KEPServerEX Client Connectivity Guide

For GE's Cimplicity

KTSM-00003
v. 1.03

KEPWARE END USER LICENSE AGREEMENT AND LIMITED WARRANTY

The software accompanying this license agreement (the Software) is the property of Kepware Technologies, and is protected by United States and International Copyright laws and International treaty provisions. No ownership rights are granted by this Agreement or possession of the Software. Therefore, you must treat the Licensed Software like any other copyrighted material (e.g., a book or musical recording), except that you may make a single copy for backup or archival purposes. Your rights and obligations in its use are described as follows:

1. You may use and display this software on a single computer.
2. You may make one copy of the software for archival purposes or you may copy the software onto your hard disk and hold the original for archival purposes.
3. You may not modify or attempt to reverse engineer the software, or make any attempt to change or even examine the source code of the software.
4. You may transfer the software to another computer using the utilities provided. However, the software must be used on only a single computer at one time.
5. You may not give or distribute copies of the software or written materials associated with the software to others.
6. You may not sub-license, sell, or lease the software to any person or business.

Return Policy
The original licensee of the software can return it within sixty (60) days of purchase. Please call us for a Return Material Authorization Number.

Limited Warranty
Kepware does not warrant that the Software will be error free, that it will satisfy your planned applications or that all defects in the Software can be corrected. If Kepware provides information or assistance regarding the use of the Software or otherwise, Kepware is not assuming the role of engineering consultant. Kepware disclaims responsibility for any errors or omissions arising in connection with engineering in which it's Software or such information or assistance is used.

The foregoing is the sole and exclusive warranty offered by Kepware.
Kepware disclaims all other warranties, express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose, with regard to the licensed software and all accompanying materials.

In no event shall Kepware be liable for incidental or consequential damages, including lost profit, lost savings, lost opportunities, or other incidental or consequential damages arising out of the use or inability to use the licensed software, even if Kepware has been advised of the possibility of such damages.

Kepware's entire liability shall be, at Kepware's option, either (a) return of the price paid for the Software (or component), or (b) repair or replacement of the Software (or component) that does not meet Kepware's Limited Warranty and which is returned to Kepware within the warranty period. This shall be the sole and exclusive obligation of Kepware and your sole and exclusive remedy with respect to any such failure. The Limited Warranty is void if failure of the Software (or component) has resulted from accident, abuse or misapplication.

Support
Kepware provides *unlimited e-mail* support for all Software whether a demo or registered license. Kepware will provide a total of *two hours* free phone support for all registered Software after paying the applicable license fees. Kepware will provide *unlimited phone* support so long as you have paid Kepware any applicable maintenance or support fees and subject to the terms of those agreements. All corrections and maintenance releases will be made available through Kepware's Internet site. All major product releases of the Software are subject to upgrade fees. At no time will on-site support be provided without advance payment to Kepware for a minimum of two days on-site engineering support services, plus all expenses.

Trademarks
Microsoft® and Microsoft Excel® are registered trademarks, Windows™ is a trademark of Microsoft Corporation.

32 Bit KEPServerEX Connectivity Guide

Kepware Technologies
P.O. Box 579
Portland, Maine 04112
Sales: (207) 775-1660 x208
Technical Support: (207) 775-1660 x211
Fax: (207) 775-1799
E-mail: sales@kepware.com or
technical.support@kepware.com
www.Kepware.com

Table of Contents

Introduction to KEPServerEX

KEPServerEX is a 32 bit windows application that provides a means of bringing data and information from a wide range of industrial devices and systems into client applications on your windows PC. KEPServerEX falls under the category of a "Server" application. It is very common to hear the term "client/server application" in use across many software disciplines and business segments. In the industrial market, it has usually come to mean the sharing of manufacturing or production data between a variety of applications ranging from human machine interface software and data historians, to large MES and ERP applications.

Regardless of the business segment served, client/server applications have one thing in common: a standardized method of sharing data. In the industrial segment, many client/server technologies have been developed over the last ten years. Initially, some of these technologies were proprietary. In many cases these proprietary client/server architectures were in wide use but remained unavailable to third party applications. Early in the development of windows, Microsoft provided a generic client/server technology called DDE or Dynamic Data Exchange. DDE did provide a basic architecture that would allow many windows applications from a wide range of vendors to share data, but there was one problem. DDE was not designed for the industrial market. It lacked much of the speed and robustness desired in an industrial setting. However, this did not stop DDE from becoming a dominant client/server architecture, largely due to its availability in most windows applications. In time, variations on Microsoft's DDE were developed by some of the leading vendors in the market. These variations addressed some of the speed and reliability issues of DDE but many people in the industrial segment agreed that a better system needed to be developed.

With the advent of 32 bit Operating Systems, and the use of Ethernet to provide communications between devices, there was a need for quicker and cleaner data transfer between software applications. This is where OPC saw its birth into the industry.

OPC (OLE for Process and Control) servers provide a standardized method of allowing multiple industrial applications to share data in a quick and robust manner. The OPC server provided in this package has been designed to meet the demanding requirements found in the industrial environment.

This OPC server has been designed as a two-part program. The primary component provides all of the OPC and DDE connectivity as well as the user interface functions. The second part is comprised of plug-in communications drivers. This two-part design allows you to add multiple communications options to your SCADA application while utilizing a single OPC server product thus reducing your learning curve as your project grows.

OPC technology reflects the move from closed proprietary solutions to open architectures that provide more cost-effective solutions based on established standards.

Accessing KEPServerEX

A Windows based client application must be used to view data from the KEPServerEX application. In this section we will cover the basics of connecting a number of common OPC clients to KEPServerEX. While we cannot possibly cover every client application that exists, we believe that after reviewing this document you should be able to deal with most client applications.

The intention of this section is to show connectivity to KEPServerEX. It is assumed that you have already either configured your KEPServerEX application by selecting the appropriate driver and settings or you have run the Simulator demo (Simdemo.opf) which is included with KEPServerEX. For simplicity, the Simdemo project will be used for all examples contained in this section.

Before beginning any of the examples, start the KEPServerEX application by selecting it from your Start Menu or from its desktop icon. Once the server is loaded, use the File|Open command to load the "Simdemo" project. The KEPServerEX application is always active once you have opened an existing project or configured at least one channel and device in a new project. After you have selected a project, in this case the Simdemo project, KEPServerEX will automatically load this project when an OPC client application invokes KEPServerEX's OPC server component.

Users have always had the ability to create what we refer to as "user defined tags" in their KEPServerEX application. Prior to OPC, defined tags gave a DDE application designer the ability to create a label for device data. Assume register 1000 contained the value of parts made, without defined tags a DDE application would have directly accessed register 1000. Using defined tags a label can be created like "PartsMade". Now the DDE application could access the data via this new label, removing the machine level knowledge from the client application and keeping it at the server level where it belongs. This label, while useful for DDE is a necessity for OPC clients. For OPC clients, defined tags take on a greater role. Like the DDE example, defined tags allow you to create labels for your device data and keep the configuration of those tags in the server. OPC clients have a major advantage over DDE clients. OPC clients can browse the defined tags you create in your KEPServerEX application, which allows you to simply point and click on a tag to add it to your OPC client project.

OPC Tag Browsing allows you to see a list of the defined tags you have created in your KepserverEx application, directly within your OPC client application.

For more information on defined tags see the "Designing a Project" section of the KEPServerEX help file, which can be accessed from the Help/Contents menu selection of the KEPServerEX application.

Using KEPServerEX Drivers

Part of the innovative design of Kepware's OPC/DDE Server Technology is the separation of the Hardware Protocol Driver from the Server Technology. This separation allows the user to use one or more drivers in the server at the same time. Each driver has its own help file which provides information on devices supported, communications parameters, cabling, addressing, and error messages.

The driver help file should contain all of the information you will need to connect your device to the PC so that the we can talk to it. If you do not connect to the device be sure to check the error messages and look up their meaning in the help file.

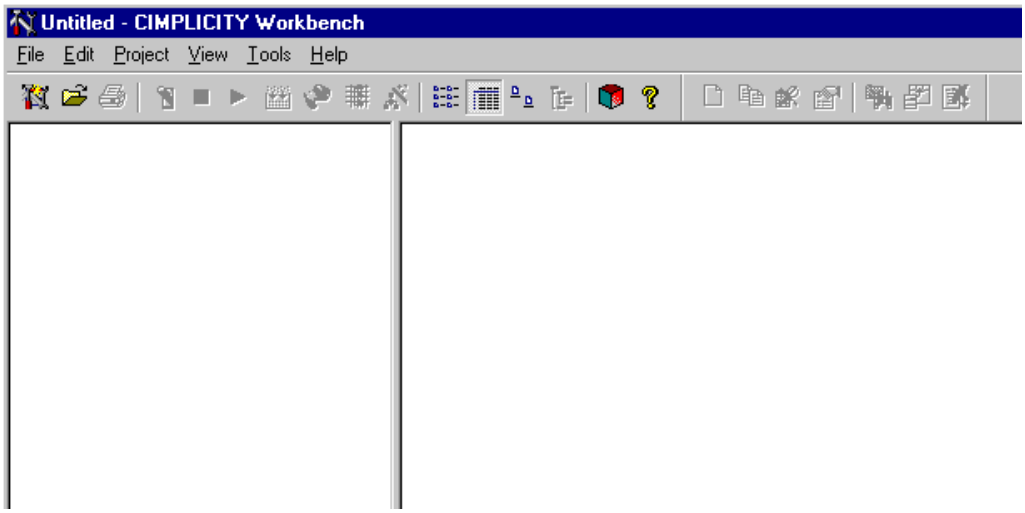
GE's Cimplicity® as an OPC Client

Connect to the Server from Cimplicity V5.0

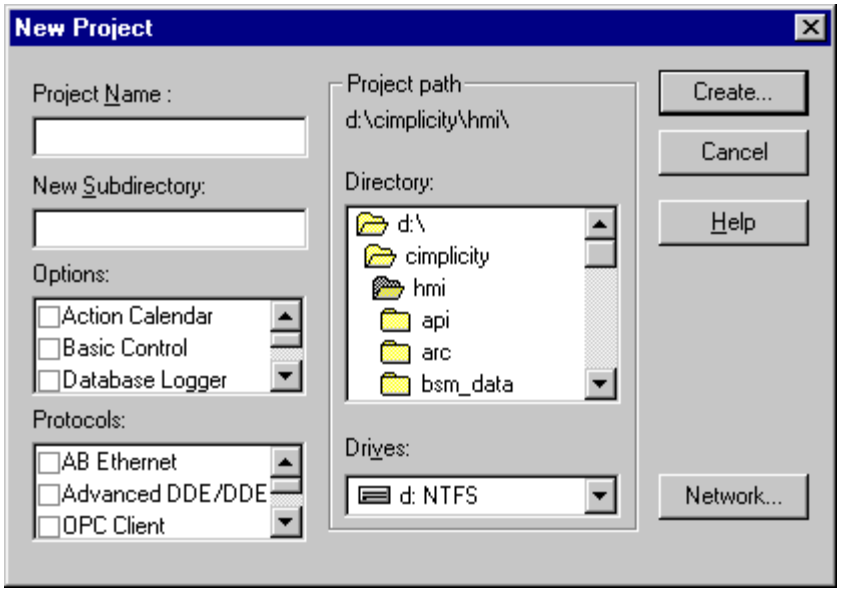
The following steps will show you how to create an OPC connection to the Server from Cimplicity V5.0. Cimplicity's Workbench is the project control center.

Create a New Project

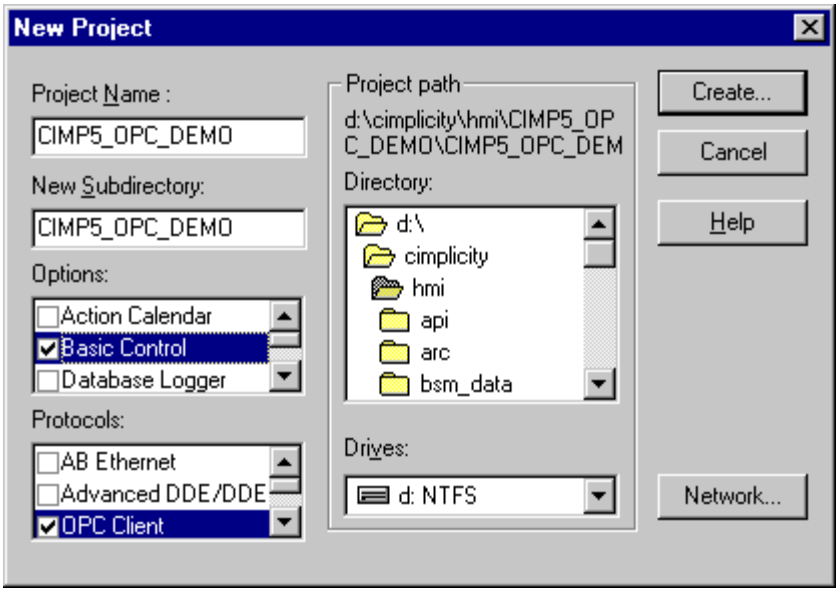
1. Start Workbench and click on the New Project button  or click File|New|Project on the Main Menu.



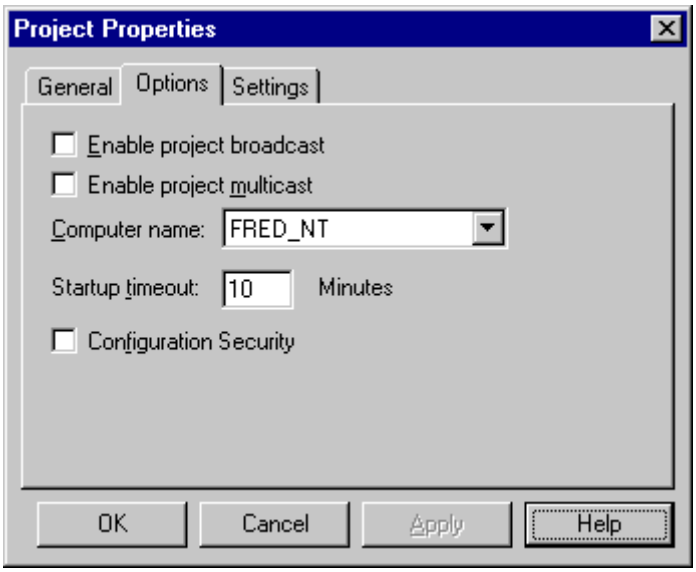
2. The New Project Dialog box will appear.



3. First enter a Project Name for your new project. By default the Sub Directory is auto filled with the Project Name
4. Next select the Options that you wish to include in your project. For our example we chose only Basic Controls.
5. Then select OPC Client as the Protocol you want to use in your project



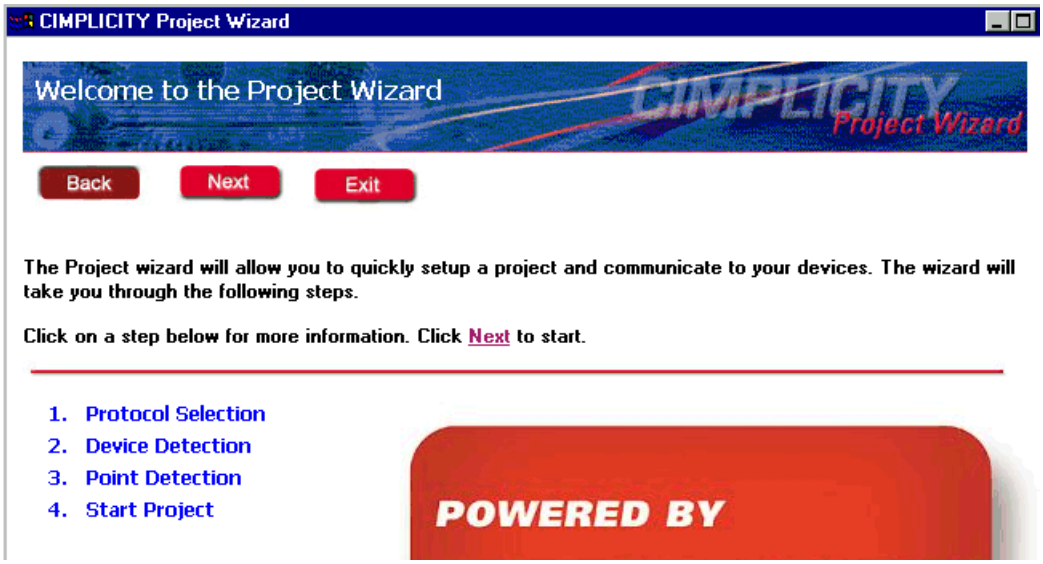
6. Lastly, select the Project Path and click Create… to create your project.



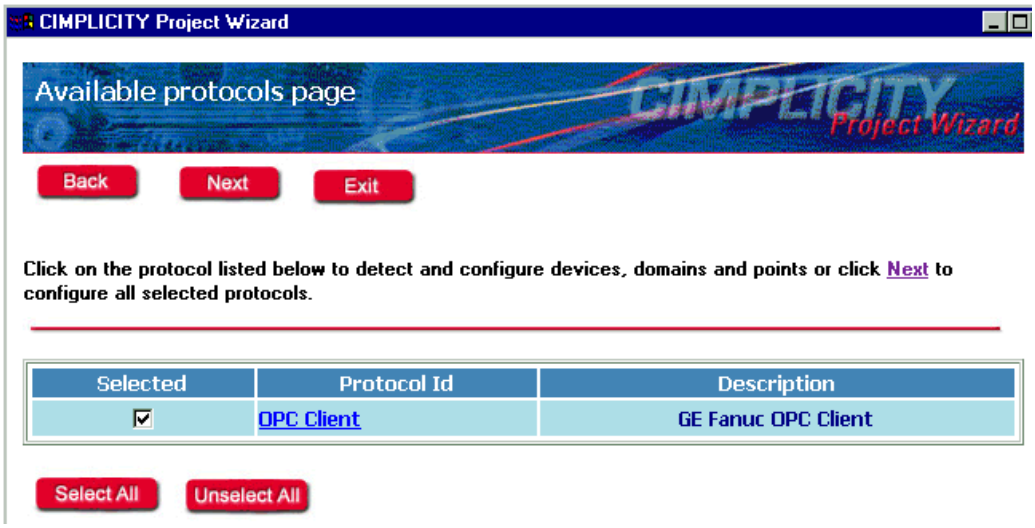
7. The Project Properties dialog box will open. If you plan to provide information from the project to the rest of your network then you will want to check Enable Project Broadcast, otherwise click OK to accept the Project Properties.

Run the Project Wizard

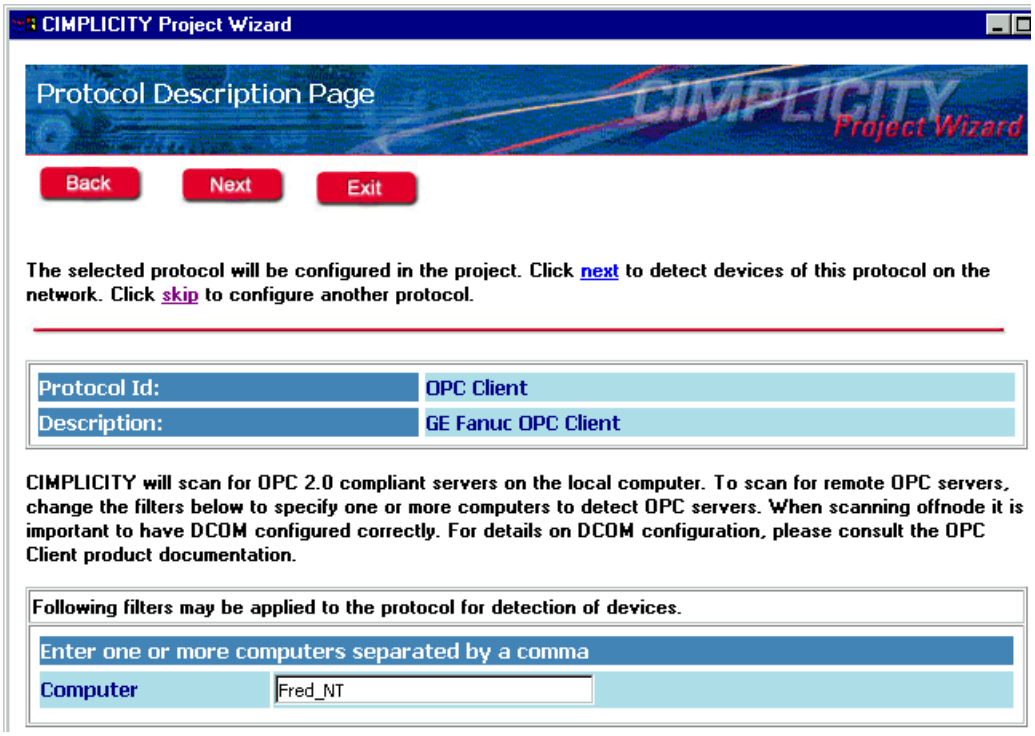
8. When you create a new project that has OPC Client protocol enabled the Project Wizard should start automatically. To start the wizard, manually click on Project|ProjectWizard in the Main Menu.



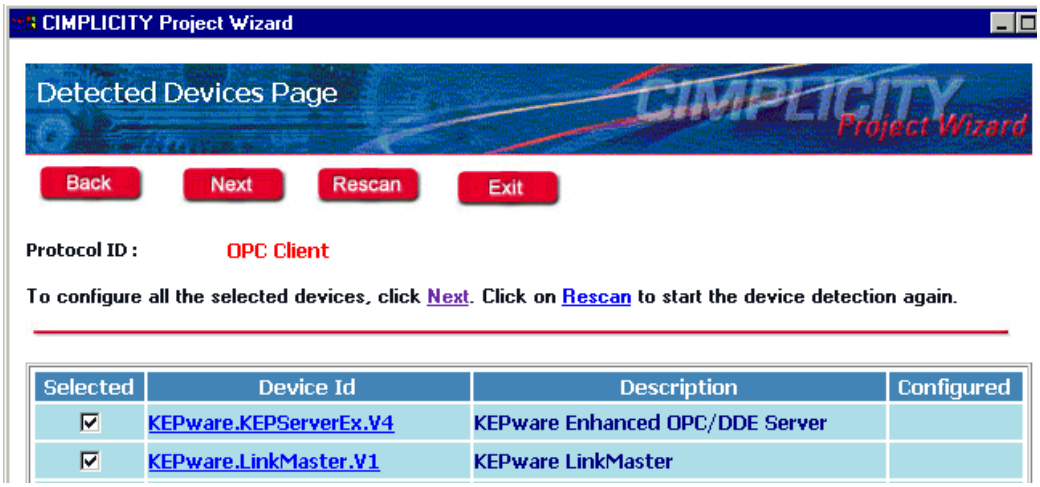
9. Click the Next button  to start the first step of the wizard.



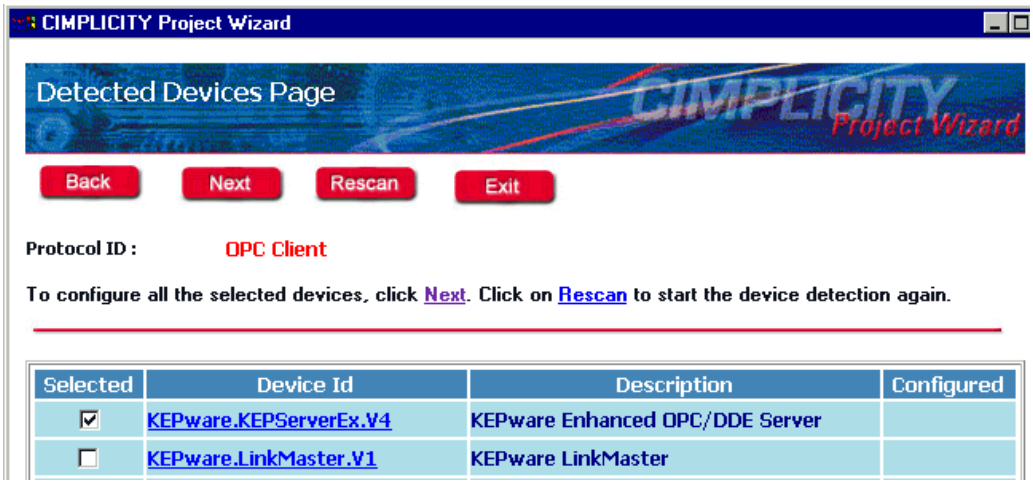
10. When we created the project we selected the single protocol of OPC Client which is the protocol we now wish to configure.
11. Click the Next button  to configure the selected protocol.



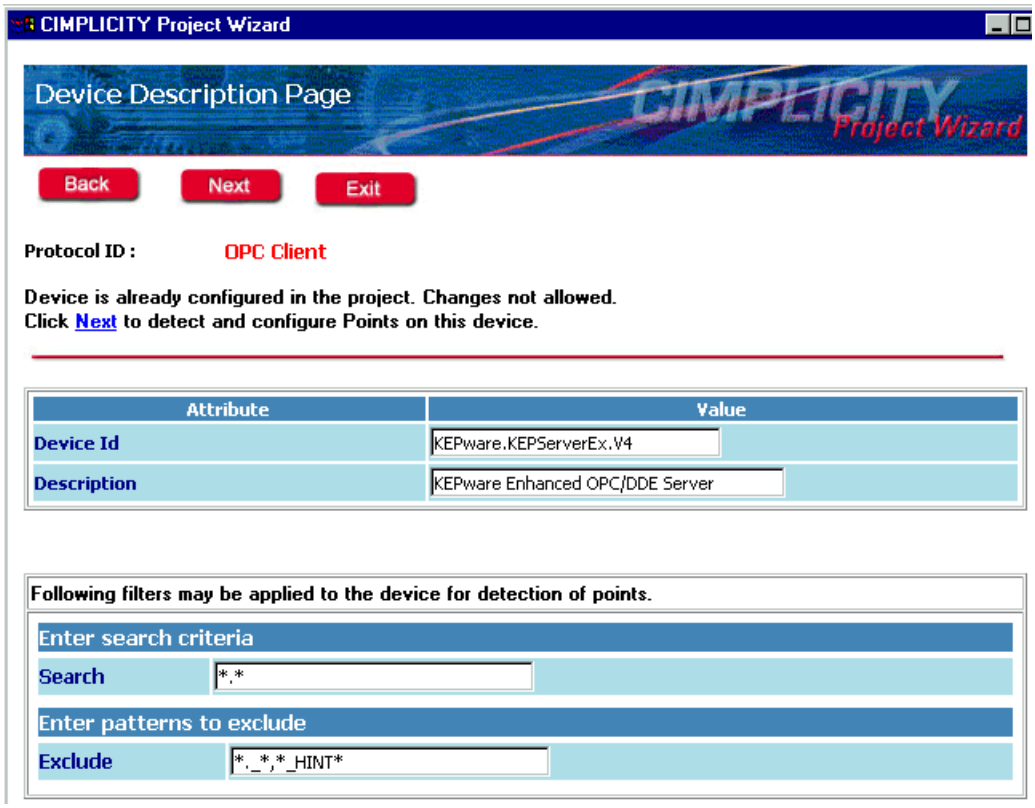
12. Verify that you have selected the correct protocol. You should also see at least your computer named in the list of computers to be searched for valid devices. By default the wizard sets its server browse filter to look on the local PC only for servers. If you want to look on another PC you would replace the local PC's name with that of the remote one. You could also browse for servers on multiple PC's by entering each name separated by a comma. If you doing a remote connection to the server you will need to configure DCOM. See Kepware's DCOM Configuration for KEPServerEX guide for details.
13. Click the Next button  to detect valid devices.



14. You will notice that the Project Wizard has detected all of the devices (servers) that are installed and available to it
15. At this point we are only concerned with connecting to and configuring KEPServerEX (Kepware Enhanced OPC/DDE Server). Uncheck all of the other devices except it.

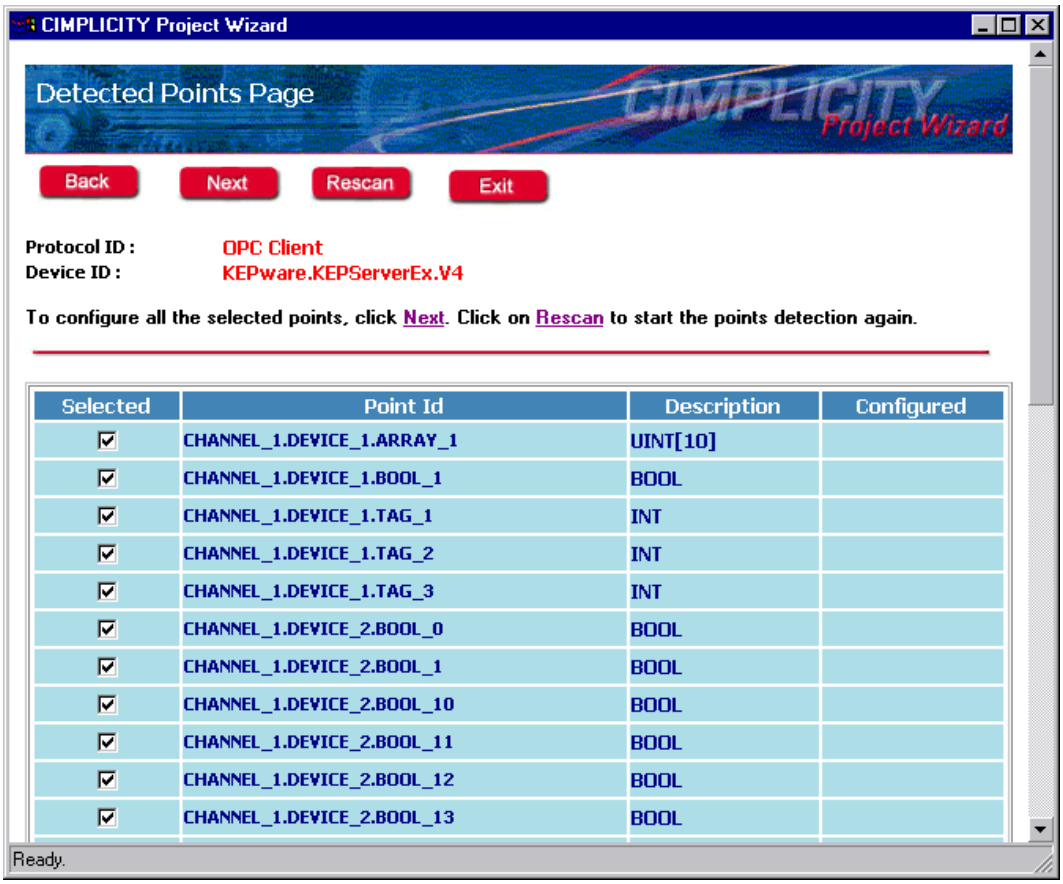


16. Click the Next button  to show the device details.



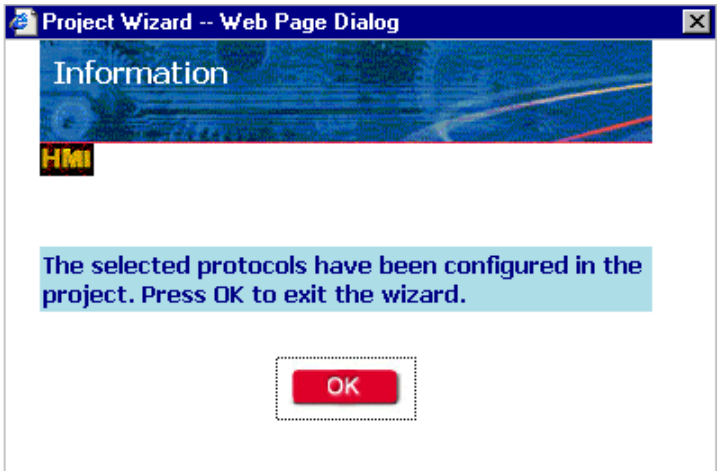
17. Verify that you have indeed selected the correct server. Next enter or edit the search for the points in the device. You may also wish to exclude certain points or groups of points. The Project Wizard was tested with KEPServerEX so you can see that by default, the wizard excludes *System tags* and *Hints*.

18. Click the Next button  to detect the points in the server that meet the search criteria and exclusions.



At this point you should see all of the points configured in the server that meet your search criteria. If you do not see all of your tags, then may want to go back one step and re-evaluate your search and exclusion criteria, then select the rescan option to browse the server again. If you add new tags to the server you will want to restart the Project Wizard and rescan the server.

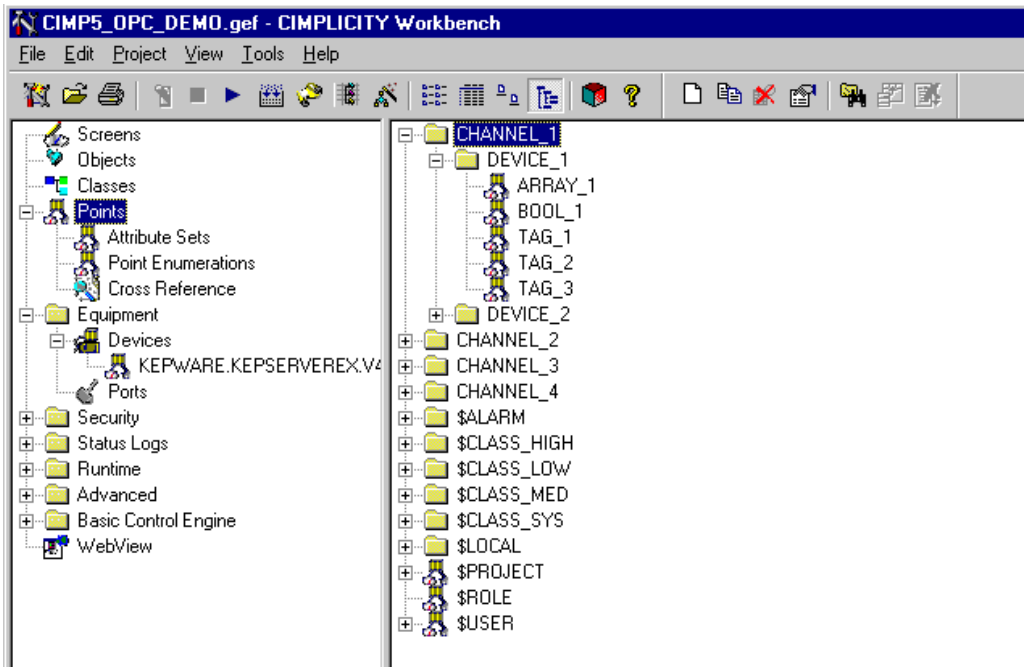
19. By default all of the points will be checked or selected. At this time you may uncheck any points that you do not wish to configure
20. Click the Next button  to configure the selected points



21. Once the Project Wizard has extracted the point data from the server and added it to your project you will see the completion window.

22. Click on the OK button  to close the Project Wizard
23. In the project select Points in the left-hand pane of the project window and expand the folders under Channel_1 in the right hand window to display the points that were added.

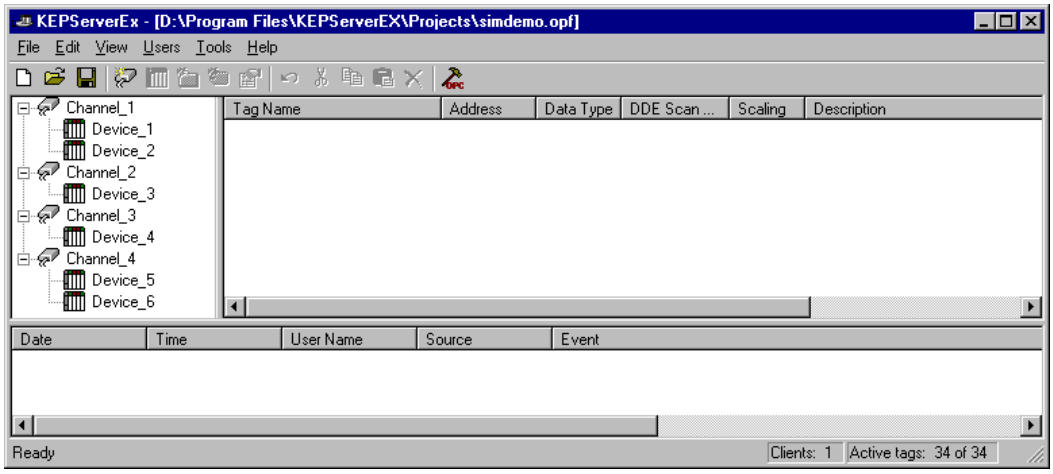
This is the Tree view representation of the “Point ID”. If you were to look at the point “Array_1” in the Detail view instead of the Tree view the point would have a Point ID of “Channel_1.Device_1.Array_1”.



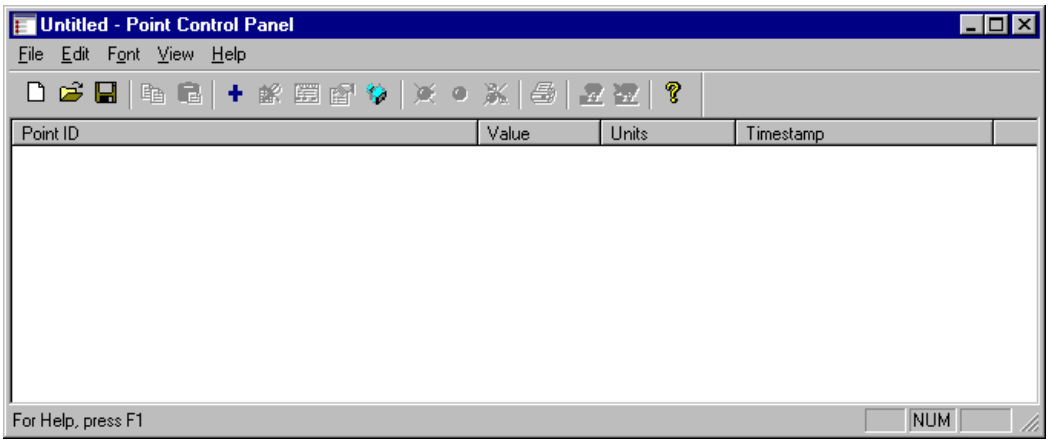
Running the Project and Connecting to KEPServerEX

Now is a good time to verify that you can poll data through KEPServerEX.

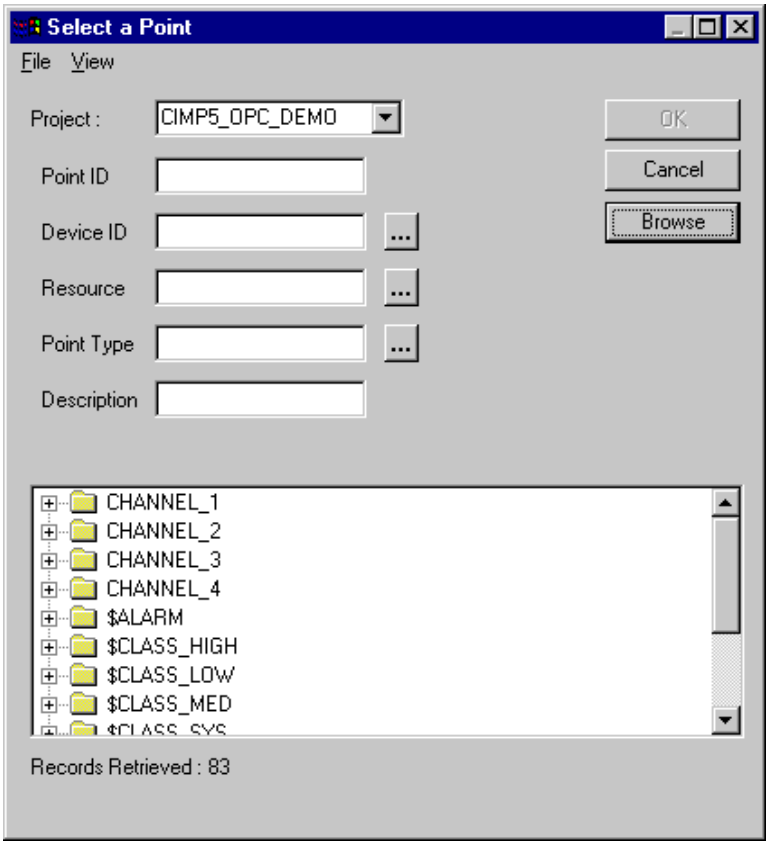
1. Click on the Run button  to start the Cimplicity project
2. Once your project is running, look at KEPServerEX and check the Status Bar at the bottom right edge of the server. You should see at least one Client and 1 of 1 Active tags. In our example we have 34 Active tags out of 34 because we selected all of the available tags in our project wizard. If the Connection Status bar is blank, then Cimplicity is not connected to the server.



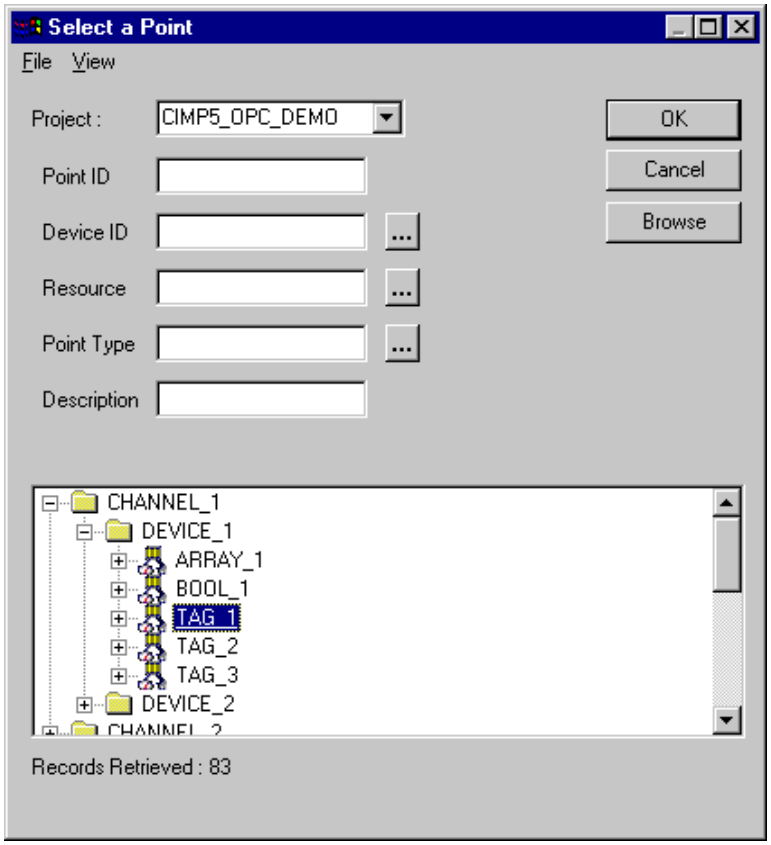
3. Back in your Cimplicity project click on the “+” sign to the left of Runtime in the project tree and double click on Point Control Panel.



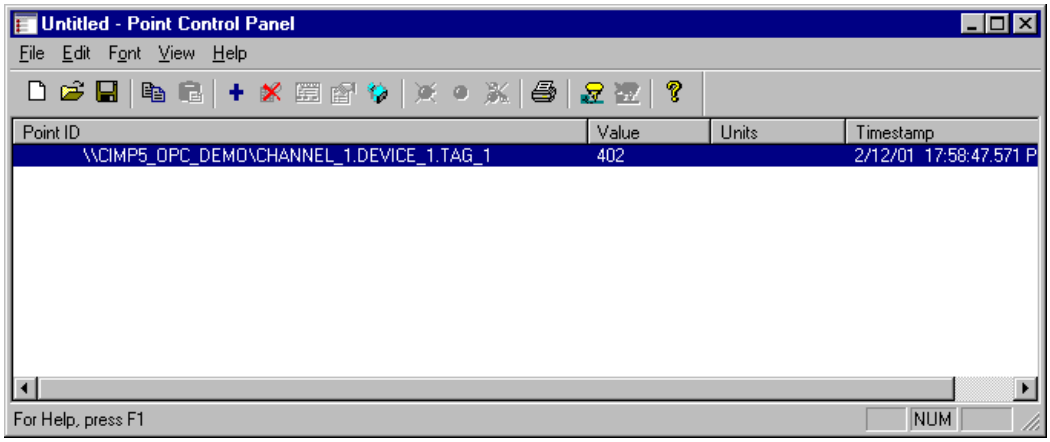
4. Unless you have already done so you will probably be prompted to enter a login before you are allowed to browse the points database and select the point or points you wish to view.



5. In the Tree view of the Point Selection dialog box expand the branches of the tree and select the point or points you want to view and then click OK. As you can see we have chosen point “Tag_1”.



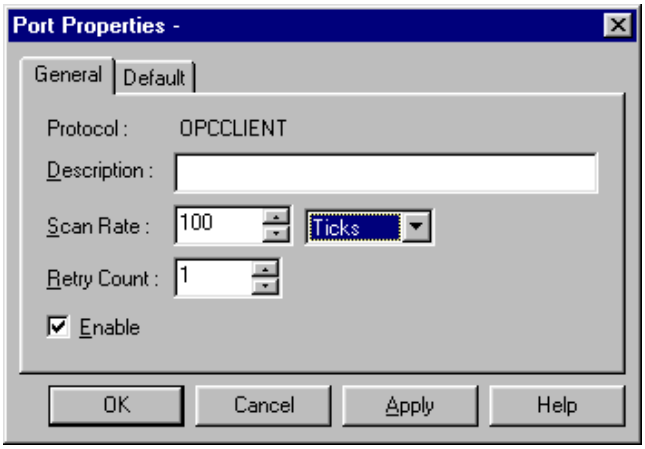
You should now see the point displayed in the panel. In this example we used our sample simulation project called "Simdemo" which simulates changing data for TAG_1. If you are also using the Simdemo project and you are viewing TAG_1, you should see its value ramping up.



You have probably noticed that the point is incrementing once per second. That is because the simulator we are using increments the data values every time they are scanned by a client application. In the next section we will explain how to increase the scan rate (group update rate) in your Cimplicity project.

Optimizing Your Cimplicity 5.0 Project

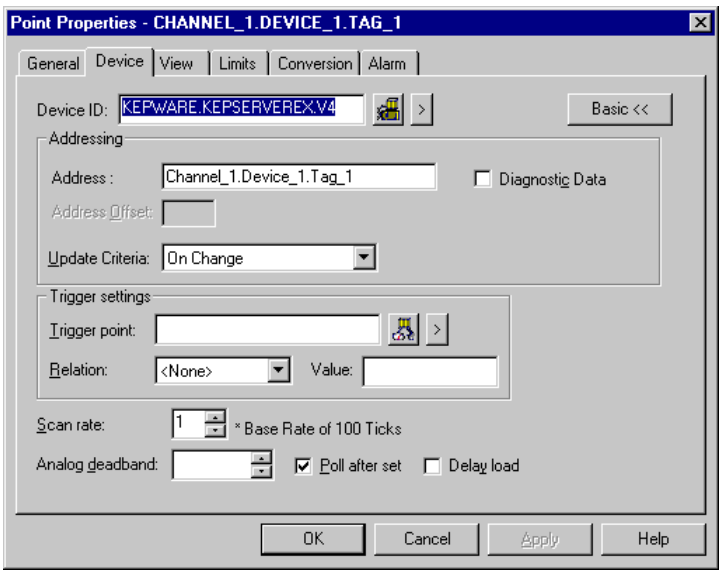
In Cimplicity 4.01 you would set the update and scan rate in the Port Properties by setting the Scan Rate to a certain number of Ticks. Each Tick is approximately 10 msec so in our example the server is scanned and this project is updated at 1000 msec.



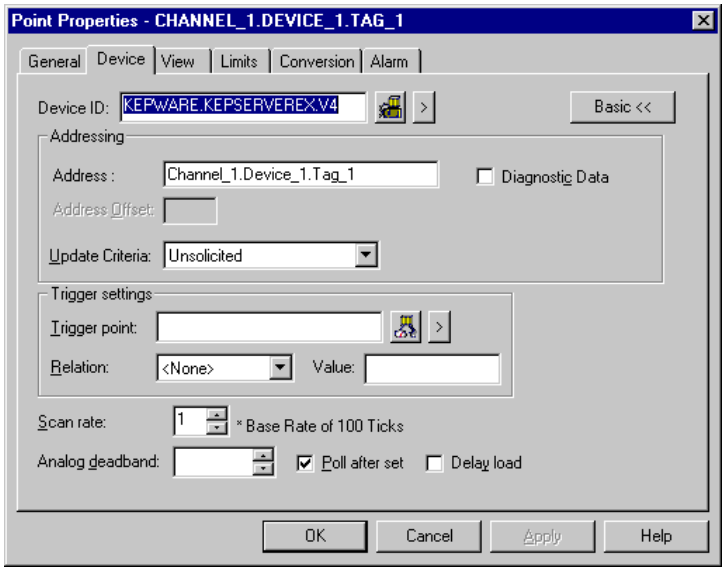
Scan rates are handled a bit differently in Cimplicity 5.0. Cimplicity 5.0 now uses an INI file for each port that allows you to establish the rate at which data is scanned

To explain this a bit we will need to step back for a moment, to when we added the points to our project with the Project Wizard. When the wizard adds the points it uses a default Update Criteria of "On_Change". This Update Criteria instructs your project to use Synchronous read/write operations with the server

Synchronous Read/Write – The client sends a read or write request to the server and waits for a response before making the next request. If your application reads and writes large amounts of data, the use of Synchronous operations could significantly hinder your application's performance..



We strongly recommend changing the update criteria to "Unsolicited" which will instruct your project to use Asynchronous read/write operations with the server.



Asynchronous Read/Write – The client sends a read or write request to the server and continues processing. The server processes the request and upon confirmation from the device that it was successful, it notifies the client via the Callback() function.

Note: In the KepserverEx writes are executed as soon as they are received from the client.

Ok, back to the update rate that you set in the Port Properties. That is the rate at which the points are updated from the servers cache when using the "On Change" update criteria. The Scan rate is controlled from an INI configuration file created for each port. This file can be found in the data folder located within the project folder. If you have a port named Master_OPC_0 then you should find a file named "Master_OPC_0.ini".

; This INI file is for use by the OPC Client devcom provided with CIMPLICITY 4.0.08 or higher, ; ; and CIMPLICITY 5.0.1
[PortLevel]
EightByteReals=0
UseServerTimeStamp=1
MessageTicks=50
CircularLog=1
UseDataTypePromotion=0

[DEVICE1]
StartupDelay=0
ReadDelay=0
PingInterval=5000
PingTimeout=3000
AbortShutdown=0
UseLocalReg=0
DCOMTimeoutThreshold=10000
PingBeforePoll=1
PingBeforeWrite=1
ForceOPC1Server=0
HRBothActive=0

[DEVICE2]
StartupDelay=0
ReadDelay=0
PingInterval=5000
PingTimeout=3000
AbortShutdown=0
PingBeforePoll=1
PingBeforeWrite=1

[DEFAULTPOLL]
ScanRate=1000
NoAccessPath=0
DeviceReadAfterSet=1

[MYGROUP01]
ScanRate=1000
DeviceReadAfterSet=1
NoAccessPath=0

[**DEFAULTUNSO]**
ScanRate=1000
DeviceReadAfterSet=1
NoAccessPath=0

You will immediately notice that your file may not look like the one we have displayed here. In our example we have removed all of the comments.

When you use the "On_Change" update criteria, if you change the ScanRate value under [DEFAULTPOLL] it will increase or decrease the number of times your project requests data from the server.

Our demo project currently has an update rate of 100 Ticks (1000Msec) with update criteria of "On_Change" and a scan rate of 1000Msec. This means that we see one update for every scan. If we were to change the scan rate to 100Msec then we would update our data in the project once for every 10 scans.

If you decide to stay with this configuration then you will want to adjust the two scan rates so that you update the client more frequently

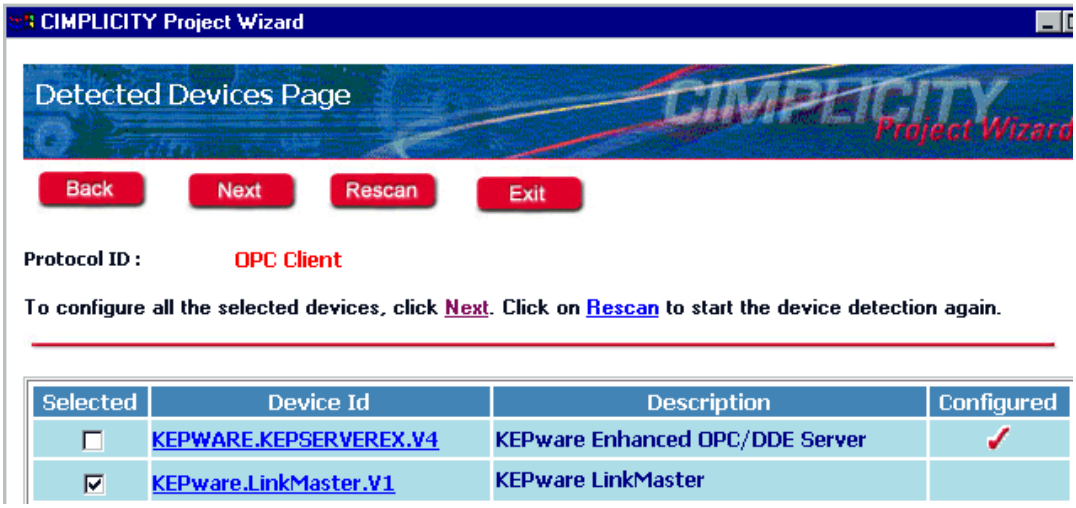
As we said earlier if you are reading large amounts of information from the server then you will want to change the tag update criteria to "Unsolicited". This update criteria uses a different set of defaults and updates as fast as the scan rate. You will want to change the ScanRate under [DEFAULTUNSO] to optimize your processing. Remember that the project is updated at the same rate that the scan occurs.

Check your KEPServerEX driver help file for additional ways to optimize the server's performance and your Cimplicity user manual for other ways to optimize Cimplicity's performance.

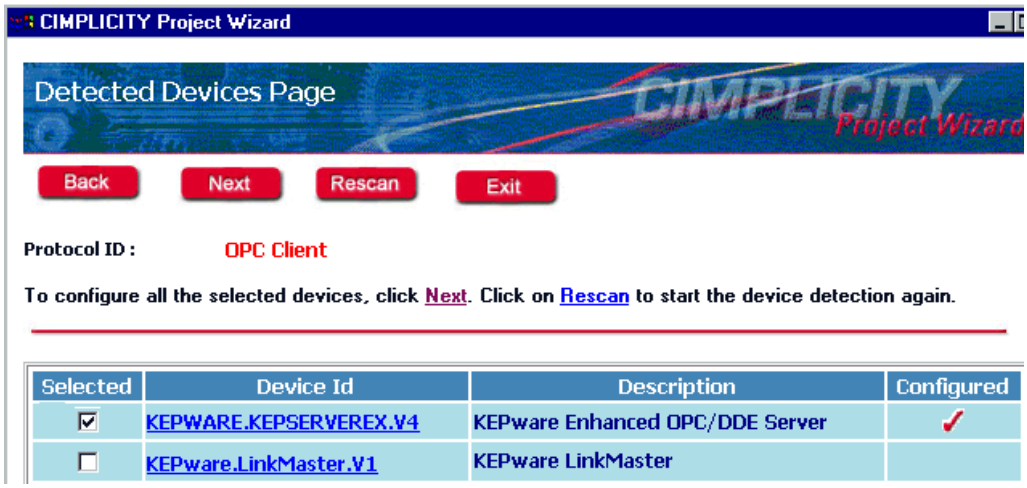
Adding Additional Points to your Project

Suppose your project has been running for several months in its current configuration but you need to add more points to it to monitor new hardware. The easiest way to do this is to use the Project Wizard to rescan the server for the new tags or, you can add them one at a time if you like

1. Click on Project|ProjectWizard in the Main Menu to start the wizard. You will notice that when you get to the Detected Devices page that it lists KEPServerEX as configured and leaves it unchecked.



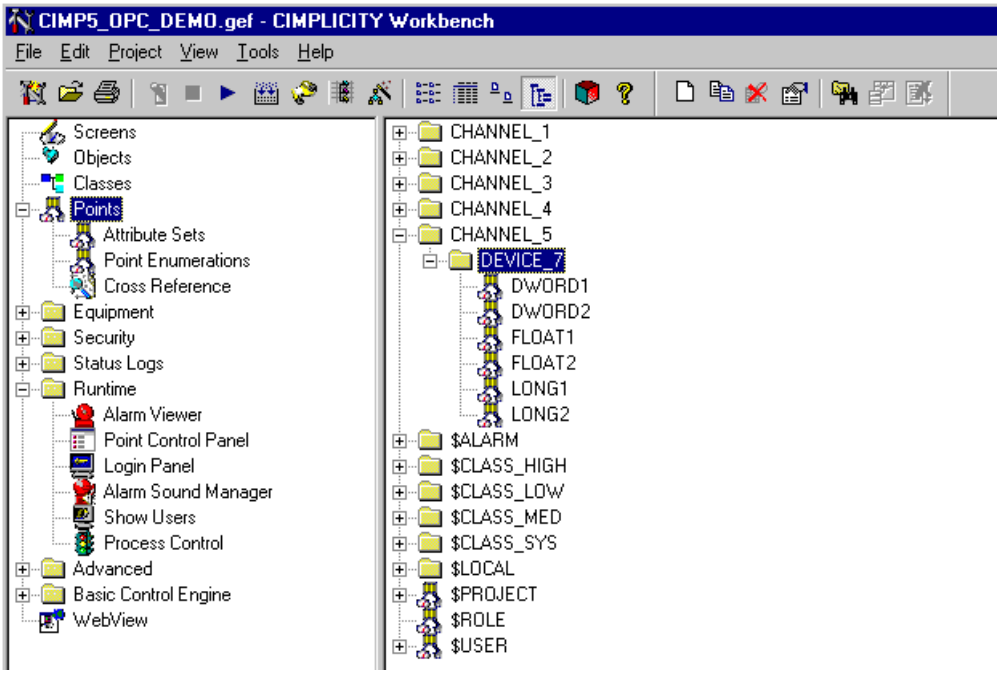
2. Deselect all the devices you do not want to configure and make sure that you re-select the KEPServerEX.



3. Click the Next button  to verify the device description and search criteria then Click the Next button  to detect the points in the server that meet the search criteria and exclusions.

4. The wizard will detect all of the points that are currently available and it will check the points that are not currently configured in you project. Uncheck the points that you do not wish to add to your project and click the Next button 

to configure the selected points.
5. Once the points have been added click OK in the completion window to close the wizard. Click View|Refresh in the main menu to refresh the project display and verify that your tags have been added.



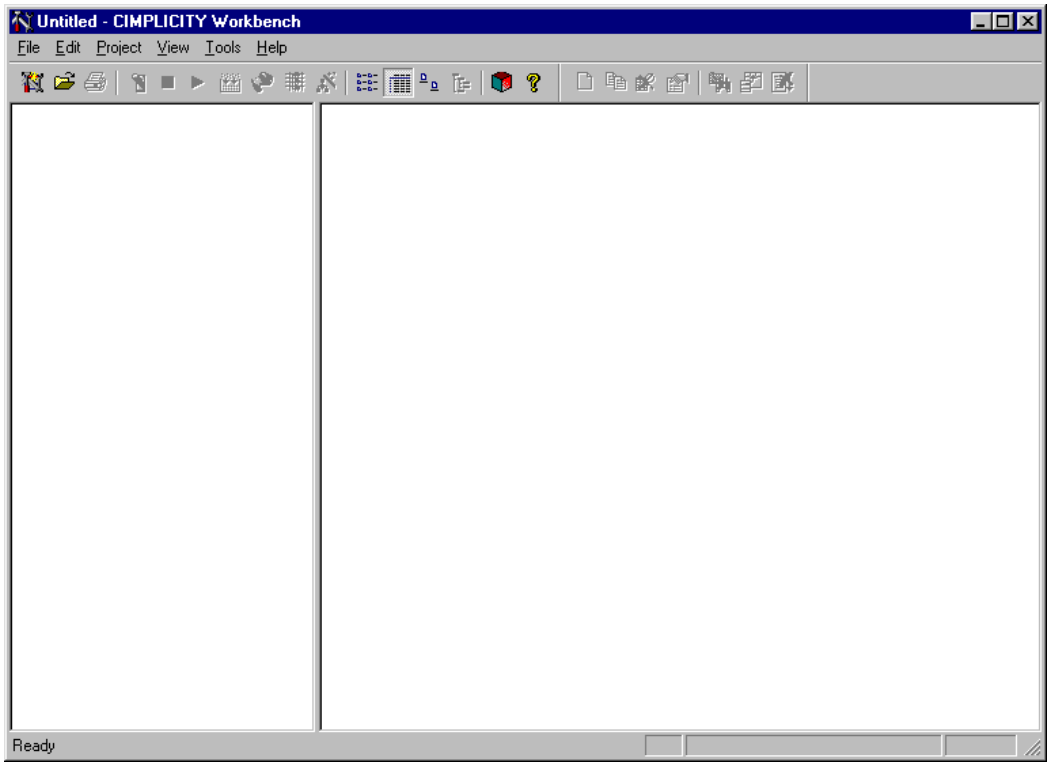
6. If you have recently upgraded from Cimplicity Version 4.01 to Version 5.0 you will be able to add tags to your project in the same way.

Connect to the Server from Cimplicity V4.01

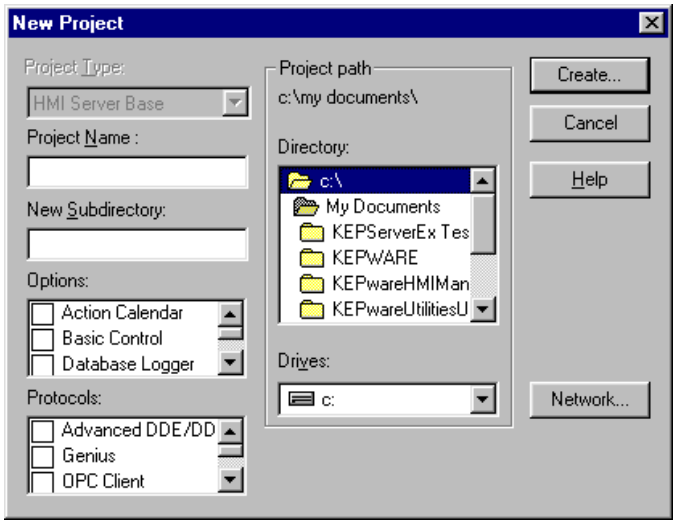
The following steps will show you how to create an OPC connection to the Server from Cimplicity V4.01. Cimplicity's Workbench is the project control center.

Create a New Project

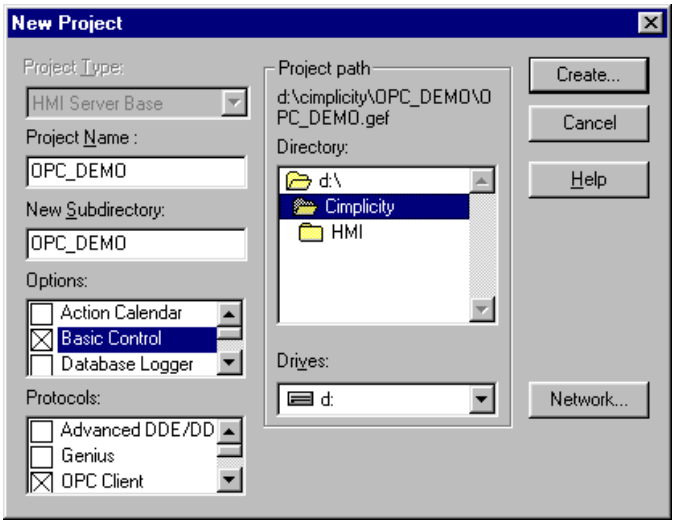
1. Start Workbench and click on New Project.



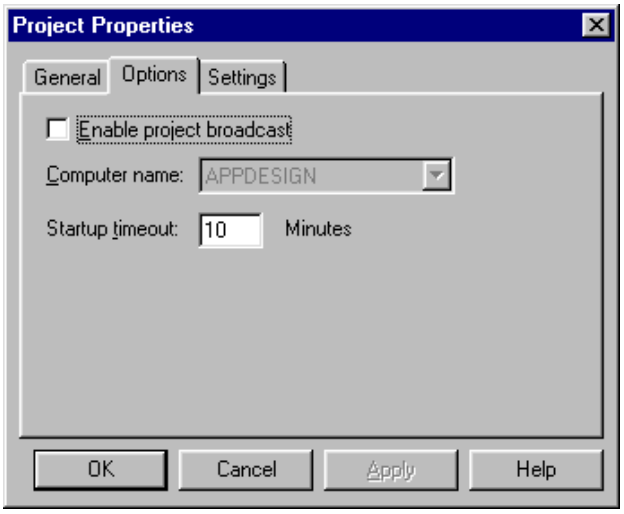
2. The New Project Dialog box swill appear.



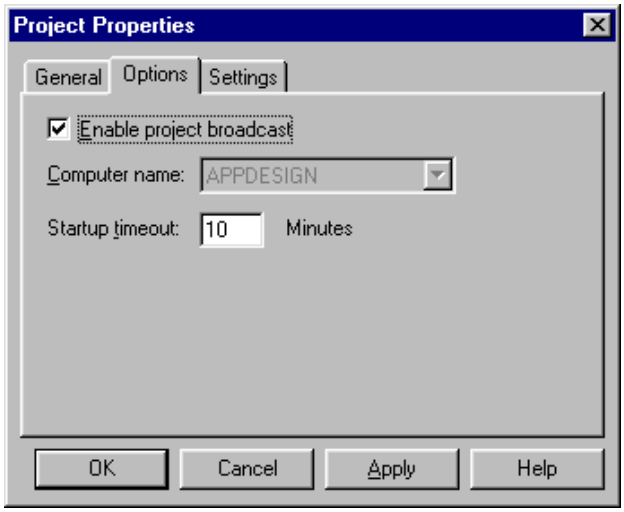
3. First enter a Project Name for your new project. By default the Sub Directory is auto filled with the Project Name.
4. Next select the Options that you wish to include in your project. For our example we chose only Basic Controls
5. Then select OPC Client as the Protocol you want to use in your project.



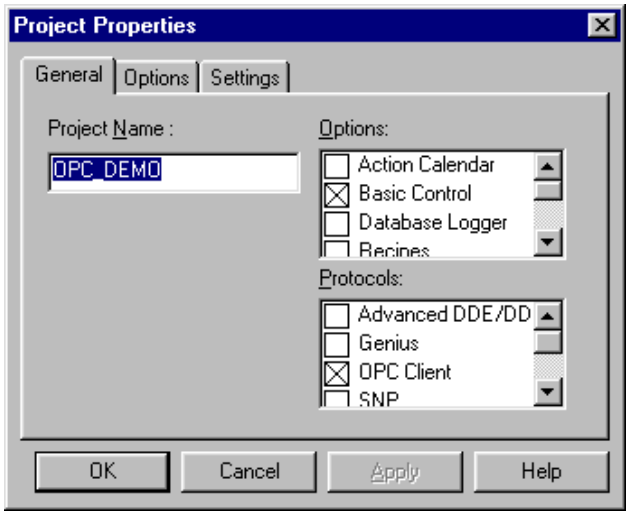
6. Lastly, select the Project Path and click Create… to create your project.



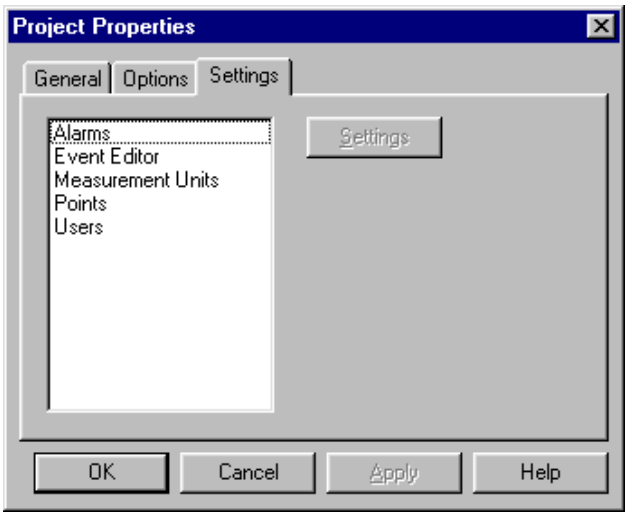
7. The Project Properties dialog box will open. If you plan to provide information from the project to the rest of your network then check Enable Project Broadcast.



8. You can modify any general project options by selecting the General Tab. In our example we accepted the defaults



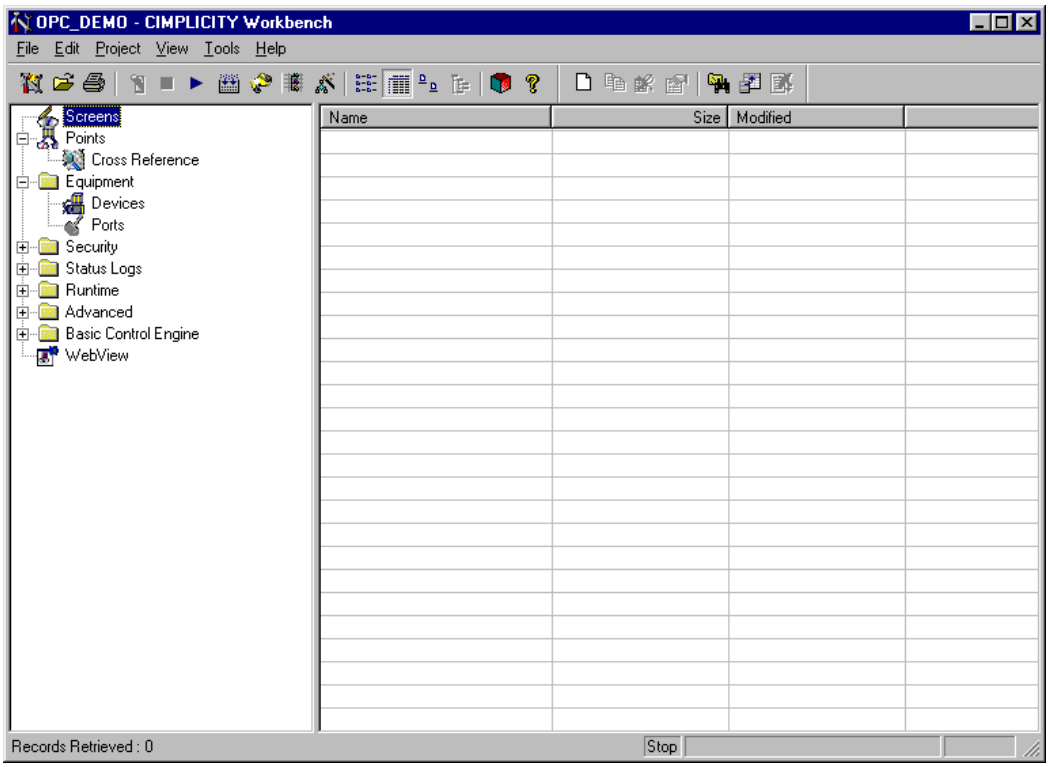
9. Selecting the Settings Tab allows you to setup specific areas of your project. In our example we accepted the defaults



10. To accept the defaults click OK

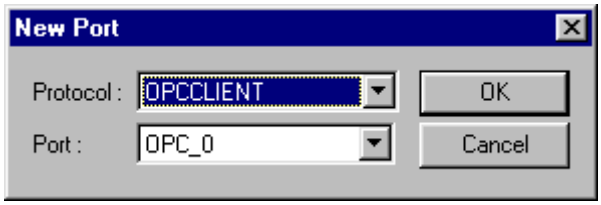
At this time the Project Wizard will open. In our example we are not using the wizard however the process is the same

11. To close the Project Wizard click Close.

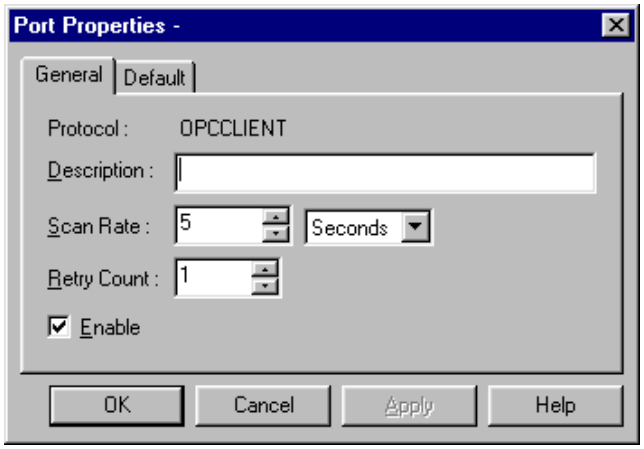


Create an OPC Port

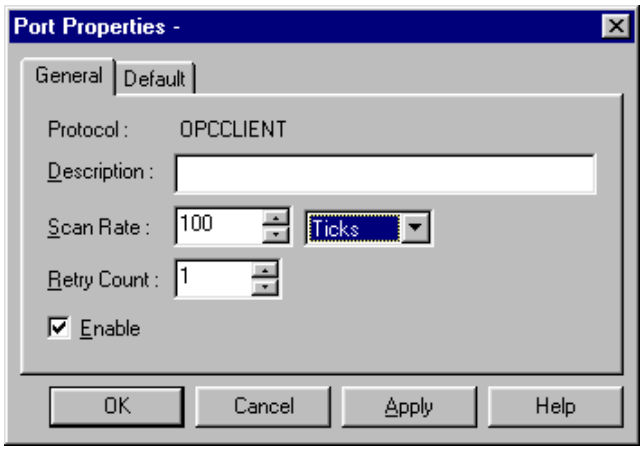
12. In the project tree view click expand Equipment then right click on Ports, click New to add a new port for the OPC connection.
13. In the New Port dialog box select the OPC port you wish to assign the client protocol to.



14. Click OK to accept

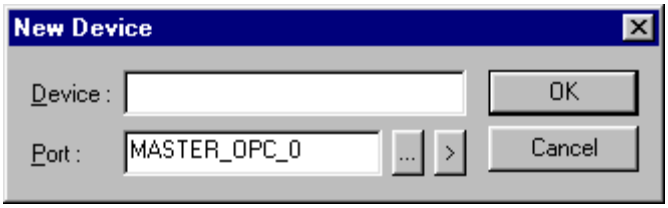


15. In the Port Properties dialog box you can enter a Description of the new port to help identify it.
16. Enter a base Scan Rate for the port.
17. Click Enable to enable the port and then click OK.

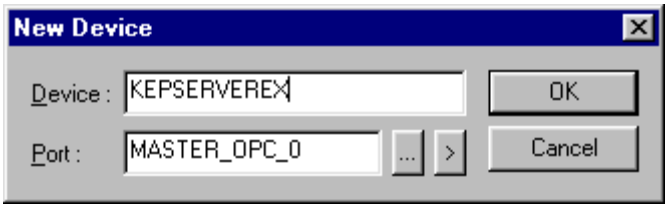


Create a Device

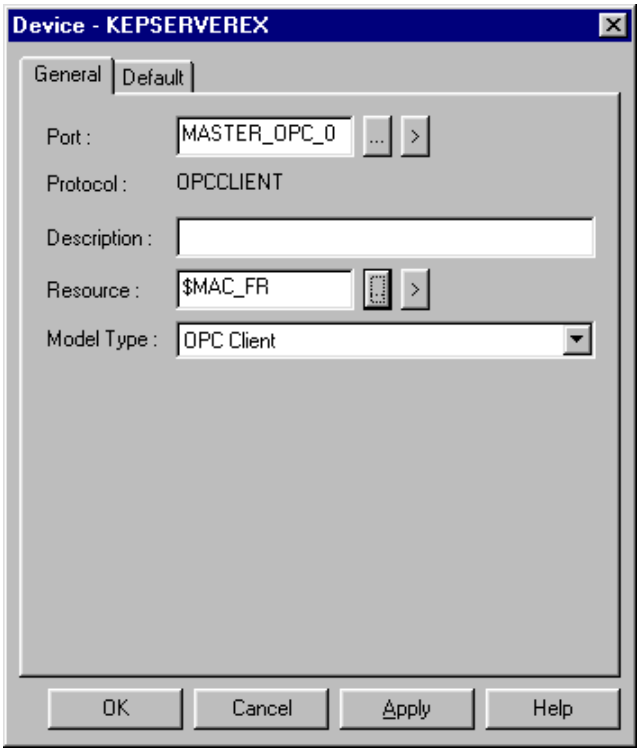
18. In the project tree view click expand Equipment then right click on Devices, click New to add a new Device for the OPC connection.



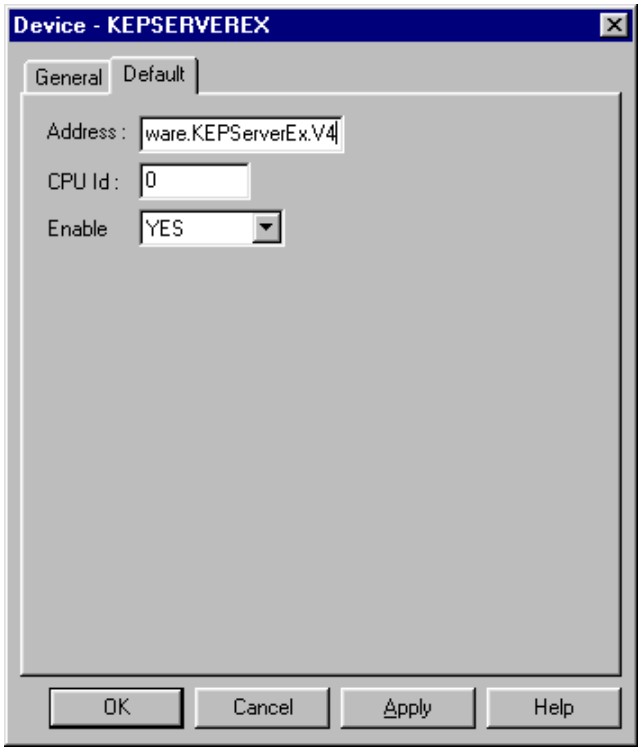
19. In the New Device dialog box enter a Device name for the new device. We chose KEPServerEX.
20. If you have created more than one Port then be sure to select the proper one for this device now and click OK.



21. In the Device Properties dialog box you may add a Description for the device.

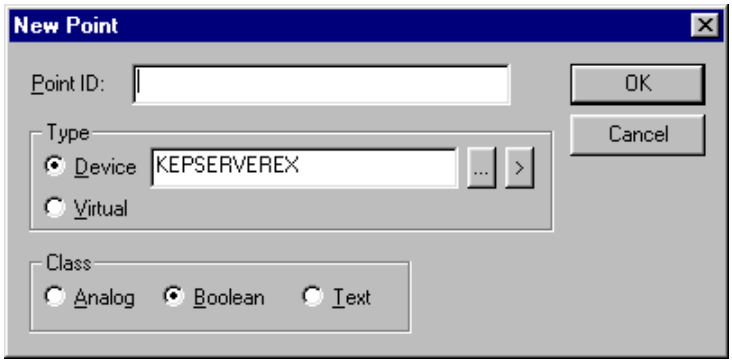


22. Next select a Resource to log device events to, then click the Default tab. See your Cimplicity user manual for details.
23. In the Default tab assign an Address to the device. You will need to enter "KEPware.KEPServerEX.V4". This is the OPC Server id.
24. Next be sure to set Enable to Yes and click OK.

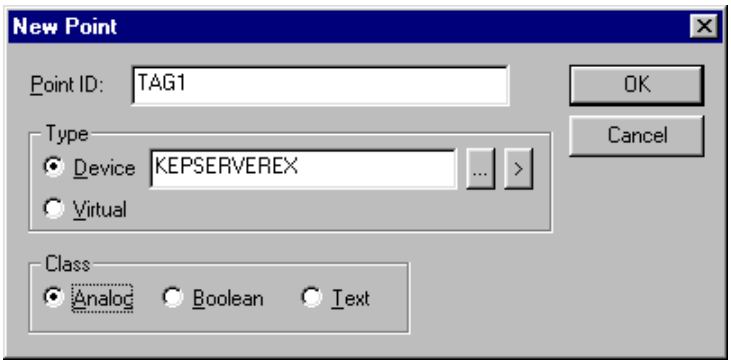


Create a Data Point

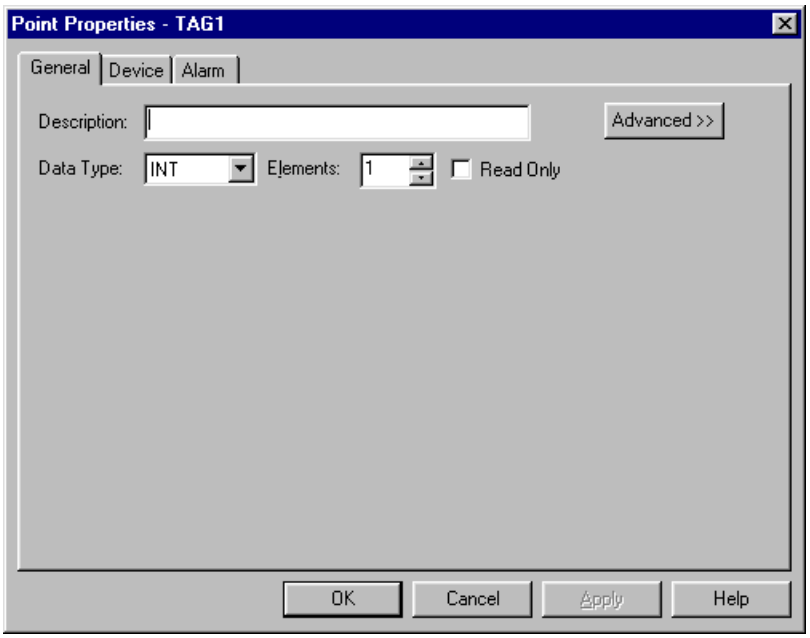
25. In the project tree view click expand Equipment then right click on Points, click New to add a new Device for the OPC connection.



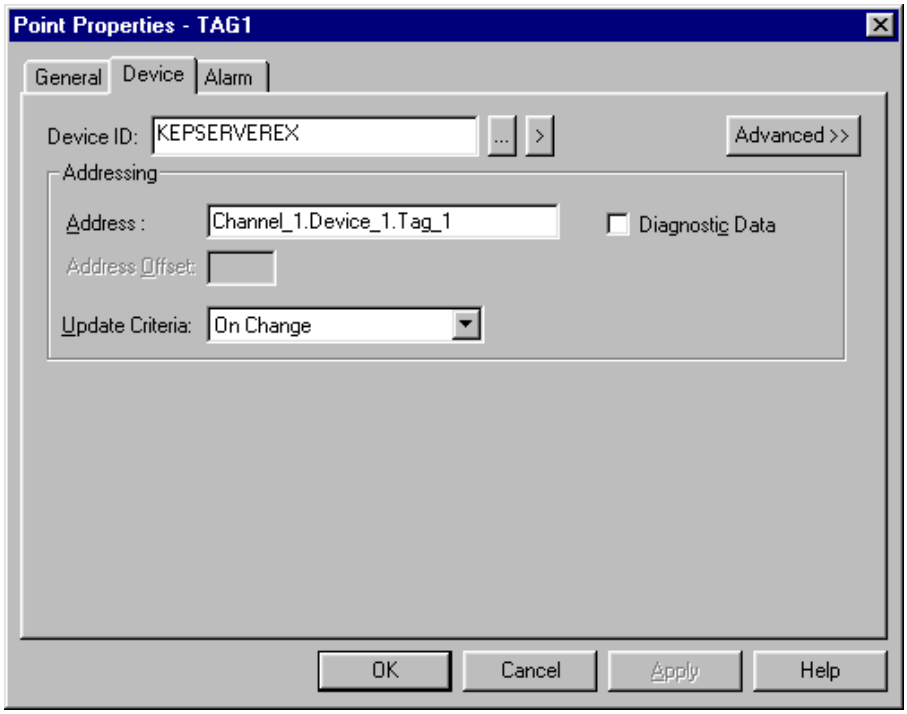
26. Enter a Point ID to identify the point
27. The Type of point will be Device. If you have created more than one device be sure to select KEPServerEX.
28. Select a Class for the point and then click OK. We are creating an Analog tag for our example



29. In the Point Properties dialog box you may enter a Description for the point.
30. Next select the Data Type, if the point is for display only, click Read Only and then click on the Device tab.



31. If the proper Device ID is not selected then do so now.
32. Next enter the Address of the OPC point. We are using the KEPServerEX SimDemo project so the address for this point will be "Channel_1.Device_1.Tag_1".
33. Lastly, select the Update Criteria for the point. We selected On Change so that we will see the newest data all of the time.
34. Now click OK to add the point and you are ready to use it in your project.



Check the OPC Connection

35. As a quick test, if you select run in the project it will connect the points to the server. Next, check the Status bar at the lower right edge of the KEPServerEX window, you should see at least one Active Item. You can also view the data points in a Process Control Panel.
36. Now click OK to add the point and you are ready to use it in your project.

Upgrading a Project from Kepserver V3.2 to V4.0

Some user may be upgrading projects that were using the Kepware legacy server V3.2. If you are one of these users then you will want to read the following steps closely to ensure a quick and easy upgrade to the new server. Since you are upgrading your project to use the Kepware Enhanced OPC/DDE Server you have probably have already seen that the Kepserver model was redesigned. This was done to take advantage of improved technology and methods.

1. The biggest impact to you as a user is the change in the way you connect to the server.
2. Once you have installed KEPServerEX on your PC start it and open your old server project file in the new server and save it. Your project is now running you no longer have to turn it on and off line.
3. Next you will want to open your Cimplicity project and select you the device that you created and edit its Device properties.
4. Select the Default tab and edit the device address.
5. You will notice that the old ID is “Kepserver”. The new server’s address is “KEPware.KEPServerEX.V4”. Enter the new address and click OK to accept the changes.
6. You should now be able to run your project and connect to the new server.

Using Kepware's OPC Quick Client

Kepware provides an OPC client application for testing purposes with each installation of KEPServerEX. For more information on Kepware's **OPC Quick Client**, please see the OPC Quick Client help file.

Appendix A

Cimplicity Master OPC INI files

This section explains the use of some of the sections and parameters found in the Master OPC INI file. Unless otherwise specified in the connectivity guide you will use the defaults when connecting to KEPServerEX.

About the Master OPC INI File

This INI file is for use by the OPC Client devcom provided with CIMPLICITY 4.0.08 or higher, and CIMPLICITY 5.0.1.

This file must reside in the “data” subdirectory of the project directory. There is one “INI” file per OPC port configured in the project. The name reflects the port's ID. For example “MASTER_OPC_0.INI” for the first port, and “MASTER_OPC_1.INI” for the second port configured, etc.

The purpose of this file is to allow the user to manually configure the OPC Port, Devices, and Groups. Unlike the global parameters file (GLB_PARMS) where settings applied to all devices or groups, settings here apply only to the specified device or group.

The sample provided here contains all of the possible parameters. These parameters are set to their default values! If this INI file is not present, or if a particular Device or Group is not specified by a [Section Header], then the default values are used.

For typical BOOLEAN parameters, a value of 1 means TRUE/ENABLE. A value of 0 means FALSE/DISABLE.

> Note: The OPC client always creates TWO GROUPS where ALL points are placed unless you configure them otherwise. These groups are named: DEFAULTPOLL and DEFAULTUNSO. You can set parameters for these groups as well as the groups you define.
>
> Note: There are a couple of parameters that apply to the 'Port' as a whole. These are placed in the section called [PortLevel].

The INI File Sections

[PortLevel]

EightByteReals=0

UseServerTimeStamp=1

MessageTicks=50

CircularLog=1

UseDataTypePromotion=0

[DEVICE1]
StartupDelay=0
ReadDelay=0
PingInterval=5000
PingTimeout=3000
AbortShutdown=0
UseLocalReg=0
DCOMTimeoutThreshold=10000
PingBeforePoll=1
PingBeforeWrite=1
ForceOPC1Server=0
HRBothActive=0

[DEVICE2]
StartupDelay=0
ReadDelay=0
PingInterval=5000
PingTimeout=3000
AbortShutdown=0
PingBeforePoll=1
PingBeforeWrite=1

[DEFAULTPOLL]
ScanRate=10
NoAccessPath=0
DeviceReadAfterSet=1

[MYGROUP01]
ScanRate=1000
DeviceReadAfterSet=1
NoAccessPath=0

[DEFAULTUNSO]
ScanRate=10
DeviceReadAfterSet=1
NoAccessPath=0

PortLevel Section

EightByteReals – *Default = 0*

Should Cimplicity treat Real values as four byte (float) or 8 byte (double) if the OPC server sends single float values, then setting this to one (1) will have no affect on precision. If the server sends doubles, setting this to one (1) will maintain the precision as the value is passed into Cimplicity.

UseServerTimeStamp – *Default = 1*

Should CIMPLICITY use the timestamp passed from the OPC server or provide its own timestamp at the time the update is received.

MessageTicks – *Default = 50*

Controls the rate in Ticks (1 tick = 1/100 seconds) that Windows Messages are pumped by the OPC client. While the client has no use for these messages, it is required by the OS to process them. By default, it drains the message queue once every half second.

CircularLog – *Default = 1*

By default, tracing is to a circular file. If disabled, logging goes to the output file.

UseDataTypePromotion – *Default = 0*

Conversion of data types for many older servers was accomplished via a method called data type promotion. Because the VARIANT was not able to hold signed eight bit (SINT), unsigned sixteen bit (UINT), and unsigned thirty-two bit (UDINT) values, these types were 'promoted' to the next larger type that the VARIANT was capable of holding.

As of Service Pack 4 for Windows NT, the VARIANT was redefined to allow these data types, making data type promotion obsolete. While newer OPC servers are making use of the new VARIANT types many older OPC servers still expect data type promotion to occur.

Since data type promotion is no longer typical, this parameter defaults to disabled.

Device1 Section

StartupDelay – *Default = 0 (Value is milliseconds)*

Some OPC servers take extra time to load their own point databases. Often, points added to the server by Cimplicity are declared invalid because the server was not ready to process additions.

This delay occurs after the OPC server is started, but, before any points are added by Cimplicity.

ReadDelay – *Default = 0 (Value is milliseconds)*

Some OPC servers take extra time, after a client has added points, getting initial values for those points. If a poll occurs too soon values with “Bad Quality” are provided. To avoid this, the Read Delay causes Cimplicty to hesitate after adding points but before the very first poll.

PingInterval – Default = 5000 (Value is milliseconds)

Cimplicity now pings the OPC server on a regular basis. The Ping Interval determines how often this ping occurs.

PingTimeout – *Default = 3000 (Value is milliseconds)*

If the method call made by the Ping (IOPCServer::GetStatus) takes too long to return, Cimplicity declares a communication error. The Ping Timeout determines how long to wait for this method call.

AbortShutdown – *Default = 0*

If this parameter is TRUE (1), Cimplicity will shut down (on project shutdown) without releasing any references to objects in the OPC server. This is strictly against the rules of OLE!!!

However, it will allow the Cimplicity project to shut down in a minimum amount of time. The OPC server will most likely not shut down at all, since it will believe that our client is still attached.

UseLocalReg – *Default = 0*

If this parameter is TRUE (1), Cimplicity will access the local registry information to find the OPC Server ID first. If the devcom fails to find the server information in local registry, it will then log warning message and search the information in the remote node. This parameter is used to resolve the multi-sessions issue.

DCOMTimeoutThreshold – *Default = 10000 (Value is milliseconds)*

After a ping fails, this is the number of milliseconds that the client will wait after detecting a lack of response from the server via a ping before it will force an abort of the connection

PingBeforePoll – *Default = 1*

When a network connection is broken between the Cimplicity and server; in a DCOM situation, the very next method call the client makes (usually a Poll [ISyncIO::Read]) can hang for upwards of 90 seconds. This causes all device communication in Cimplicity to hang until the method call times out. By pinging the OPC server immediately before the method call, Cimplicity eliminates most of these potential DCOM timeouts (since Pinging occurs on a different thread).

This does add the overhead of an extra call to the server per poll request (or write request) but with a DCOM timeout of 90 seconds this may be well worth it.

PingBeforeWrite – *Default = 1*

Same as above but is done after the poll.

ForceOPC1Server – *Default = 0*

Cimplicity by default establish a connection using OPC 2.0 methods if the Server will support it. (It will use 1.0 if the server does not.) To force the use of 1.0 methods, enable this variable.

HRBothActive – *Default = 0*

In Host Redundant environments, by default, the acting master will advise for the points that are defined to be unsolicited/unsolicited on change/poll once/poll once on change (i.e. the server polls the data and notifies the client only of those items that have changed). Some servers may be slow in collecting the data resulting in transition times for the unsolicited data. All data will be advised and refreshed (unless refresh is disabled) on host transition. Enabling this variable will result in both servers advising the client. The result will be more traffic with a lower transition time. It is possible that data that changes during the transition between hosts may be lost for points with an unsolicited or unsolicited on change update criteria.

DefaultPoll Section

The [DEFAULTPOLL] section controls scanning for "On Change" events and issues Synchronous reads and writes to the server. The [DEFAULTUNSO] section controls scanning for unsolicited events and issues Asynchronous reads and writes to the server. The latter type is much mote efficient.

ScanRate – *Default = 1000*

This is the Scan Rate passed to the OPC server for this group. It tells the OPC server how often to update the items in this group with fresh device data. It also determines how often to fire DataChange events to Cimplicity. The unit of measure is in milliseconds and the default value is 1 second.

If you require faster value updates, you can decrease this value to a value as low as zero. Zero tells the OPC server to update values 'as fast as possible'. For servers able to collect rapidly changing data very rapidly, in a DCOM environment, it is possible to send the data faster than can be supported by the hardware/operating system on a sustained basis.

Caution is urged before setting this to a value of zero.

NOTE: As the scan rate decreases CPU usage may rise (somewhat significantly) and a corresponding increase in network traffic may be observed in a DCOM environment.

NoAccessPath – *Default = 0*

Some OPC servers do not use the Access Path when adding points. Instead, the Access Path must be pre-pended in front of the Item ID. Enabling this feature allows these OPC servers to have the Item IDs include the access path, separated by a period.

DeviceReadAfterSet – *Default = 1*

For polled operations, Cimplicity will retrieve required data from the Servers cache. For other operations, (i.e. poll after set) by default, it will request that the OPC Server read the data from the device. This can significantly affect performance where the functionality is fully supported. To force all reads from the cache, disable this variable. It is enabled by default.