BridgeWorX – Integrating with Remote Telemetry Units

APPLICATIONS NOTE

October 2013

Description: How to set up BridgeWorX to integrate with Remote Telemetry Units.
OS Requirement: Win XP/Server 2003/ Server 2008/Vista/Windows 7
General Requirement: General knowledge of the ICONICS BizViz Suite is required. Installation of GENESIS32 and BridgeWorX

Introduction

This Application Note explains how to use Remote Telemetry Units (RTU) with BridgeWorX. We will use the KepServerEX Server to dial out to the remote sites.

Kepware OPC Server allows you to configure communication channels to field devices with modem dial-out support.

> **NOTE:** This example uses the KepServerEX OPC Server, so if you want to follow along exactly, you should download it from Kepware.

Setup and Test the Dial-out Connection

The first thing is to setup and test the OPC communication with the KepServerEX server.

1. Create a configuration and enable the use modem checkbox in the Channel Properties of the device. More information on how to configure connecting to a Modem can be found at http://www.kepware.com/Support_Center/SupportDocuments/KEPServerEX_Modem_Use_Notes.pdf.

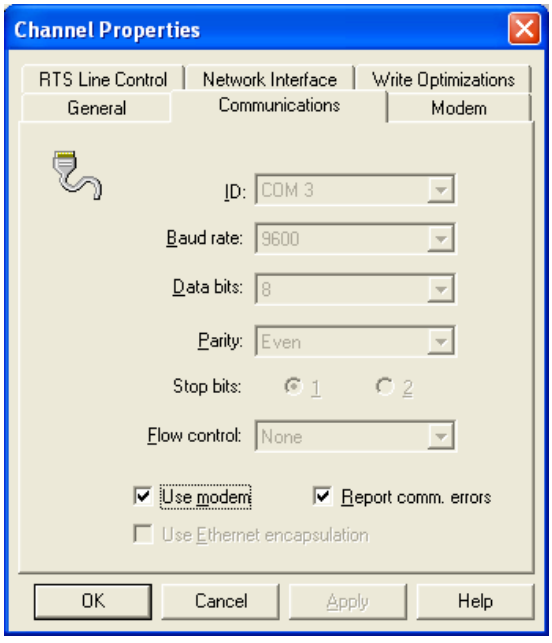

Figure 1 - Channel Properties for a Device

2. Figure 2 shows the Kepware server configuration with one channel and one device

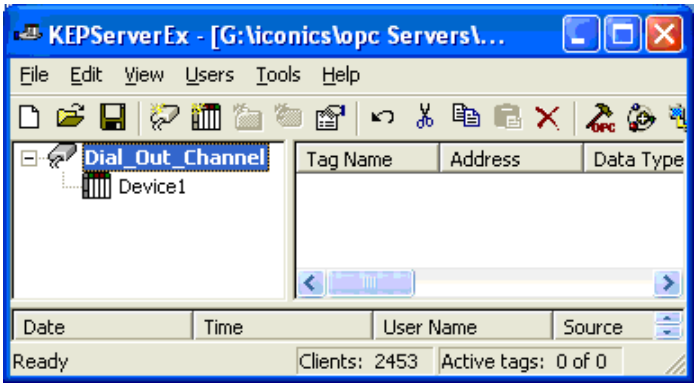

Figure 2 - Kepware Server Configuration

3. The channel is named **Dial_Out_Channel** and it has **Device1** configured. The **ICONICS Unified Data Browser** can be used to browse to the **_Modem** folder in the **Dial_Out_Channel**. The _Modem folder contains the control tags for the modem.
4. To test the Dial-Out configuration we will create a display with GraphWorX32. We will use the following tags in the display:

   - **KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Phone Number**
     This tag is used to specify the phone number of the remote site.
   - **KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Dial**
     When writing a 1 to this tag, it will dial out
   - **KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Hangup**
     This tag is used to hang-up the modem connection
   - **KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Status**
     This tag shows the connection status as a number. The values are explained in Table 1.
   - **KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._StringStatus**
     This tag shows the connection status as a string

Table 1 - Status Tag Values

Value	Meaning
0	Un-initialized, the channel is not usable
1	Initialized, no line open
3	Line open and the state is idle
7	Connected
11	Calling
19	Answering

5. The user can enter a phone number, press the dial button and see the status of the connection. When connected, the OPC tags will present live device data.

Figure 3 - Sample GraphWorX32 Display with Modem Information

Using BridgeWorX to Dial-out

Now, imagine that the RTU is a remote pumping station that requires periodic connection to collect relevant data and store in a database. This section explains how BridgeWorX could be used for the project implementation.

Data Connectors

The first thing we have to do in BridgeWorX is to configure the Data Connectors. These are the OPC tags and database tables we are going to use in the Transaction diagram. We need to configure the following OPC tags, especially the Modem handing tags, with both read and write permission.

The modem handing tags:

KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._PhoneNumber
KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Dial
KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Hangup
KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Status

The OPC tag from the Remote Telemetry Unit:

KEPware.KEPServerEx.V4\Dial_Out_Channel.Device1.test

To log the data, we need a database as well. In this example, we will make a simple Access Database with one table containing three columns, as shown in Figure 4.

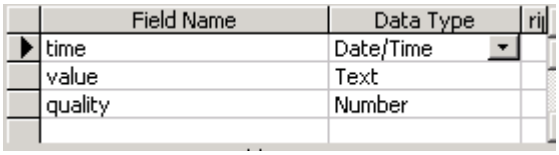

Figure 4 - Simple Access Database for Logging RTU Data

For the transaction, we will need a counter and a date/time field. The easiest way is to configure Global Variables. The **counter** Global Variable is configured as an INT (integer). The date/time Global Variable is configured shown in Figure 5.

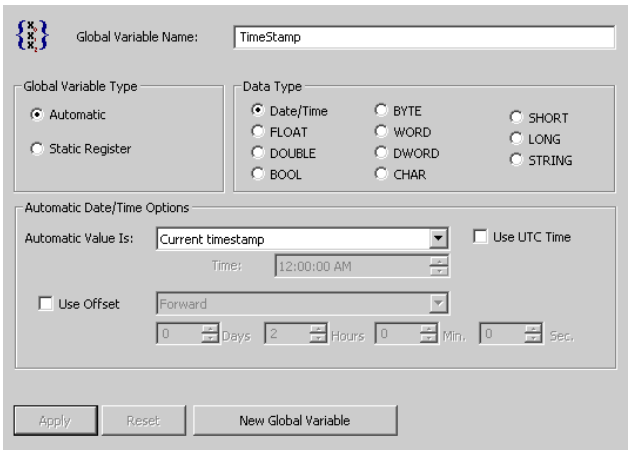

Figure 5 - TimeStamp Global Variable Configuration

Transaction Diagram

The Transaction Diagram can be created. The complete transaction will have a work flow like the one shown in

Table 2 - Transaction Workflow

Step	Actions
1	Trigger the transaction
	↓
2	Set Dialing parameters and Dial out
	↓
3	Wait 20 seconds and test if connection is mad
	↓
4	Wait 10 seconds to get fresh OPC values
	↓
5	Store the OPC values to the database
	↓
6	Close the connection

Step 1 can be automatically triggered by BridgeWorX or it can be triggered on demand by the User. In this example, we will use an On Demand trigger (pushbutton execution by the user).

Step 2 is to configure the phone number to be dialed and the dialing process. We will use an **OPC to OPC** block for this with the following OPC to OPC data mapping (note that we are using this block to set the counter to zero as well)

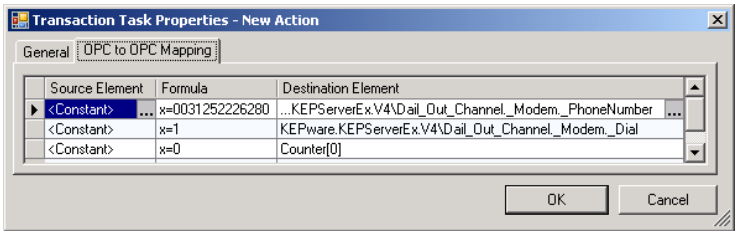

Figure 6 - OPC to COP Block Data Mapping

Step 3 is a delay block that waits 20 seconds followed by a Conditional Branch Block to check the Modem Status. The diagram now looks like Figure 7.

Figure 7 - Transaction Diagram for the First Three Steps

The Check Line Status Block is a Conditional Branch Block that will check the status of the connection. If this OPC tag equals "7" it will return TRUE. The expression to use should read similar to the one below.

```
x= {{Modem.In[6]
'KEPware.KEPServerEx.V4\Dial_Out_Channel._Modem._Status'}}
== 7
```

If the expression returns FALSE, we can have it loop and test the status again a bit later (in this example, five seconds). This loop is protected by a counter to allow only a maximum number of retrials (10 in this case).

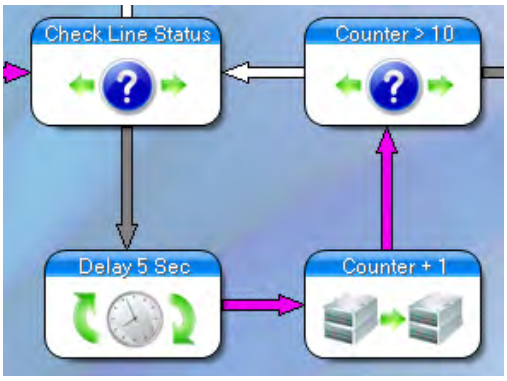

Figure 8 - Transaction Retry Loop

The Counter + 1 block is configured as shown in Figure 9.

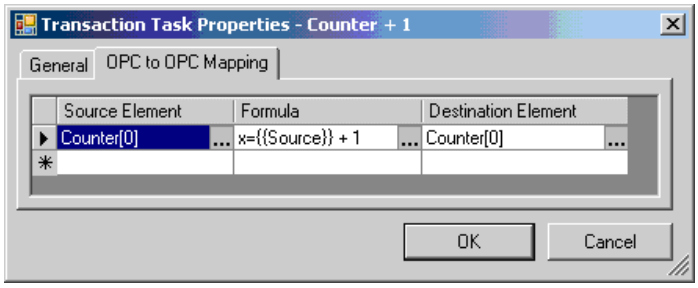

Figure 9 - Counter + 1 Block Configuration

The Expression in the Counter > 10 Block uses the expression: x= {{Counter\Counter\Counter[0]}} > 10

NOTE: Alternatively you could use an Automatic Global Variable with the option to automatically increment the value each time the Global Variable is accessed. This is another way of emulating a "For" loop, and would eliminate the OPC to OPC block from the picture.

If the counter reaches 10 it will return FALSE and follow the grey arrow. This path will report a failure and disconnect the line, as shown in Figure 10

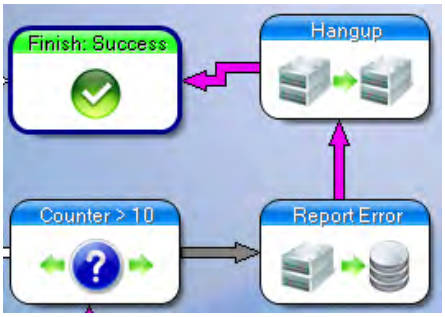

Figure 10 - Report Error that Modem Could not be Reached

The reporting of the error is done in the same table that we use for writing the data from the Remote Telemetry Unit. In the first step, we are writing the quality value, date/time value, and the value of the tag. This block is configured like the one shown below.

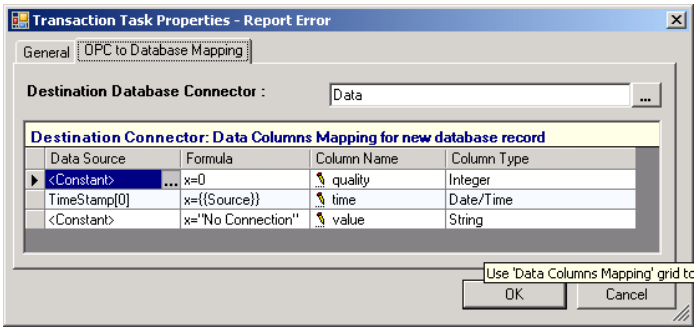

Figure 11 - Error Reporting Block

The Hang Up block uses the OPC tag to hang up the connection. This block is configured like the one shown in Figure 12

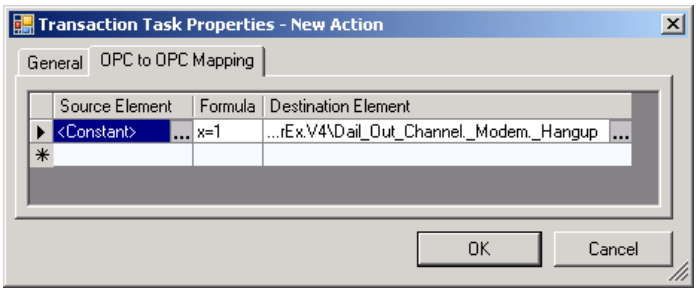

Figure 12 - Hang up Block Configuration

We have created a delay loop with a counter to protect it. Now let's continue with the Check Line Status block. Let's assume the expression is a connection and the Check Line Status will return TRUE.

Step 4 begins after the connection is made, and we will wait for 10 seconds before collecting the data. This is to ensure that the OPC tags contain fresh device data. The waiting is done with an Execution Delay block.

Step 5 is where we actually write the data to the database. The **OPC to Database** block is used for this. This block collects the OPC value of the RTU and writes this value to a new value in the database. Configure the OPC to database as shown in Figure 13.

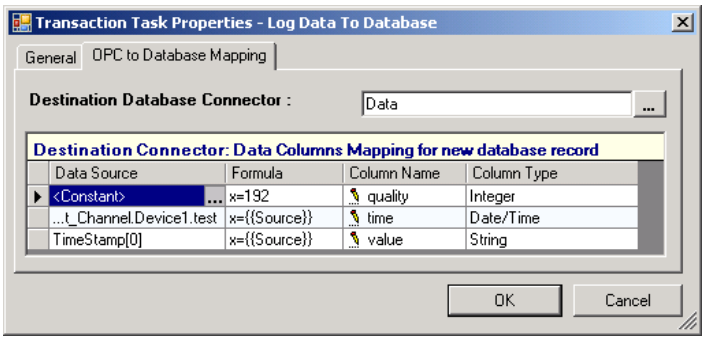

Figure 13 - Log Data to Database block Configuration

For **Step 6** we can disconnect from the Remote Telemetry Unit. For this we will reuse the existing Hang Up block from Figure 12. The Transaction Diagram should now look something the one shown in Figure 14.

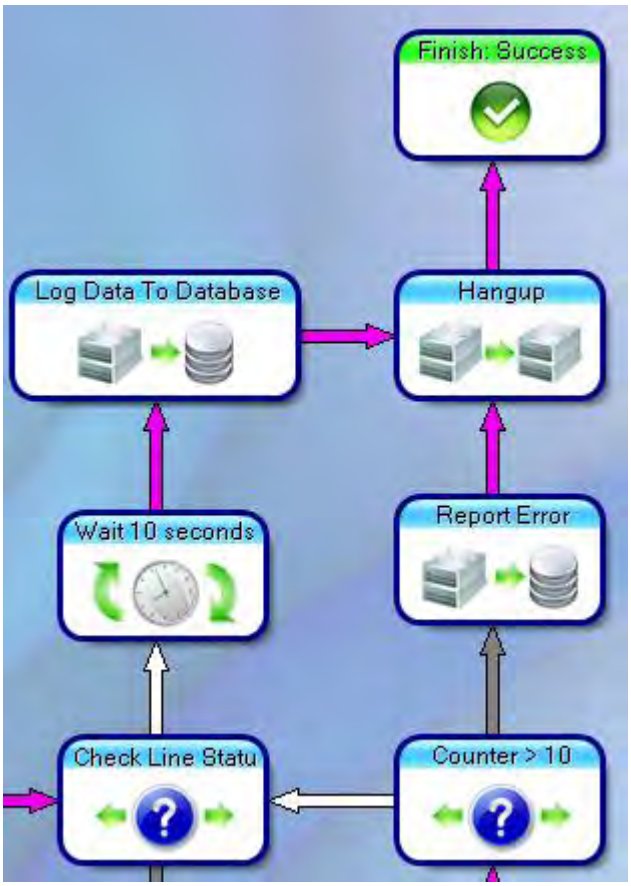

Figure 14 - Transaction Flow Logic

The Transaction is now ready. Close the Transaction Diagram and save the configuration.

Create a new Transaction and use the Transaction Diagram in the configuration. Check "Enable" and click on "Execute Now" to test the BridgeWorX Transaction.

NOTE: If you need further information on how to create a transaction, please refer to the *BridgeWorX – Quick Start* Application Note.