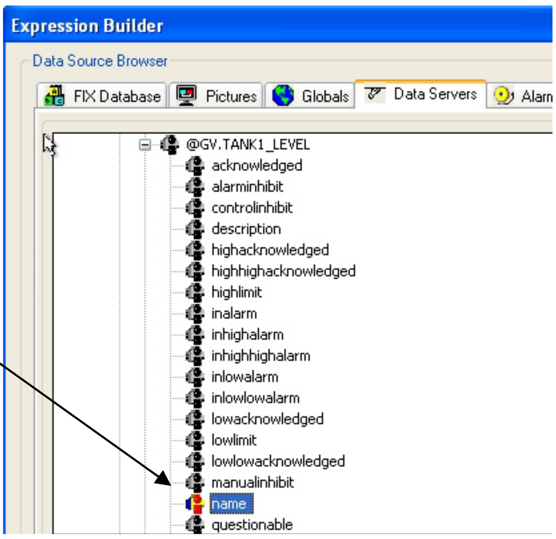
Source

Data Entry
Type: None
Confirm

Error Configuration
Output Error Mode: Use Error Table

Formatting
Raw Format
Justify: Left
Type: Alpha-Numeric
Lines: 1
Chars/Lines: 8

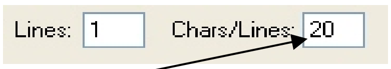
3. Select the Data Servers tab and then select the name tag from the list of tags available for the @GV.TANK1_LEVEL signal. Then close this dialog by clicking **OK**. The Datalink dialog displays,.


Expression Builder

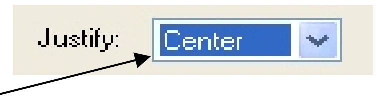
Data Source Browser

@GV.TANK1_LEVEL
acknowledged
alarminhibit
controlinhibit
description
highacknowledged
highhighacknowledged
highlimit
inalarm
inhighalarm
inhighhighalarm
inlowalarm
inlowlowalarm
lowacknowledged
lowlimit
lowlowacknowledged
manualinhibit
name
questionable

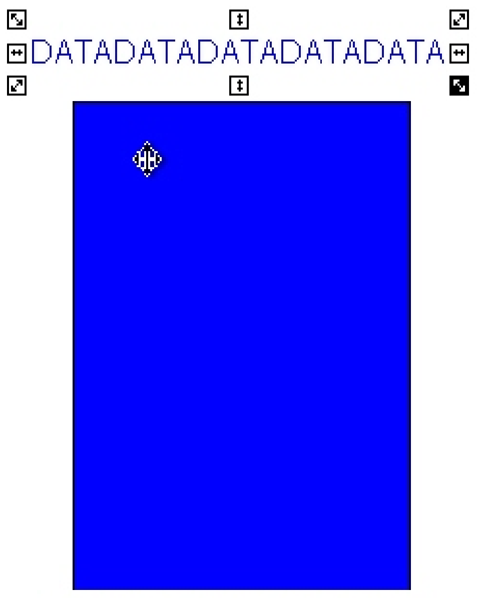
4. Back on the Datalink dialog, set the Chars field to an appropriate number.


Lines: 1 Chars/Lines: 20

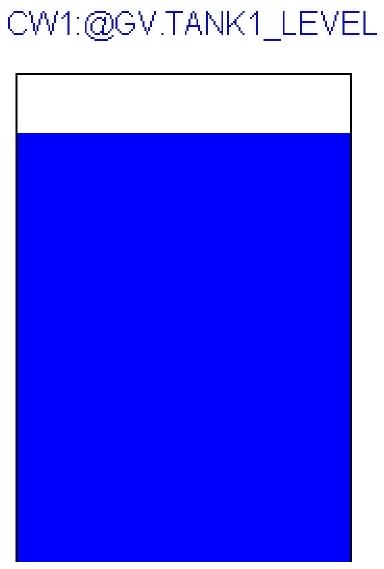
5. Set the Justify option to Center.


Justify: Center

6. Position the Datalink object on the picture, and place the picture into Run mode as described previously.



7. Note that the name of the signal displays above the rectangle.



Getting Real-time Data into OPC Clients from a Remote Workstation

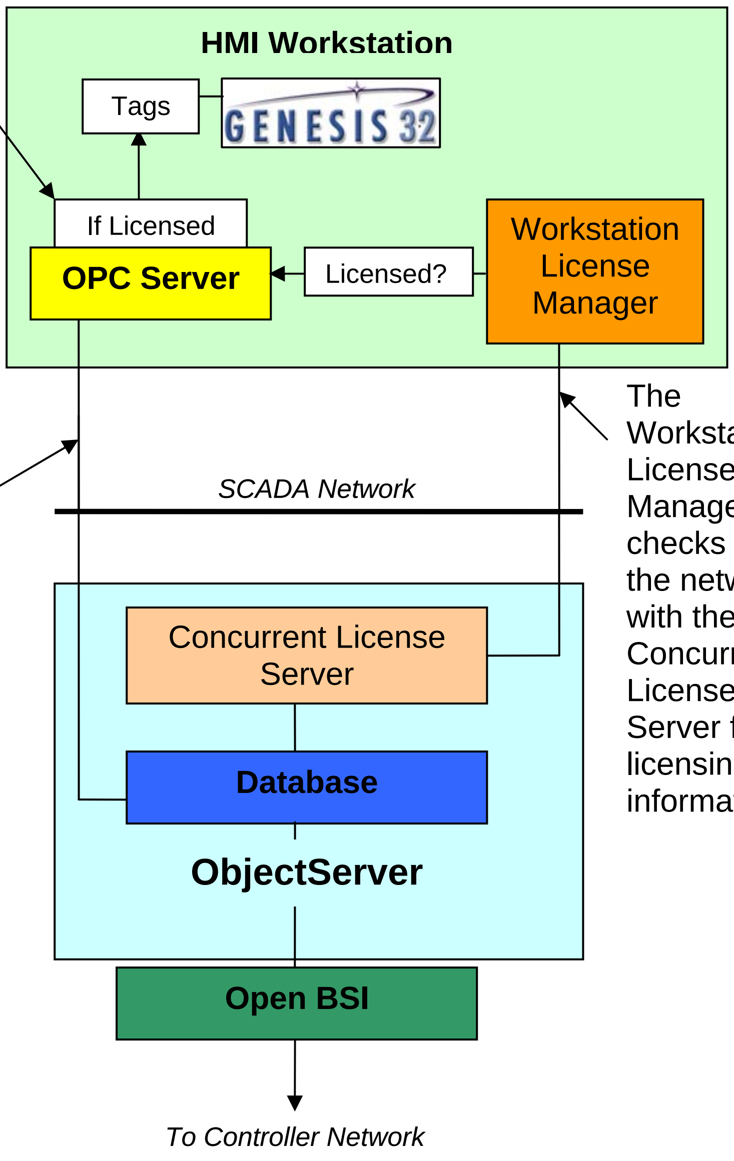
The previous examples show how to get ObjectServer real-time data into an OPC client installed on the **same** computer as ObjectServer. But what if you want to get OPC data to OPC clients installed on a **remote** workstation? For this you need to install just the ObjectServer clients (that is, the OPC servers) on the workstation.

ObjectServer and Licensing for Remote Workstation Clients

As described in *Getting ObjectServer Real-time Data into OPC Client”* earlier in this chapter, ObjectServer consists of a number of applications running alongside each other to provide data to third party OPC clients. We have seen that as soon as ObjectServer is installed, the OPC servers serve this data to OPC clients running on the **same** computer. However, the OPC server can also serve process data to OPC clients installed on remote workstations that are licensed with ObjectServer as concurrent users.

The OPC server will allow you to browse for tags from within the OPC client, but will not update the value of tags in Runtime mode unless the ObjectServer PC is licensed for the **concurrent** workstation connection.

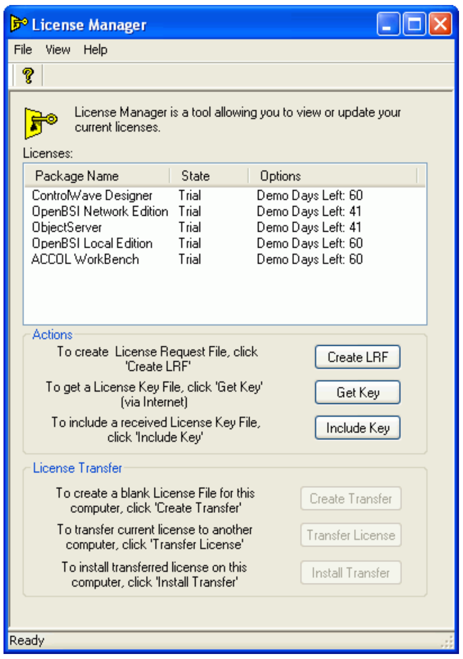
The OPC server on the workstation connects to the ObjectServer database over the network using a TCP/IP socket connection. This is faster and more efficient than COM and requires no configuration other than providing the ObjectServer computer name during the workstation installation.



The Workstation License Manager checks over the network with the Concurrent License Server for licensing information

1. Ensure that ObjectServer on your server machine is licensed for more than one client. If you are running the demo version, you should have a fully functional trial license for a 60-day period.

2. Select **Start → All Programs → Bristol Babcock Licensing → License Manager**. The Licence Manager displays. Check that the trial license is still operative. If not you will have to create an LRF file to obtain a license from us. See the License Manager online help file for instructions on how to do this.


License Manager

File View Help

License Manager is a tool allowing you to view or update your current licenses.

Licenses:

Package Name	State	Options
ControlWave Designer	Trial	Demo Days Left: 60
OpenBSI Network Edition	Trial	Demo Days Left: 41
ObjectServer	Trial	Demo Days Left: 41
OpenBSI Local Edition	Trial	Demo Days Left: 60
ACCOL WorkBench	Trial	Demo Days Left: 60

Actions

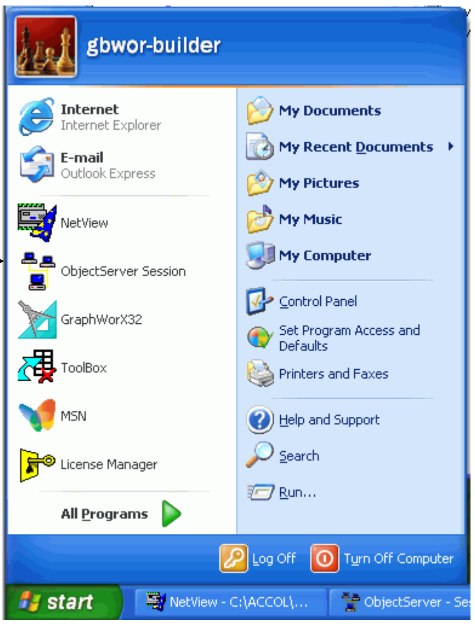
To create License Request File, click 'Create LRF'	Create LRF
To get a License Key File, click 'Get Key' (via Internet)	Get Key
To include a received License Key File, click 'Include Key'	Include Key

License Transfer

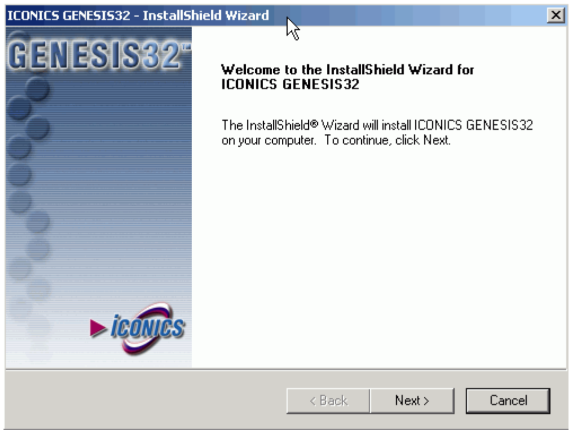
To create a blank License File for this computer, click 'Create Transfer'	Create Transfer
To transfer current license to another computer, click 'Transfer License'	Transfer License
To install transferred license on this computer, click 'Install Transfer'	Install Transfer

Ready

3. Start ObjectServer (if it is not already running) by clicking on the ObjectServer Session item from the Start menu.



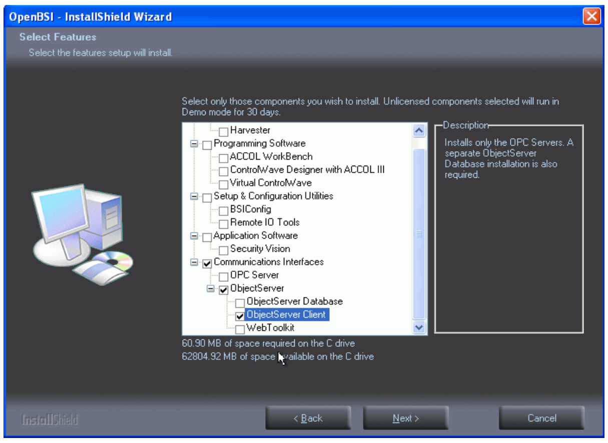
4. Install your OPC client on the remote workstation. For this example, we use Genesis32.


ICONICS GENESIS32 - InstallShield Wizard

Welcome to the InstallShield Wizard for ICONICS GENESIS32

The InstallShield® Wizard will install ICONICS GENESIS32 on your computer. To continue, click Next.

5. Install the ObjectServer client (which means the ObjectServer OPC servers **and** the Workstation License Manager component) onto your remote workstation. When you get to the Select Features page of the installation wizard, de-select everything except for ObjectServer client.


OpenBSI - InstallShield Wizard

Select Features

Select the features setup will install.

Select only those components you wish to install. Unlicensed components selected will run in Demo mode for 30 days.

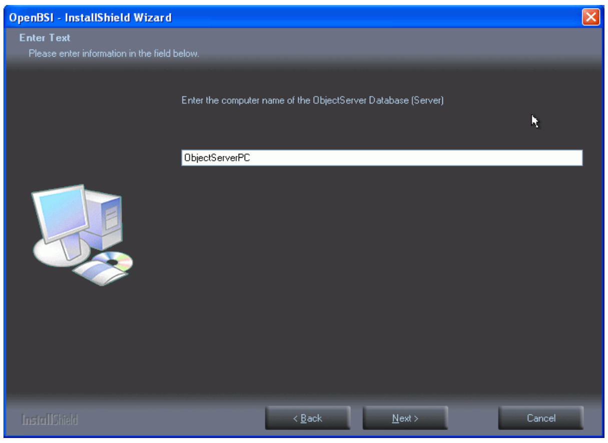
- ☐ Harvester
- ☐ Programming Software
  - ☐ ACCOL WorkBench
  - ☐ ControlWave Designer with ACCOL III
  - ☐ Virtual ControlWave
- ☐ Setup & Configuration Utilities
  - ☐ BSIConfig
  - ☐ Remote IO Tools
- ☐ Application Software
  - ☐ Security Vision
- ☑ Communications Interfaces
  - ☐ OPC Server
  - ☑ ObjectServer
    - ☐ ObjectServer Database
    - ☑ ObjectServer Client
    - ☐ WebToolkit

Description: Installs only the OPC Servers. A separate ObjectServer Database installation is also required.

60.90 MB of space required on the C drive
62804.92 MB of space available on the C drive

6. On the next page of the wizard, type the real name of the ObjectServer computer. **You cannot use an alias from the HOSTS file here. You must use the actual name of the ObjectServer computer.**

   When the installation finishes, you are ready to create your first display on the remote workstation.


OpenBSI - InstallShield Wizard

Enter Text

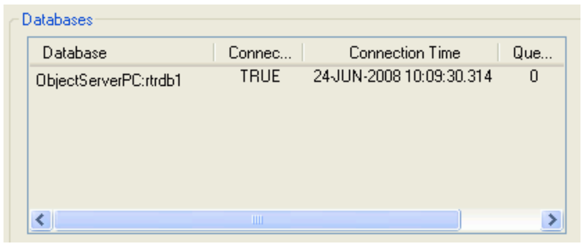
Please enter information in the field below.

Enter the computer name of the ObjectServer Database (Server)

ObjectServerPC

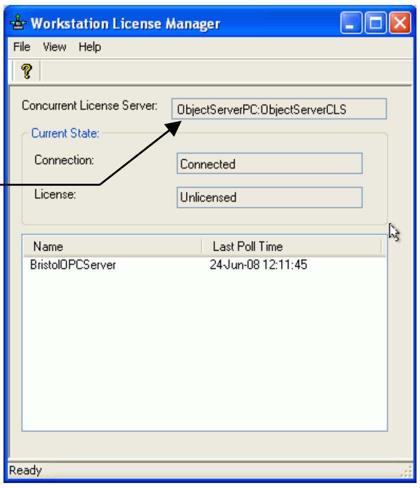
7. Create a display in your OPC client application on your workstation containing some process data. (This process has already been explained in the previous section *Getting ObjectServer Real-time Data into Genesis32™*. Please review that section for instructions.)

8. Check that your workstation is working correctly. Double-click the OPC server icon in the Windows system bar to open the OPC server's user interface. You should see that the remote OPC server has connected to the specified ObjectServer database.


Databases

Database	Connec...	Connection Time	Que...
ObjectServerPC:rtrdb1	TRUE	24-JUN-2008 10:09:30.314	0

9. Double click the Workstation License Manager icon from the Windows system bar. You should also see that the WLM on the remote workstation has also connected to the Concurrent License Server service on the ObjectServer PC.



Concurrent License Server: ObjectServerPC:ObjectServerCLS

Current State:

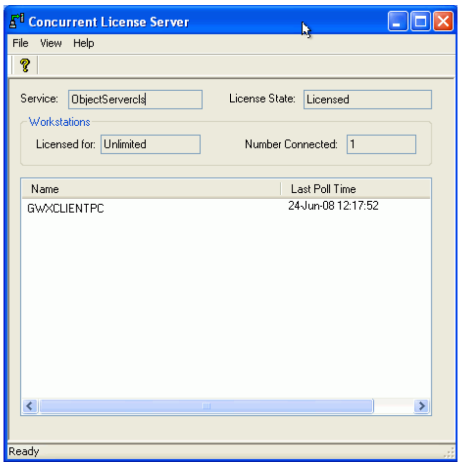
Connection: Connected

License: Unlicensed

Name	Last Poll Time
BristolOPCServer	24-Jun-08 12:11:45

Ready

10. Double-click the Concurrent License Server icon in the Windows system bar on the ObjectServer PC to bring up the user interface. You should see that the Concurrent License Server has registered the workstation connection.



Service: ObjectServercls License State: Licensed

Workstations

Licensed for: Unlimited Number Connected: 1

Name	Last Poll Time
GWXCLIENTPC	24-Jun-08 12:17:52

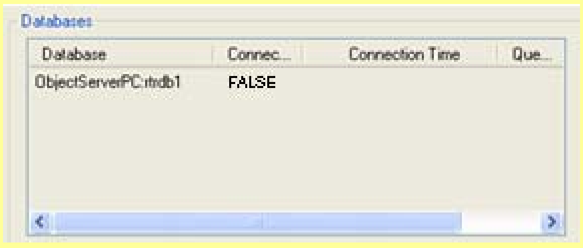
Ready

Troubleshooting

This section covers some problems you may experience after installing ObjectServer as a client only.

My OPC Server doesn't connect to the ObjectServer database!

When you double-click on the OPC Server icon - on the Windows system bar, you see **FALSE** in the Connected field:



Databases

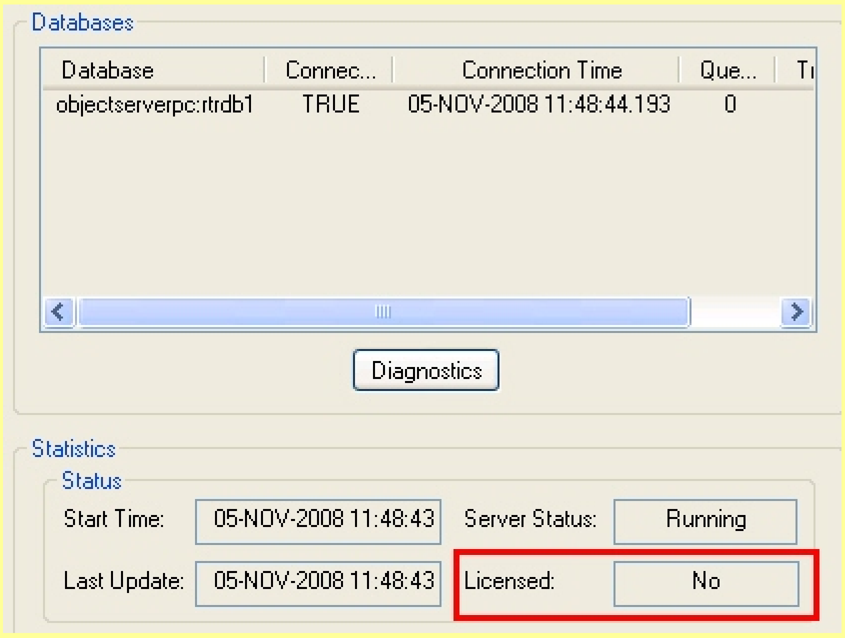
Database	Connec...	Connection Time	Que...
ObjectServerPC:rtrdb1	FALSE		

Check the following:-

- Is ObjectServer running properly on your server machine? If not, start ObjectServer and make sure it is running properly.
- Can you "ping" the Server machine from your workstation (using the Server's actual name)? If not, do the following:
  - Check your network cables.
  - Open a command window on the server and workstation and use the **ipconfig** command to check that the IP addresses and masks given to your server and workstation enable them to make a connection.
- Is there a firewall blocking port rtrdb1 (port 11101 by default)? If so, allow this port to work through the firewall.
- Is either your server or workstation a Windows Virtual Machine? The Workstation License Manager cannot make a connection between a virtual ObjectServer workstation and a remote server.

My OPC server connects to the ObjectServer database, but it is not licensed!

You find that the OPC server says it is connected to the ObjectServer database, but the interface tells you it is not licensed…



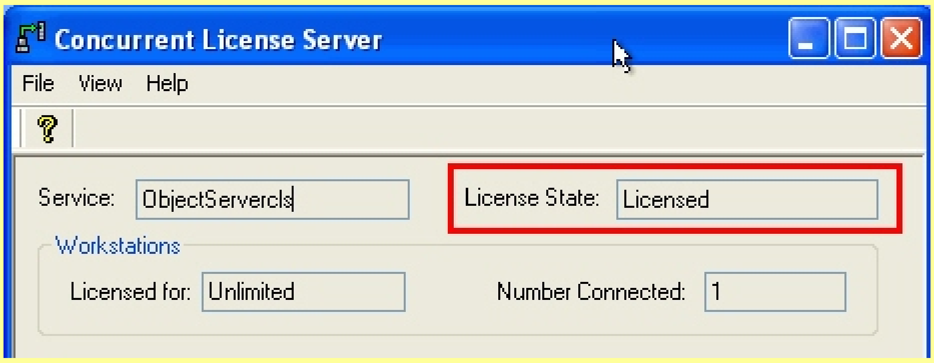
Database	Connec...	Connection Time	Que...	Ti
objectserverpc:rtrdb1	TRUE	05-NOV-2008 11:48:44.193	0	

Start Time:	05-NOV-2008 11:48:43	Server Status:	Running
Last Update:	05-NOV-2008 11:48:43	Licensed:	No

Check the following:

On the computer that is running your ObjectServer database, double-click on the Concurrent License Server icon ( ) in the system tray: The Concurrent License Server dialog displays.

Check that your server is licensed for workstation clients:-



Service: ObjectServercls

License State: Licensed

Licensed for: Unlimited

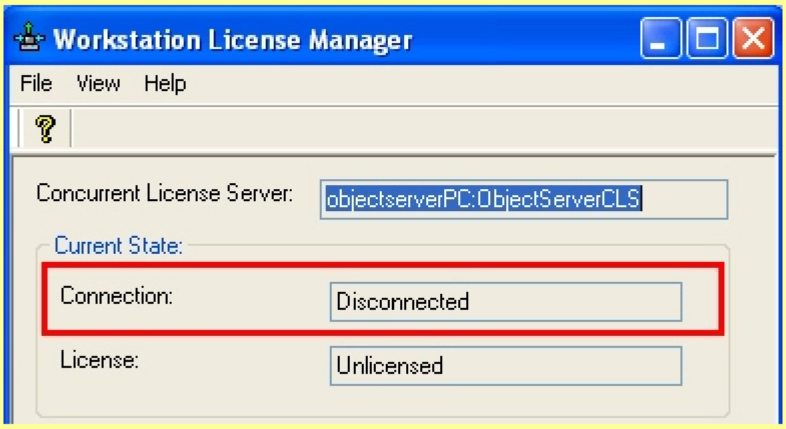
Number Connected: 1

If it is not licensed, apply for a license.

2. If your Concurrent License Server says you **are** licensed for workstation connections, the fault may be in the setup on your workstation. Double-click the Workstation License Manager icon in the system tray on your ObjectServer Workstation ( ).

   The Workstation License Manager dialog displays.

   If the Connection field displays **Disconnected**, this indicates a problem with your workstation setup.



Concurrent License Server: objectserverPC:ObjectServerCLS

Current State:

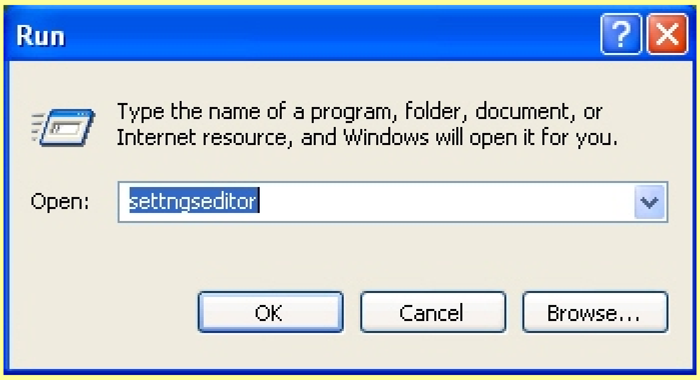
Connection: Disconnected

License: Unlicensed

   Check the following:-

   a. Are you using an alias in the Hosts file rather than the actual name of the computer that is running the ObjectServer database? If so, please do the following:-

      i. Remove the entry in the Hosts' file on the workstation machine. A Hosts entry is not required for an ObjectServer workstation to connect to a remote ObjectServer server as long as the workstation knows the actual name of the Server.
      ii. Select **Start → Run,** then type **settingseditor** into the Run dialog:

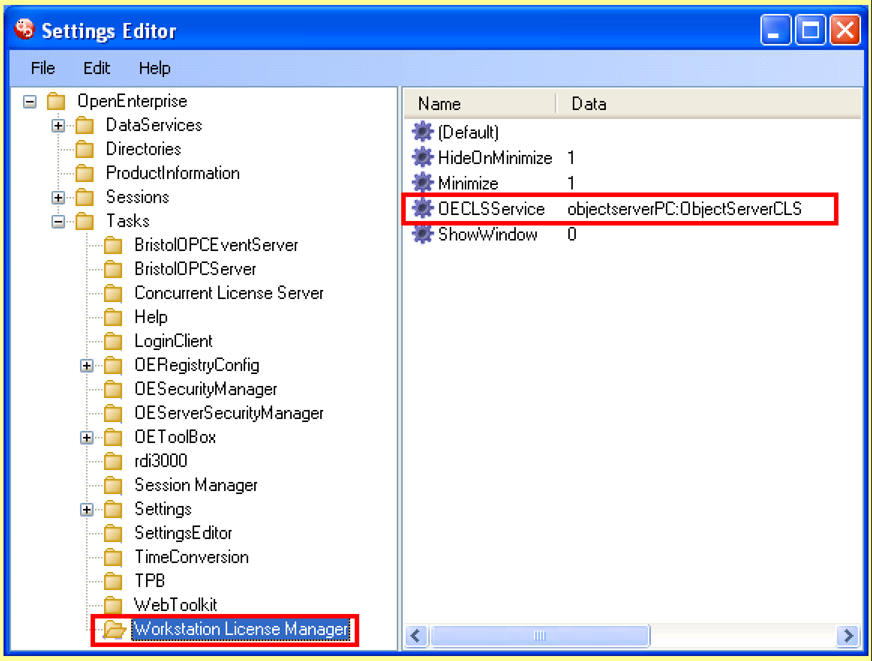


Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: settngseditor

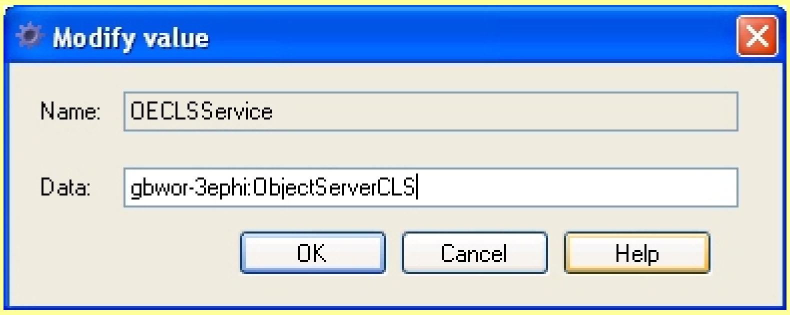
Click **OK** to open the ObjectServer Settings Editor dialog.

iii. Select the Workstation License Manager from the left pane, right-click on the OECLService value in the right pane, and select the Modify option from the shortcut menu.



Name	Data
(Default)	
HideOnMinimize	1
Minimize	1
OECLSService	objectserverPC:ObjectServerCLS
ShowWindow	0

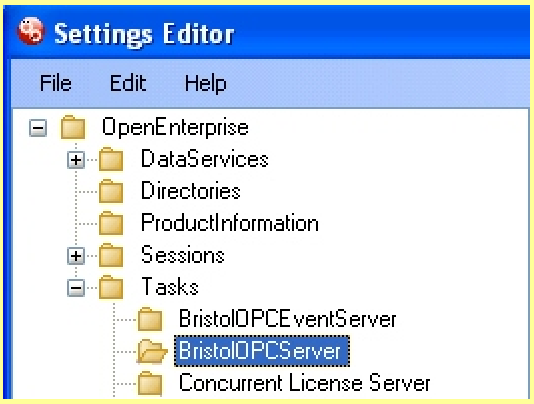
iv. Change the data to refer to the actual name of the computer that is running the ObjectServer Concurrent License Server service (ObjectServerCLS).



Name: OECLSService

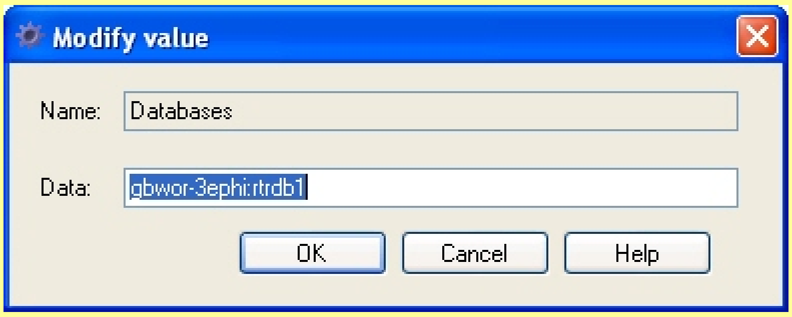
Data: gbwor-3ephi:ObjectServerCLS

v. Close this dialog and then select the BristolOPCServer key from the left pane of the Settings Editor.



vi. In the right pane, right-click the Databases value and select **Modify** from the shortcut menu.

vii. Change the data service string so that the server name part of it (the part before the colon) refers to the actual name of the machine running the ObjectServer data service. Click **OK**.



Name: Databases

Data: gbwor-3ephi:rtrdb1

viii. Close the Settings Editor.

ix. On the workstation use the Windows Task Manager to stop the OEWorkstationLicenseManager.exe and BristolOPCServer.exe processes.

x. Restart the BristolOPCServer.exe process. Both the OPC Server and the Workstation License Manager should now connect to the relevant service on the server.

Getting ObjectServer Alarm and Event Data into an OPC Client

So far we have only dealt with how to get real-time data from ObjectServer into OPC DA (Data Access) clients. We will now see how to get alarm and event data from ObjectServer into an OPC Alarm and Event client.

What's all this about different types of OPC client?
We have already explained what OPC is in *Chapter 1*. However, we haven't explained the different defined OPC standards.

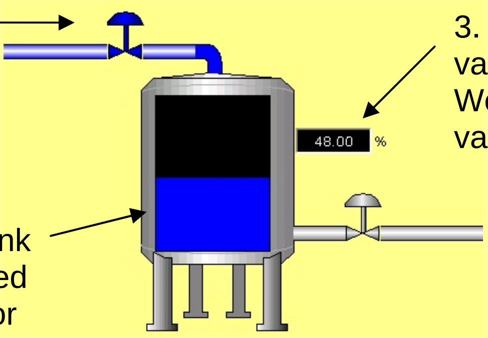
Each OPC standard defines a way of getting different types of data from field devices (such as Bristol controllers) into an OPC client for viewing.

The two types of data that ObjectServer accesses are **real-time** and **alarm-event** data. These two types of data are defined in the OPC DA (Direct Access) and OPC Alarm and Event standards. Let's see what these different types of data are.

Real-time Data
Read-time data occurs when the control strategy program or an operator sends commands to the objects in the system, and reads back the resulting values from the objects. The OPC DA standard is made for visualizing process signal values.

2. This valve changes color from grey to blue in the OPC DA client to indicate it is open

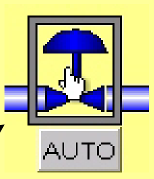
1. The control strategy program tells this valve to open.



3. As the Tank Level value begins to rise. We see the changing value here as text.

4. We also see the Tank Level value represented here as this dynamic or animated rectangle grows in size.

5. We can configure some objects on the display so that we can send values down to the controller. The square around this valve and the cursor shape tells us we could turn it off, for instance.



6. But, one thing you can't do with OPC DA data is visualize alarms. This is because alarms are special objects created by the controller in response to specific signal values. These objects can only be visualized by clients that can handle alarm-event data. These are called **OPC Alarm and Event clients**.

Alarm Data

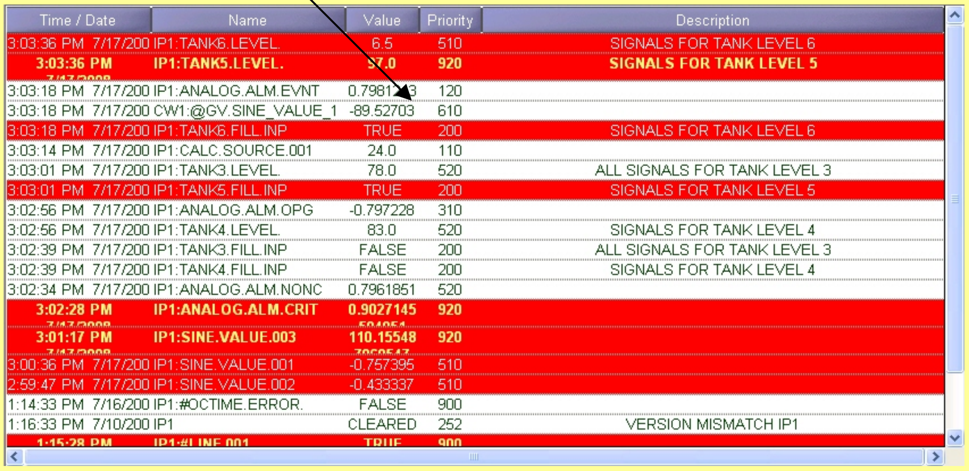
We are all familiar with home alarm systems. When a window or door contact on a home alarm system opens...

The alarm siren goes off.

Alarm data can be thought of as warning messages that switch on or off according to process values. In this way, you can create an alarm if the level of fluid in a tank goes higher than a certain measurement. The actual level measurement and the alarm are related, like the door contact and the siren, but they are separate objects, and their values are determined by different agents.

OPC Alarm and Event clients can retrieve and display alarm and event data that process control data raises. This kind of visualization is usually shown as a table (think of it as a list of messages) rather than as animated graphic objects. Below is an example display from AlarmWorX32, the ICONICS OPC Alarm and Event client. .

Alarms are listed in tabular format.

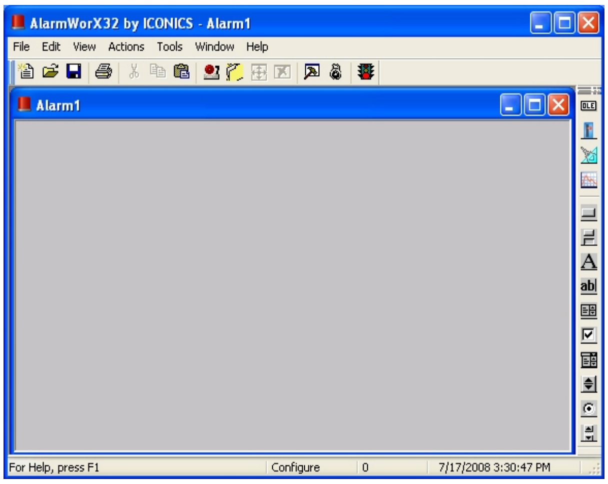


Time / Date	Name	Value	Priority	Description
3:03:36 PM 7/17/200	IP1:TANK6.LEVEL.	6.5	510	SIGNALS FOR TANK LEVEL 6
3:03:36 PM	**IP1:TANK5.LEVEL.**	**97.0**	**920**	**SIGNALS FOR TANK LEVEL 5**
3:03:18 PM 7/17/200	IP1:ANALOG.ALM.EVNT	0.7981[illegible]3	120	
3:03:18 PM 7/17/200	CW1:@GV.SINE_VALUE_1	-89.52703	610	
3:03:18 PM 7/17/200	IP1:TANK6.FILL.INP	TRUE	200	SIGNALS FOR TANK LEVEL 6
3:03:14 PM 7/17/200	IP1:CALC.SOURCE.001	24.0	110	
3:03:01 PM 7/17/200	IP1:TANK3.LEVEL.	78.0	520	ALL SIGNALS FOR TANK LEVEL 3
3:03:01 PM 7/17/200	IP1:TANK5.FILL.INP	TRUE	200	SIGNALS FOR TANK LEVEL 5
3:02:56 PM 7/17/200	IP1:ANALOG.ALM.OPG	-0.797228	310	
3:02:56 PM 7/17/200	IP1:TANK4.LEVEL.	83.0	520	SIGNALS FOR TANK LEVEL 4
3:02:39 PM 7/17/200	IP1:TANK3.FILL.INP	FALSE	200	ALL SIGNALS FOR TANK LEVEL 3
3:02:39 PM 7/17/200	IP1:TANK4.FILL.INP	FALSE	200	SIGNALS FOR TANK LEVEL 4
3:02:34 PM 7/17/200	IP1:ANALOG.ALM.NONC	0.7961851	520	
3:02:28 PM	**IP1:ANALOG.ALM.CRIT**	**0.9027145**	**920**	
3:01:17 PM	**IP1:SINE.VALUE.003**	**110.15548**	**920**	
3:00:36 PM 7/17/200	IP1:SINE.VALUE.001	-0.757395	510	
2:59:47 PM 7/17/200	IP1:SINE.VALUE.002	-0.433337	510	
1:14:33 PM 7/16/200	IP1:#OCTIME.ERROR.	FALSE	900	
1:16:33 PM 7/10/200	IP1	CLEARED	252	VERSION MISMATCH IP1
1:15:28 PM	**IP1:#LINE.001**	**TRUE**	**900**	

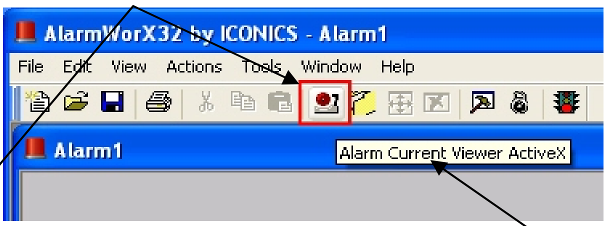
Now, let’s learn how to get OPC Alarm and Event data from ObjectServer into AlarmWorX32.

Getting ObjectServer OPC Alarm and Event Data into AlarmWorX32

1. Select **Start → All Programs → ICONICS GENESIS32 → AlarmWorX32 → AlarmWorX32.** The initial AlarmWorX32 window displays.

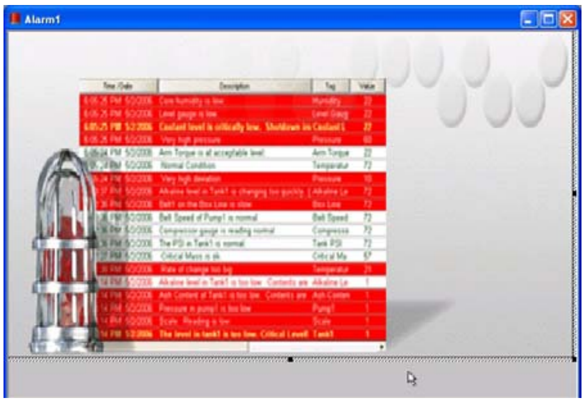

AlarmWorX32 by ICONICS - Alarm1
File Edit View Actions Tools Window Help
Alarm1
For Help, press F1
Configure
0
7/17/2008 3:30:47 PM

2. Next, you need to get the AlarmWorX ActiveX control embedded into the window. The quickest way to do this is to click this button on the toolbar.

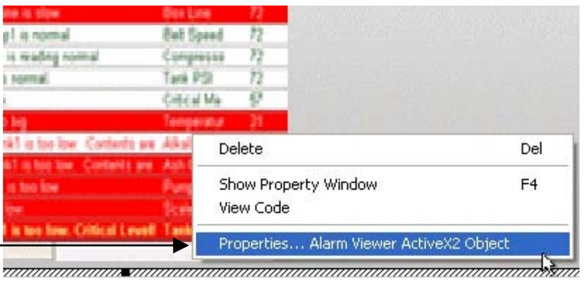

AlarmWorX32 by ICONICS - Alarm1
File Edit View Actions Tools Window Help
Alarm1
Alarm Current Viewer ActiveX

If you hold the mouse over the button, the tooltip **Alarm Current Viewer ActiveX** displays.

3. The ActiveX control is added to the window. Resize it to fit the window.

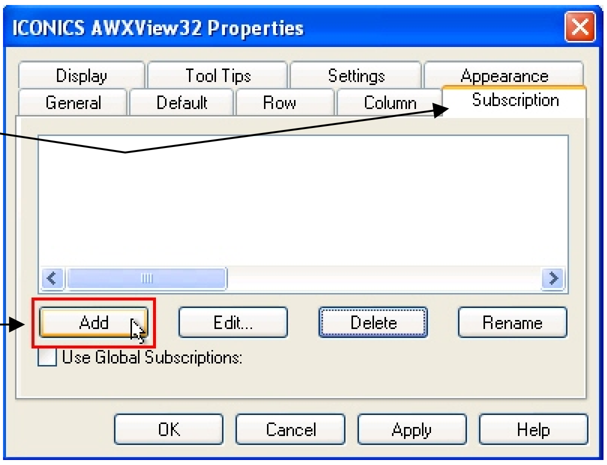


4. You now need to configure the ActiveX control, so right-click it and select the **Properties… Alarm Viewer ActiveX2 Object**' option.



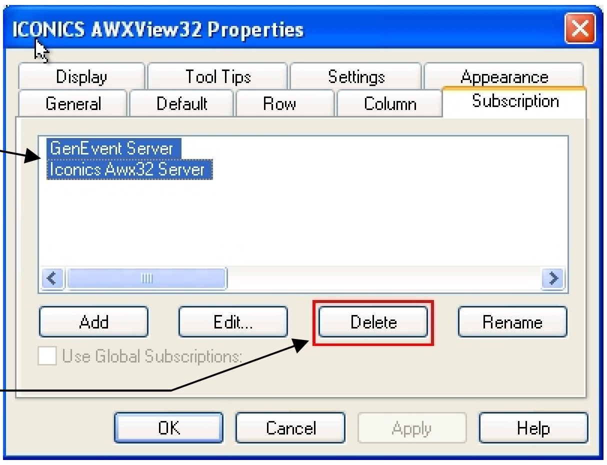
5. Select the Subscription tab from the ICONICS AWXView32 Properties dialog.

   Click **Add**.

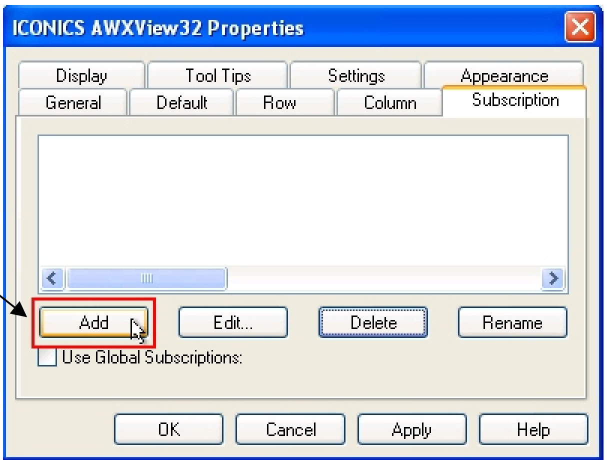


6. We need to delete the default server subscriptions, so select them both.

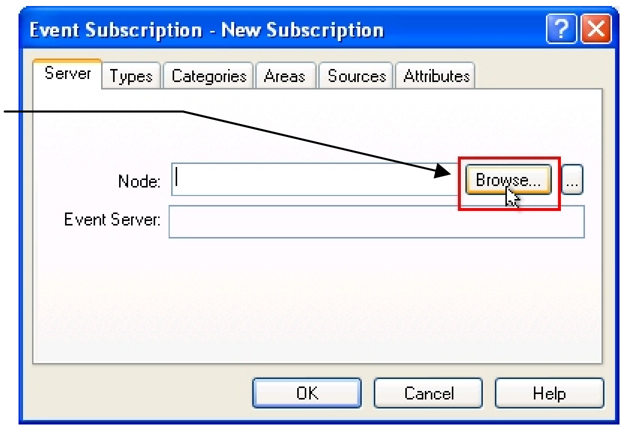
   Then click **Delete**.

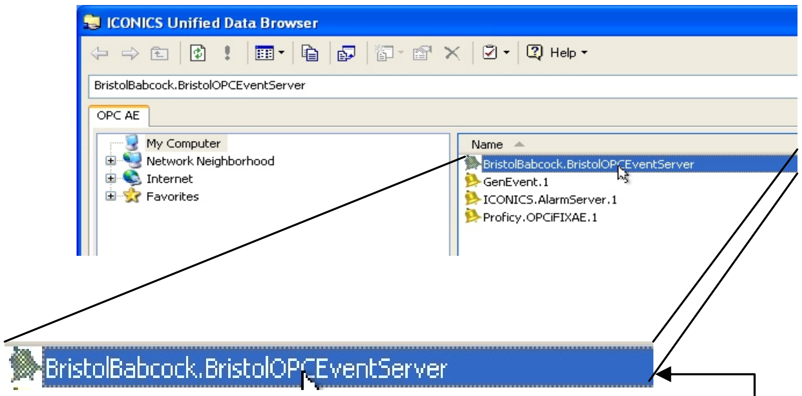


7. Now we need to subscribe to the ObjectServer OPC Alarm and Event server, so click **Add**.



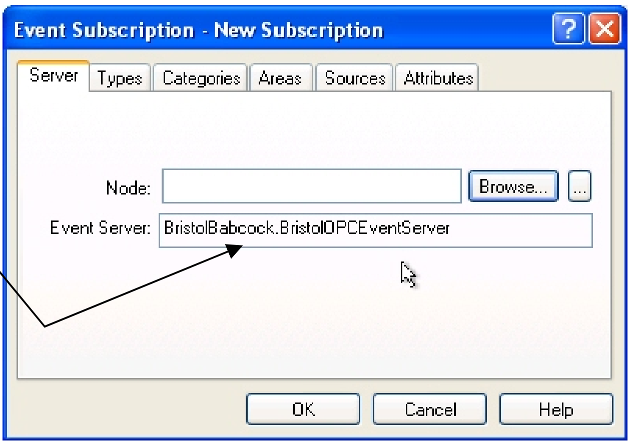
8. Click **Browse** on the Server tab of the Event Subscription – New Subscription dialog in order to browse for the ObjectServer OPC server.



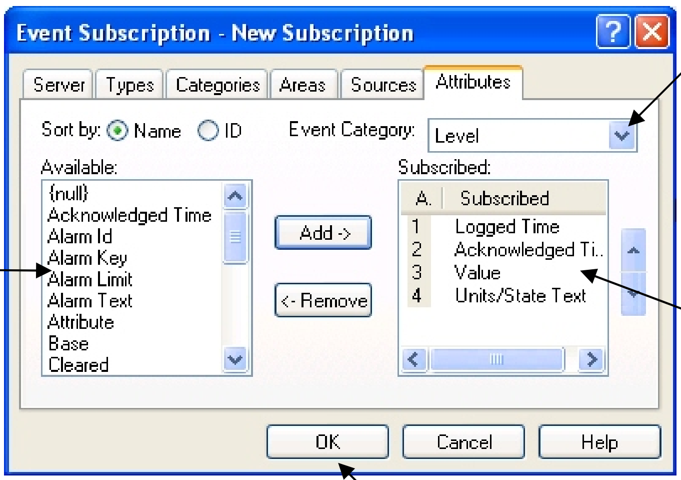

ICONICS Unified Data Browser
Help
BristolBabcock.BristolOPCEventServer
OPC AE
My Computer
Network Neighborhood
Internet
Favorites
Name
BristolBabcock.BristolOPCEventServer
GenEvent.1
ICONICS.AlarmServer.1
Proficy.OPCiFIXAE.1
BristolBabcock.BristolOPCEventServer

9. Select **BristolBabcock.BristolOPCEventServer** from the OPC Servers listed in the ICONICS Unified Data Browser.
Then click **OK**.

10. The ObjectServer Event Server is now appears in the Event Server field on the Server tab.



8. Now you need to select some subscription attributes. These are special attributes that the OPC Server exposes. Select the Attributes tab and select the attributes you want from the Available list.

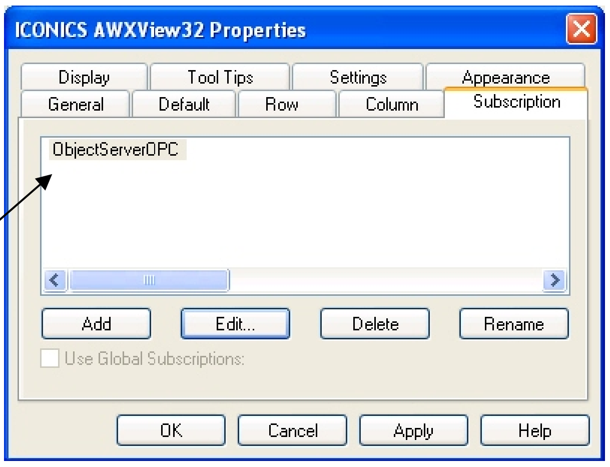


You need to select the Deviation, Discrete, and Level 'Event Category' from this drop-down list and add the subscribed attributes for each category.

The subscribed attributes are added to the Subscribed list.

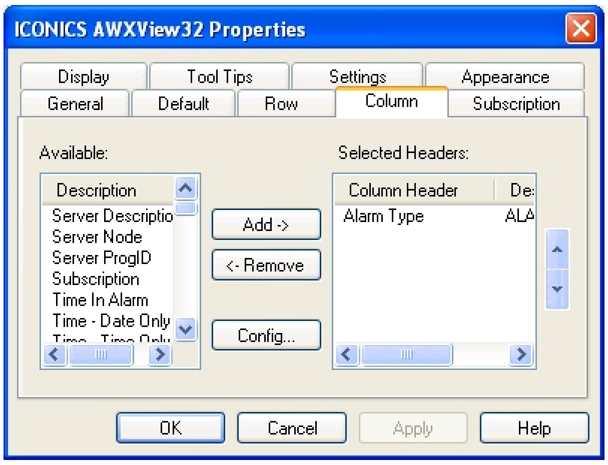
When you have finished, click **OK** to return to the AVXView32Properties' dialog.

We have renamed the new subscription as "ObjectServerOPC" to remind us that we are subscribing to the ObjectServer OPC Alarm/Event Server.

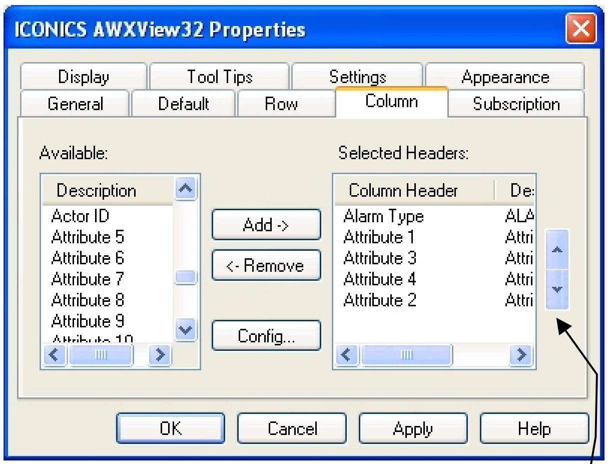


9. Now we need to configure the actual attributes that we want to display in our alarm list.

   Remove all of the default column headers (except Alarm Type, which you cannot remove) from the Selected Headers list by selecting them and clicking **<- Remove**.

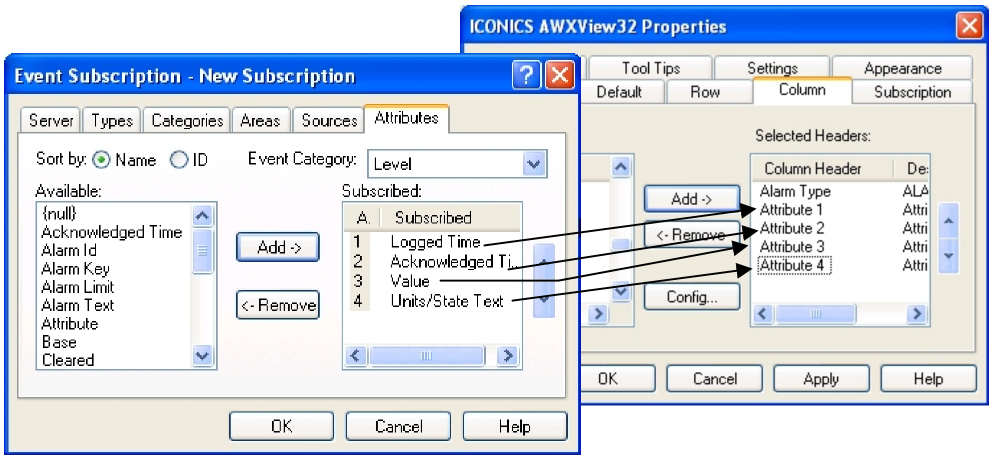


10. Next, select Attribute 1 to 4 from the Available list and move them over to the Selected Headers list by clicking **Add ->**.

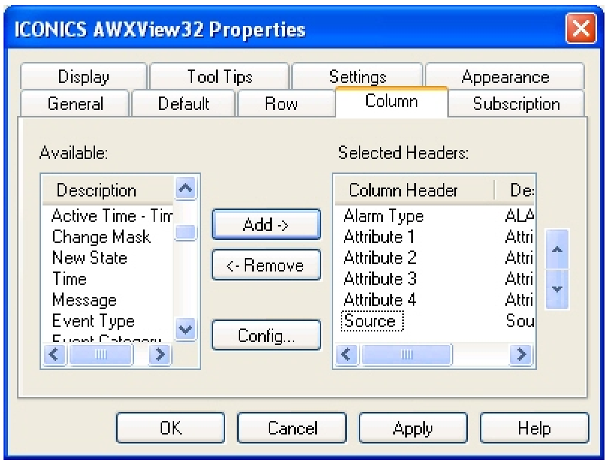


You can re-order the selected columns by using the up and down arrows here.

The reason we have done this is that the four attributes we selected on the Attributes tab of the Event Subscription dialog are represented on the Column tab in the dialog above as Attributes 1 to 4.

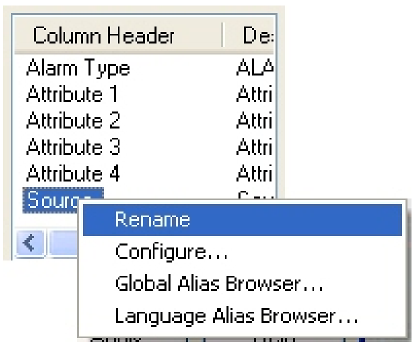


11. Add the Source column header from the Available list.



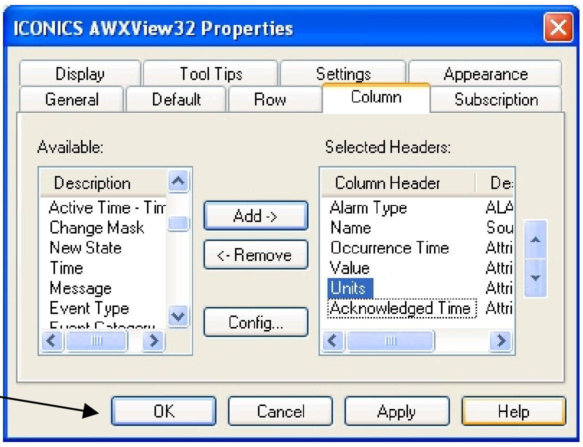
12. Now right-click to rename these attributes:

Source = Name
Attribute 4 = Units
Attribute 3 = Value
Attribute 2 = Acknowledged Time
Attribute 1 = Occurrence Time

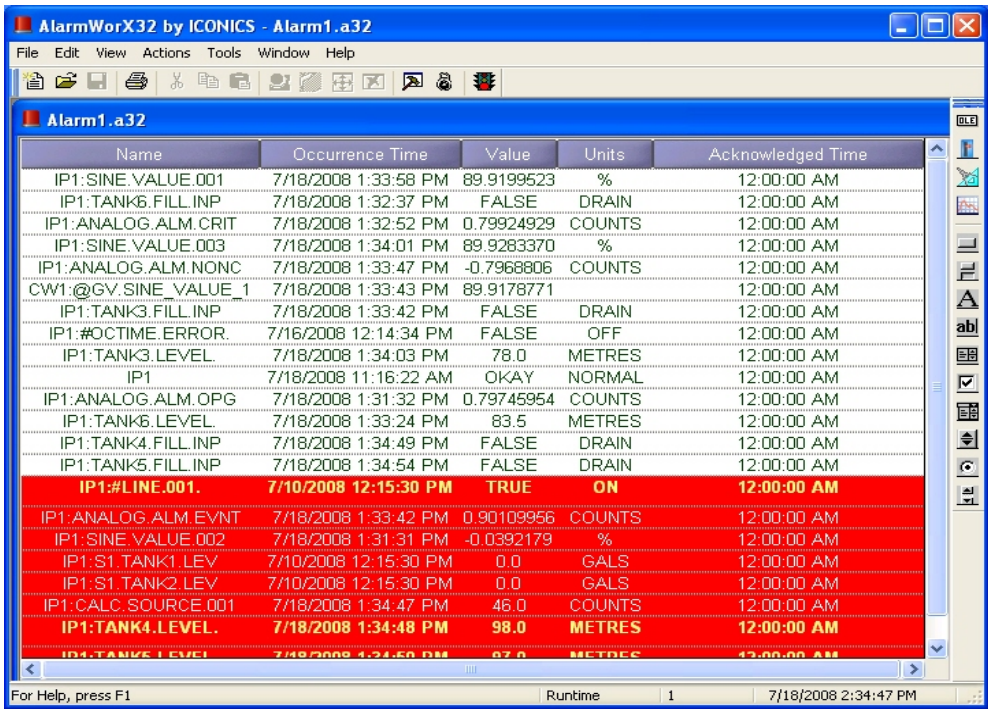


13. Finally, re-order the list of columns so that they appear where you want them to in the list. The columns at the top appear to the left of the list. The Alarm Type column is not visible in the list.

Click **OK** to return to the main AlarmWorX32 window.



Select **Actions → Animation Mode** from the menu bar to begin visualizing ObjectServer alarm and event data.


AlarmWorX32 by ICONICS - Alarm1.a32
File Edit View Actions Tools Window Help
Alarm1.a32

Name	Occurrence Time	Value	Units	Acknowledged Time
IP1:SINE.VALUE.001	7/18/2008 1:33:58 PM	89.9199523	%	12:00:00 AM
IP1:TANK6.FILL.INP	7/18/2008 1:32:37 PM	FALSE	DRAIN	12:00:00 AM
IP1:ANALOG.ALM.CRIT	7/18/2008 1:32:52 PM	0.79924929	COUNTS	12:00:00 AM
IP1:SINE.VALUE.003	7/18/2008 1:34:01 PM	89.9283370	%	12:00:00 AM
IP1:ANALOG.ALM.NONC	7/18/2008 1:33:47 PM	-0.7968806	COUNTS	12:00:00 AM
CW1:@GV.SINE_VALUE_1	7/18/2008 1:33:43 PM	89.9178771		12:00:00 AM
IP1:TANK3.FILL.INP	7/18/2008 1:33:42 PM	FALSE	DRAIN	12:00:00 AM
IP1:#OCTIME.ERROR.	7/16/2008 12:14:34 PM	FALSE	OFF	12:00:00 AM
IP1:TANK3.LEVEL.	7/18/2008 1:34:03 PM	78.0	METRES	12:00:00 AM
IP1	7/18/2008 11:16:22 AM	OKAY	NORMAL	12:00:00 AM
IP1:ANALOG.ALM.OPG	7/18/2008 1:31:32 PM	0.79745954	COUNTS	12:00:00 AM
IP1:TANK6.LEVEL.	7/18/2008 1:33:24 PM	83.5	METRES	12:00:00 AM
IP1:TANK4.FILL.INP	7/18/2008 1:34:49 PM	FALSE	DRAIN	12:00:00 AM
IP1:TANK5.FILL.INP	7/18/2008 1:34:54 PM	FALSE	DRAIN	12:00:00 AM
IP1:#LINE.001.	**7/10/2008 12:15:30 PM**	**TRUE**	**ON**	**12:00:00 AM**
IP1:ANALOG.ALM.EVNT	7/18/2008 1:33:42 PM	0.90109956	COUNTS	12:00:00 AM
IP1:SINE.VALUE.002	7/18/2008 1:31:31 PM	-0.0392179	%	12:00:00 AM
IP1:S1.TANK1.LEV	7/10/2008 12:15:30 PM	0.0	GALS	12:00:00 AM
IP1:S1.TANK2.LEV	7/10/2008 12:15:30 PM	0.0	GALS	12:00:00 AM
IP1:CALC.SOURCE.001	7/18/2008 1:34:47 PM	46.0	COUNTS	12:00:00 AM
IP1:TANK4.LEVEL.	**7/18/2008 1:34:48 PM**	**98.0**	**METRES**	**12:00:00 AM**

For Help, press F1 Runtime 1 7/18/2008 2:34:47 PM

That's it for this chapter. You have now learned how to:

- Get ObjectServer OPC Data Access data into displays created with Genesis32, InTouch, and iFix OPC clients.
- Setup your OPC client on the same computer as your ObjectServer database.
- Setup your OPC client on a remote workstation.
- Get OPC Alarm and Event data from ObjectServer into the ICONICS AlarmWorX32 OPC Alarm and Event client.

We shall now look at how it is also possible to view ObjectServer data over the Internet using WebToolkit rather than a third-party OPC client.

[This page is left intentionally blank]

Chapter 8 – Troubleshooting and Maintenance

This chapter covers basic troubleshooting and maintenance issues, such as:

- Data Collection
- OPC Server
- Workstation Licensing
- System Maintenance

Data Collection

Without data collection no HMI or SCADA system can function properly, so it is essential that you know how to:

1. Confirm that data is being collected
2. Fix ACCOL version mismatches
3. Troubleshoot RBE collection failure
4. Troubleshoot alarm collection failure

Confirming that Data is Being Collected

This section briefly describes how to use the Monitor tool to test that ObjectServer is collecting data from its RTU(s).

1. Start the Monitor tool (refer to the *Start the Monitor* section of *Chapter 6 - Monitoring Controllers*).

2. Check that polled data is being collected.
   Open the Device Status view and: (refer to the *Viewing Device Status* section of *Chapter 6 - Monitoring Controllers*).
   a. Ensure that the status for each device (controller) is zero (0). This indicates that communications with the controller is taking place.
   b. Ensure that the “pollsdone” field for each controller contains a positive integer that is increasing at a regular rate.
   c. Ensure that the “pollfailed “field for each controller shows zero (0), indicating that polling is not failing.

3. Troubleshooting Polling Failures
   If the value in the “pollsfailed” field for any of your controllers is gradually rising, the most common cause is a control strategy file version mismatch for that controller. To check for a version mismatch, open the ACCOL Versions view. If the “signal”, “msds”, and “rtu” values for each controller do not match, you have a version mismatch for that controller. Consult *Fixing ACCOL Version Mismatches* in this chapter **to** find out how to fix this problem.

4. Check that RBE data is being collected.
   Open the RBE view (see the *RBE* section of *Chapter 6 - Monitoring Controllers* for help).
   a. For each controller that is configured to collect RBE data, check that the “messages” and “reports” fields are updating
   b. Check that there is a time in the ”lastrsntimestamp” field.

5. Check that alarm data is being collected
   Open the Remote Alarm Reports view (refer to *Viewing Alarm Stats* in *Chapter 6 - Monitoring Controllers*).
   a. Ensure that the “alarmreports” field for each controller that is collecting alarm data is being updated.
   b. Ensure that the “lastalarmtimestamp” field is updating.

6. Check that signal values are being updated
   Select **Signals → Analogs** from the Monitor’s menu bar to open the Analog Signals view.
   a. Watch the value field of any signal that you know is changing in the RTU to verify that the values are updating.

Fixing ACCOL/ControlWave Version Mismatches

If you download a new control strategy file to an RTU that ObjectServer is polling, a version mismatch occurs unless the Signal Builder and Template Builder are both running in monitor mode on the server. Polled data collection is affected until you resolve the mismatch. This is how to resolve a version mismatch.

1. Check that polled data is being collected (see step 2 of *Confirming that Data is Being Collected* above). If no data is being collected:

2. Run Signal Builder.
   a. Open the Toolbox (select **Start → Programs → OpenBSI Tools → ObjectServer → Toolbox**.
   b. Double-click **Signal Builder**.
   c. Click **Build** or **Build All** on the main dialog as appropriate.

3. Run Template Builder:
   a. Double-click **Template Builder** within the Toolbox.
   b. Click **Build** or **Build All** as appropriate.

This should fix any ACCOL/ControlWave version mismatches. All signals in the database then update to match the content in the control strategy load files.

Running the NW3000 Signal Builder and Template Builder in Monitor Mode

To guard against control strategy file version mismatches, which hamper data collection, you can run the “NW3000 Signal Builder” (DBB) and “NW300 Template Builder” (TPB) as part of the ObjectServer session in monitor mode. If a version mismatch occurs in this mode, the DBB and TPB detect it and correct the problem automatically.

1. Select **Session → Add Task** from Session Manager.
2. Add **–m** as an argument in the Program Arguments field of the first page of the Task wizard. This sets the command line parameters to run the DBB and TPB in monitor mode. (See the DBB and TPB Help files for more information on the available command line arguments.)

3. Click **Help** to read the online help files on each page of the Task wizard to finish adding the DBB and TPB tasks to the ObjectServer session.

Troubleshooting RBE Failure

Unlike polled signals, RBE signals are not affected by control strategy version mismatches. Once you set them up, they generally continue to send data to the designated destination server. If you are having trouble with RBE signals, however, check the following:-

1. Check the device's RBE module.
   a) In ACCOL
      i. Open NetView
      ii. Select the RTU from the Node window.
      iii. Right-click and select **RTU → Workbench**.
      iv. In Workbench, right-click **Task 0** and select Edit Code
      v. Ensure that the Mode terminal is set to **1.0**.

   b) In IEC 61131
      i. Open ControlWave Designer
      ii. Ensure that there is a program organization unit (POU) that contains the RBE module.
      iii. Ensure that there is a cyclical task that is running the POU with an interval of about one second.

      .
2. Check the IP connection and make sure that the RTU sends RBE reports to the ObjectServer server.
   a) Open NetView.
   b) Select the RTU from the Node window.
   c) Right-click and select **Properties**.
   d) Select the **IP** tab.
   e) At the bottom of the IP tab ensure the RBE pane lists the IP address of the ObjectServer server.
   f) If the RBE pane contains a different IP address, remove the address by selecting it and pressing the Delete key on the keyboard.
   g) Click **Insert** and add the correct IP address.
   h) Click **OK** at the bottom of the dialog.
   i) Right-click and select **RTU → RTU Configuration Parameters**.
   j) Select the **IP Parameters** tab.
   k) In the NHP section ensure that the IP ADDR A field contains the IP address of the ObjectServer server.
   l) If not, enter the ObjectServer server's IP address into the four boxes.
   m) Click **Save to RTU** on the right side of the dialog.
   n) Depending on your version of NetView, you may need to switch the RTU off and then on again.

Troubleshooting Alarm Collection Failure

Remote alarm signals are not affected by control strategy version mismatches. Once you set them up, they generally continue to send data to the designated destination server. If you are having trouble with alarm signals, however, check the following:-

1. Check the IP connection, making sure that the control strategy file is set to send Remote Alarm reports to the Destination Server (the server that is running the ObjectServer database).
   a. Open NetView.
   b. Select the RTU from the Node window.
   c. Right-click and select **Properties.**
   d. Select the **IP** tab.
   e. At the bottom of the IP tab ensure the Alarms pane lists the IP address of the ObjectServer server.
   f. If the Alarms pane contains a different IP address select the address and press the Delete key on the keyboard to remove the address.
   g. Click **Insert** and add the correct IP address.
   h. Click **OK** at the bottom of the dialog.
   i. Open NetView.
   j. Select the RTU from the Node window.
   k. Right click and select **RTU → RTU Configuration Parameters.**
   l. Select the **IP Parameters** tab.
   m. In the NHP section ensure that the IP ADDR A field contains the IP address of the ObjectServer server.
   n. If not, enter the ObjectServer server’s IP address into the four boxes.
   o. Click **Save to RTU** on the right of the dialog.
   p. Depending on your version of NetView, you may need to switch the RTU off and then on again.

OPC Server

The controllers may be creating data, and the ObjectServer database may be collecting that data, but if the OPC server is not running or connected to the database, the HMI (OPC client) cannot to visualize that data.

Starting the OPC Server

Start the ObjectServer OPC server either automatically or manually.

Automatically Starting the OPC Server

The OPC standard is based on a server-client model. The purpose of OPC servers is to serve data to OPC clients. Strangely, the server is a servant itself and the client is the master. When the client rings a bell for service, the servant has to wake up and serve the client. The same thing happens when an OPC client becomes active and requests service from the OPC server. The OPC server awakes and starts serving tags to the client. The COM interface (part of the Windows operating system) is the bell the OPC client uses.

As long as our OPC client is installed on the same PC as the ObjectServer OPC server, the OPC server starts automatically when the OPC client requests tags.

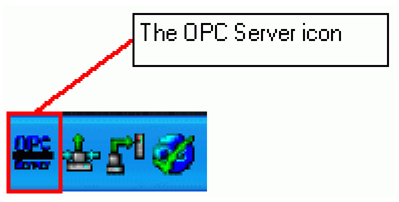
Manually Starting the OPC Server

You may want to test that the OPC server is connecting with the ObjectServer database before starting or even installing a third-party OPC client.

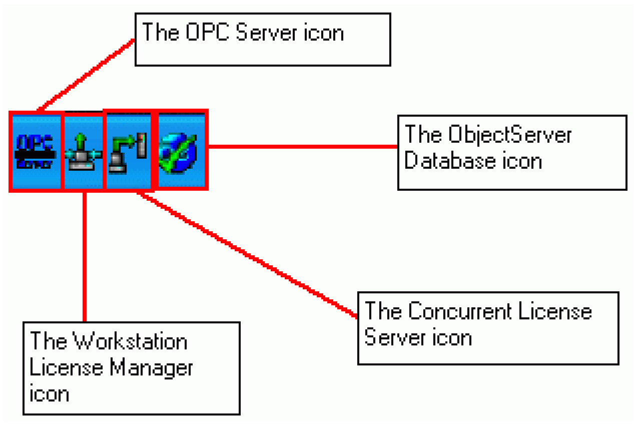
To start the OPC Data Access server manually, select **Start → Programs → OpenBSI Tools → ObjectServer → Bristol OPC Data Access Server**.

Confirming that the OPC Server is Running on a Server

To confirm that the OPC server is running, look in the System Tray ("systray") on the right of the Windows taskbar. The OPC Server icon should be present, indicating that the program is running. The following example shows how the systray looks on a PC which has both ObjectServer database and client installed:

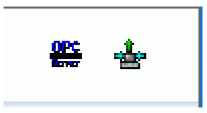


Note that the Workstation License Manager, Concurrent License Server, and ObjectServer Database are also running:


The OPC Server icon
The ObjectServer Database icon
The Concurrent License Server icon
The Workstation License Manager icon

Confirming the OPC Server is Running on a Client

If a PC has only the ObjectServer client installed, the systray looks like this when the OPC server is running:

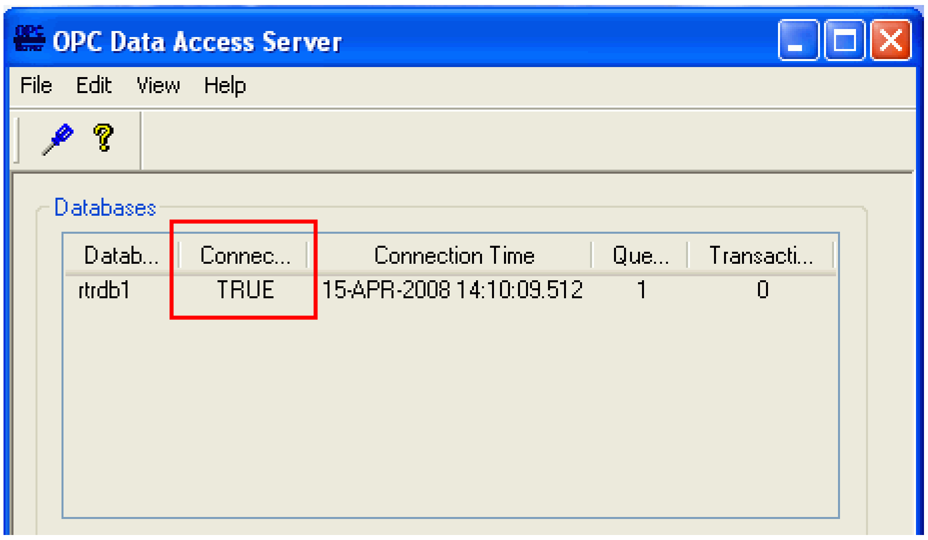


Note that only the OPC Server and Workstation License Manager are running. In this example, the Concurrent License Server and ObjectServer database are installed on the remote server.

Confirming that the OPC Server is Connected

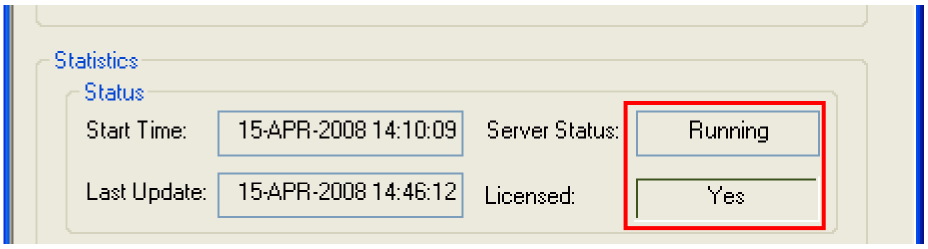
The OPC server may be running, but you also want to ensure that it is connected to the ObjectServer database.

1. Double-click on the OPC Server icon in the Windows systray.
2. Review the Connected column on the Databases list on the OPC Data Access Server dialog. The Connected column should contain **TRUE**.



Datab...	Connec...	Connection Time	Que...	Transacti...
rtrdb1	TRUE	15-APR-2008 14:10:09.512	1	0

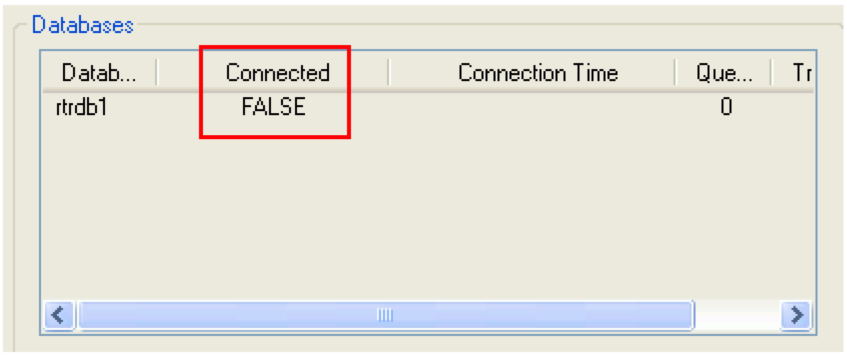
3. The Server Status field in the Status panes should display **Running**, and the Licensed field should display **Yes**:



Start Time: 15-APR-2008 14:10:09
Server Status: Running
Last Update: 15-APR-2008 14:46:12
Licensed: Yes

What to do if the OPC Server is not Connected

In this scenario, the OPC server is not connected, and the connected field displays **FALSE** in the "Databases" panel of the OPC Server's user interface (as shown in the example below).



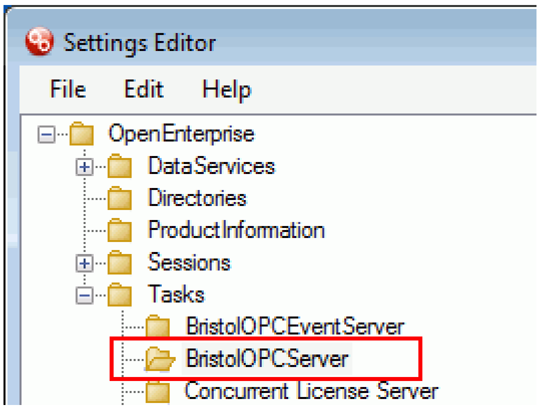
Datab...	Connected	Connection Time	Que...	Tr
rtrdb1	FALSE		0	

The following two things may have happened here:

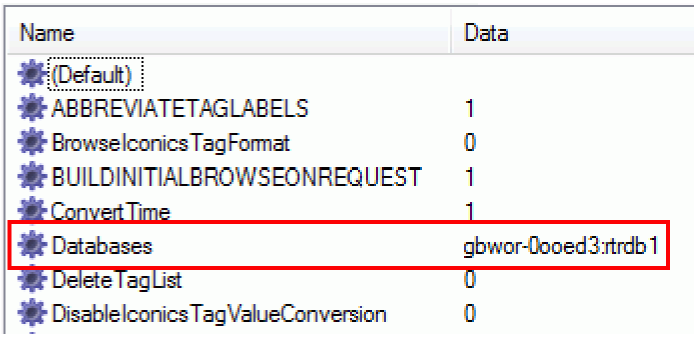
1. **This is a server installation, and the ObjectServer database has not been started.** To solve this problem, just start the database. Select **Start → Programs → OpenBSI Tools → ObjectServer → ObjectServer Session**.

2. **This is a client installation, and the network path to the remote server cannot be found**. The most likely reason is that you did not type the server name in correctly when you installed the ObjectServer client **or** you typed the name of an alias you had set up in the Hosts file rather than the server's actual domain name. To fix this:

   a. Open the Settings Editor by selecting **Start → Programs → OpenBSI Tools → ObjectServer → Settings Editor**.
   b. Select the **BristolOPCEventServer** and **BristolOPCServer** keys in turn from the Tasks key:

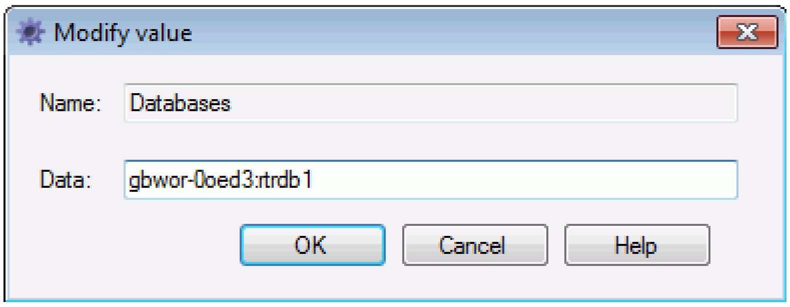


   c. Double-click **Databases** in the right pane:



Name	Data
(Default)	
ABBREVIATETAGLABELS	1
BrowseIconicsTagFormat	0
BUILDINITIALBROWSEONREQUEST	1
ConvertTime	1
Databases	gbwor-0ooed3:rtrdb1
DeleteTagList	0
DisableIconicsTagValueConversion	0

   d. Correct the server name in the Data field and click **OK**.



Name: Databases

Data: gbwor-0oed3:rtrdb1

e. Stop and restart the OPC server. It should now display as connected.
f. If you also use the Alarm and Event OPC server, follow the instructions in this list again, but at step (b) you need to select the **BristolOPCEventServer** key and change its Databases value.

Workstation Licensing

If your OPC server could not connect to the database because of an incorrect server name, you may also find that the workstation was not licensed.

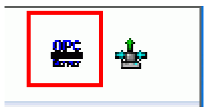
If you installed ObjectServer within the last 60 days, then this may be a configuration problem, because ObjectServer comes with a 60-day trial license. Use the instructions in the *Troubleshooting License Problems* section to fix this problem.

If you have run over the 60-day trial license time, then you must apply for a demo extension or a full license using the License Manager (select **Start → Programs → Bristol Babcock Licensing → License Manager**).

Determining if a Workstation is Licensed

If the workstation is unlicensed, the OPC server's main dialog is the first place to verify this fact.

1. Start the OPC server manually (if necessary, as described in *Manually Starting the OPC Server*).
2. Double-click the OPC Server icon.



3. On the OPC server's main dialog, verify that the licensed field displays **NO**.

Troubleshooting Licensing Problems

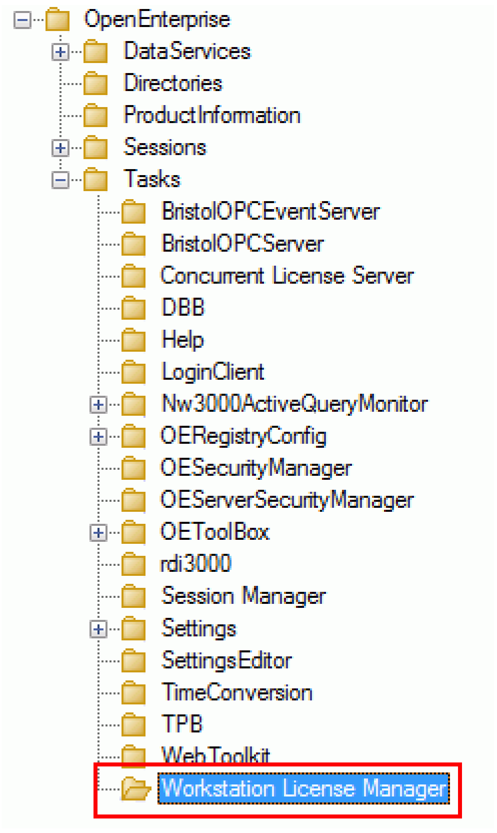
If the Licensed field in the Status section of the OPC server's user interface displays **NO**, this is probably because the Workstation License Manager (WLM) cannot connect to the Concurrent License ObjectServer server. The server may not be running or the server cannot be found on the network.

This can happen for either of two reasons:

1. During installation, a typing error occurred on the page that asks for the name of the PC that is running the ObjectServer database.
2. The name used for the PC that is running the ObjectServer database was an alias configured in the Windows Hosts file rather than the actual domain name of the server.

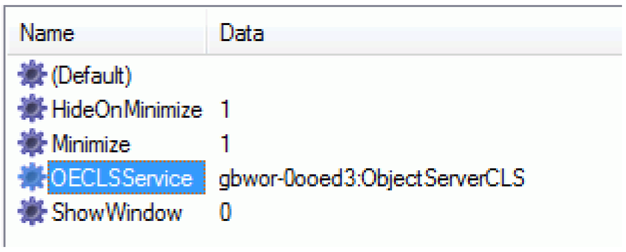
On an ObjectServer client workstation, the Workstation License Manager tries to connect to the Concurrent License Server (CLS) that is running on the server PC. If the WLM cannot find the server PC either because its name was entered incorrectly or because the name entered is an alias, the WLM cannot license the client PC.

To fix this problem follow the instructions in the *What to Do if the OPC Server is Not Connected* section, which describes how to use the Settings Editor, but in step 2b select the **Workstation License Manage**r key:



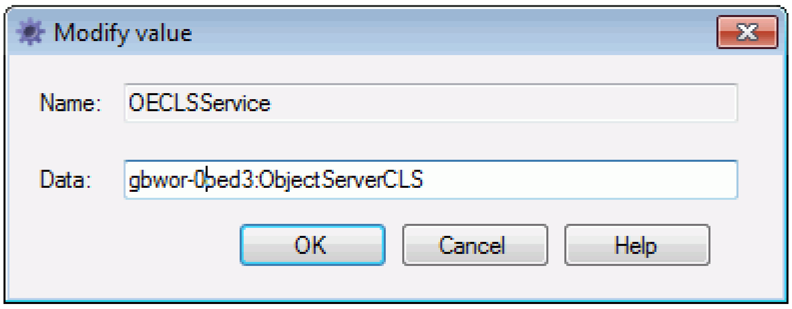
```
OpenEnterprise
    DataServices
    Directories
    ProductInformation
    Sessions
    Tasks
        BristolOPCEventServer
        BristolOPCServer
        Concurrent License Server
        DBB
        Help
        LoginClient
        Nw3000ActiveQueryMonitor
        OERegistryConfig
        OESecurityManager
        OEServerSecurityManager
        OEToolBox
        rdi3000
        Session Manager
        Settings
        SettingsEditor
        TimeConversion
        TPB
        WebToolkit
        Workstation License Manager
```

In Step 2c select the **OECLService** value from the right pane:-



Name	Data
(Default)	
HideOnMinimize	1
Minimize	1
OECLSService	gbwor-0ooed3:ObjectServerCLS
ShowWindow	0

When the Modify value dialog displays, change the value in the Data field to reflect the correct name for the server PC, and click **OK**. .



Modify value

Name: OECLSService

Data: gbwor-0ped3:ObjectServerCLS

Close the OPC server and the Workstation License Manager (you need to use the Windows Task Manager to close the WLM). When you re-open the OPC server, both it and the WLM should now connect to the Server database and be licensed.

System Maintenance

Most system maintenance issues have already been covered in previous chapters, so here is a list of maintenance tasks and the chapters which deal with them.

- How do we check that ObjectServer is collecting data from our RTUs?
  See *Chapter 5 – Monitoring Controllers*.
- What should we do if ObjectServer has stopped collecting data?
  See *Chapter 5 – Monitoring Controllers*.
- How can we change signal values using the ObjectServer Monitor tool?
  See *Chapter 5 – Monitoring Controllers*.
- How do we add or remove signals?
  See *Chapter 6 – Basic Configuration Changes*.
- How do we add or remove RTUs?
  See *Chapter 6 – Basic Configuration Change*'.
- How can we view ObjectServer data using Genesis32®, InTouch®, or iFix®?
  See *Chapter 7 – Displaying ObjectServer Data in OPC Clients*.

ObjectServer

Training Manual
D301660X412
May, 2010

Emerson Process Management
Bristol, Ltd.
Blackpole Road
Worcester, WR3 8YB
United Kingdom
Phone: +44 1905 856950
Fax: +44 1905 856969
www.emersonprocess.com/remote

Emerson Process Management
Bristol, Inc.
1100 Buckingham Street
Watertown, CT 06795
Phone: +1 (860) 945-2262
Fax: +1 (860) 945-2525
http://www.emersonprocess.com/remote

Emerson Electric Canada, Ltd.
Bristol Canada
6338 Viscount Rd.
Mississauga, Ont. L4V 1H3
Canada
Phone: 905-362-0880
Fax: 905-362-0882
http://www.emersonprocess.com/remote

Emerson Process Management
BBI, S.A. de C.V.
Homero No. 1343, 3er Piso
Col. Morales Polanco
11540 Mexico, D.F.
Mexico
Phone: (52-55)-52-81-81-12
Fax: (52-55)-52-81-81-09
http://www.emersonprocess.com/remote

Emerson Process Management
Bristol, Inc.
22 Portofino Crescent,
Grand Canals Bunbury, Western Australia 6230
Mail to: PO Box 1987 (zip 6231)
Phone: +61 (8) 9725-2355
Fax: +61 (8) 8 9725-2955
http://www.emersonprocess.com/remote

NOTICE

EMERSON
Process Management