NAPOPC_CE5 DA Server

使用手冊

[版本: 2.20]

(支援 I-7000，8000，87000 系列 I/O 模組 與 Modbus 控制器)

保固說明

泓格科技股份有限公司 (ICP DAS) 所生產的產品，均保證原始購買者對於有瑕疵之材料，於交貨日起保有為期一年的保固。

注意事項

泓格科技股份有限公司對於因為應用本產品所造成的任何損害並不負任何法律上的責任，本公司並保留在任何時候修訂本手冊且不需通知的權利。泓格科技股份有限公司將儘可能地提供本系列產品可靠而詳盡的資訊。然而，本公司並無義務需提供此系列產品詳盡的應用資訊，或對因非法或不當使用本系列產品所遭受的損害負任何責任。

著作權

商標

本書提到的所有公司商標、商標名稱及產品名稱分別屬於該商標或名稱的擁有者所有。

軟體授權

用戶可以使用，修改和備份此軟體於一台機器。用戶不得重製，轉讓或散佈本軟體，或任何全部或部分的複製行為。

目錄

1 NAPOPC_CE5 DA Server

什麼是 NAPOPC_CE5 DA Server? NAPOPC_CE5 DA Server 是一套整合了 OPC，Modbus TCP， Modbus RTU 服務 與 Scan kernel 的綜合套裝軟體。 而其特有的 “Rule Script” 設計，可讓使用者快速建立一個具有邏輯控制與多通信服務的 DCS 監控系統。

就 UI 設計而言，NAPOPC_CE5 採用 Explorer-style 的使用者介面來顯示模組的階層式樹狀結構與各群組 (Group) 相關連的標籤 (Tag)。 此 “Group” 可被定義為含有一個或多個 Tag 的子目錄。一個模組可能有許多子群的 Tag。所有掃描出屬於各模組的 Tag，可用來執行 I/O 動作。(“OPC” 代表 “OLE for Process Control”, 而 “DA” 代表 “Data Access”)

就軟體使用而言，NAPOPC_CE5 針對不同類型的使用者建立了最多 3 個步驟的設定程序。此程序簡化了程式設計者的設計過程，並確保兼具穩定性和效益的控制系統。

NAPOPC_CE5 不僅可以自動地對應實體 I/O 到特定的 Modbus 位址，也允許使用者定義自己的變數於其中。 因此，使用者可採用 eVC++， VB.NET 與 VC#.NET 程式語言來開發自己的應用程式，並透過 Modbus RTU 與 Modbus TCP 通訊協定來分享 Modbus 用戶端 (Client) 所指定的資料。此外，使用者可操作 NAPOPC_CE5 並配合 NAPOPC_ST 或 NAPOPC_XPE 建立一個獨特的解決方案來整合 SCADA 圖控軟體與線上資料。

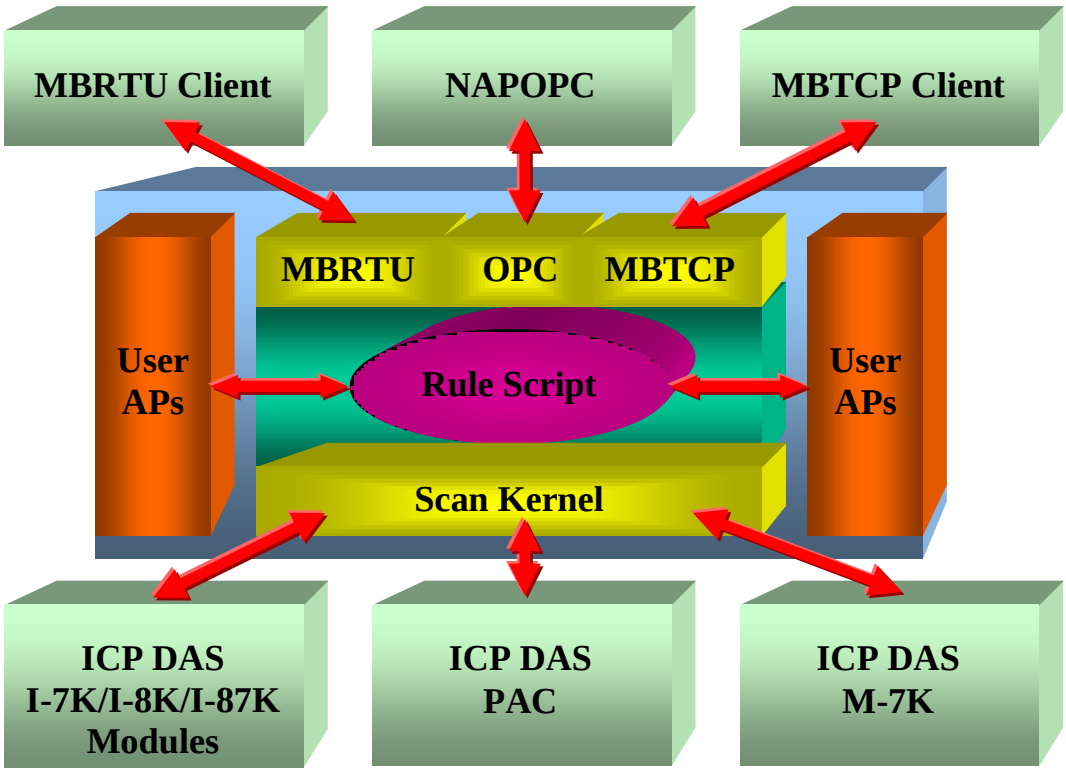


圖 1-1

NAPOPC_CE5 的主要程式為 "NAPOPCSvr_CE5.exe"。此程式會依需求調用 (call) "UARTCE.DLL"，"DCON_CE.DLL"，"MBTool.DLL"，"OPC_WinpacSDK.DLL" ，"opc_i8017HW_ce5.DLL"，"opc_i8024W_ce5.DLL"，"opc_i8084W_ce5.DLL"，"opc_i8172W_ce5.DLL" 與 "Quicker.DLL" 函數。

1.1 安裝 NAPOPC_CE5 DA Server

若您是第一次使用 NAPOPC_CE5 軟體，請執行 WinPAC-8000 控制器中，位於 \System_Disk\Tools\ NAPOPC_CE5 路徑下的 “NAPOPC_CE5Boot.exe” 檔案。

之後，“NAPOPC_CE5Boot.exe” 將會自動地註冊 NAPOPC_CE5。此外，若您希望 WinPAC-8000 控制器啟動時，即可自動地執行 "NAPOPCSvr_CE5.exe" 檔案。請參考手冊 winpac_8x4x_user_manual_v1.9.0.pdf 中 “3.5 WinPAC Utility” 章節的 “Auto Execution” 功能並將 “NAPOPC_CE5Boot.exe” 的路徑加入到 “Auto Execution”。

注意: 上述步驟之後，使用 "Save and Reboot" 功能，確保註冊值儲存於 WinPAC 中。

Address \System_Disk\tools\NAPOPC_CE5

Name	Size	Type	Date Modified
DCON_CE	18.5KB	Application Extension	6/29/2010 12:27 PM
history_v2.14	2.55KB	Text Document	6/29/2010 12:27 PM
MBTool	30.5KB	Application Extension	6/29/2010 12:27 PM
module_7K	23.5KB	INI File	6/29/2010 12:27 PM
module_87K	18.6KB	INI File	6/29/2010 12:27 PM
module_8K	12.1KB	INI File	6/29/2010 12:27 PM
module_ET	4.48KB	INI File	6/29/2010 12:27 PM
module_FRnet	633 bytes	INI File	6/29/2010 12:27 PM
NAPOPC_CE5	403 bytes	Batch File	6/29/2010 12:27 PM
NAPOPC_CE5Boot	4.50KB	Application	6/29/2010 12:27 PM
NAPOPCSvr_CE5	900KB	Application	6/29/2010 12:27 PM
opccomn_ps	60KB	Application Extension	6/29/2010 12:27 PM
OPCDAAuto	112KB	Application Extension	6/29/2010 12:27 PM
OPCProxy	72.5KB	Application Extension	6/29/2010 12:27 PM
pac_i8017hw	18.5KB	Application Extension	6/29/2010 12:27 PM
pac_i8024w	11KB	Application Extension	6/29/2010 12:27 PM
pac_i8084w	16.5KB	Application Extension	6/29/2010 12:27 PM
pac_i8172w	6.50KB	Application Extension	6/29/2010 12:27 PM
Quicker	11KB	Application Extension	6/29/2010 12:27 PM
QuickerNet	5.50KB	Application Extension	6/29/2010 12:27 PM
RegEdit	25.5KB	Application	6/29/2010 12:27 PM
regsvr32	6KB	Application	6/29/2010 12:27 PM
Rule	0 bytes	Text Document	6/29/2010 12:28 PM
StOleProxy	311KB	Application Extension	6/29/2010 12:27 PM
Test1	24KB	TDB File	6/29/2010 12:36 PM
UARTCE	19.5KB	Application Extension	6/29/2010 12:27 PM
WinpacSDK	96KB	Application Extension	6/29/2010 12:27 PM

圖 1.1-1

然後，請執行主程式 "NAPOPCSvr_CE5.exe"，此程式將會自行調用 "UARTCE.DLL", "DCON_CE.DLL", "MBTool.DLL", "OPC_WinpacSDK.DLL", "opc_i8017HW_ce5.DLL" ，"opc_i8024W_ce5.DLL", "opc_i8084W_ce5.DLL", "opc_i8172W_ce5.DLL" 與 "Quicker.DLL" 函數 來啟用 NAPOPC_CE5 軟體。

若此 “\System_Disk\Tools\NAPOPC_CE5” 路徑下的檔案已遺失或損毀，您可自行複製出貨光碟中 “/napdos/wp-8x4x_ce50/system_disk/tools/NAPOPC_CE5/” 路徑下

的所有檔案至 “\System_Disk\Tools\NAPOPC_CE5”。

1.2 功能總覽

1.2.1 搜尋模組

"Search Modules..." 功能可讓您自動設定 NAPOPC_CE5。它會搜尋 RS-485 網路與內嵌式模組並自動將其產生標籤 (Tag)。此版本的 NAPOPC_CE5 不僅可產生 AI/AO，DI/DO，Latched DI 與 Counter 標籤並可對應每個標籤至唯一的 Modbus 位址。

步驟 1: 點選功能表 "Add/ Search Modules..." 或  圖示來搜尋模組。

圖 1.2.1-1

步驟 2: 彈出 "Search Modules" 視窗。

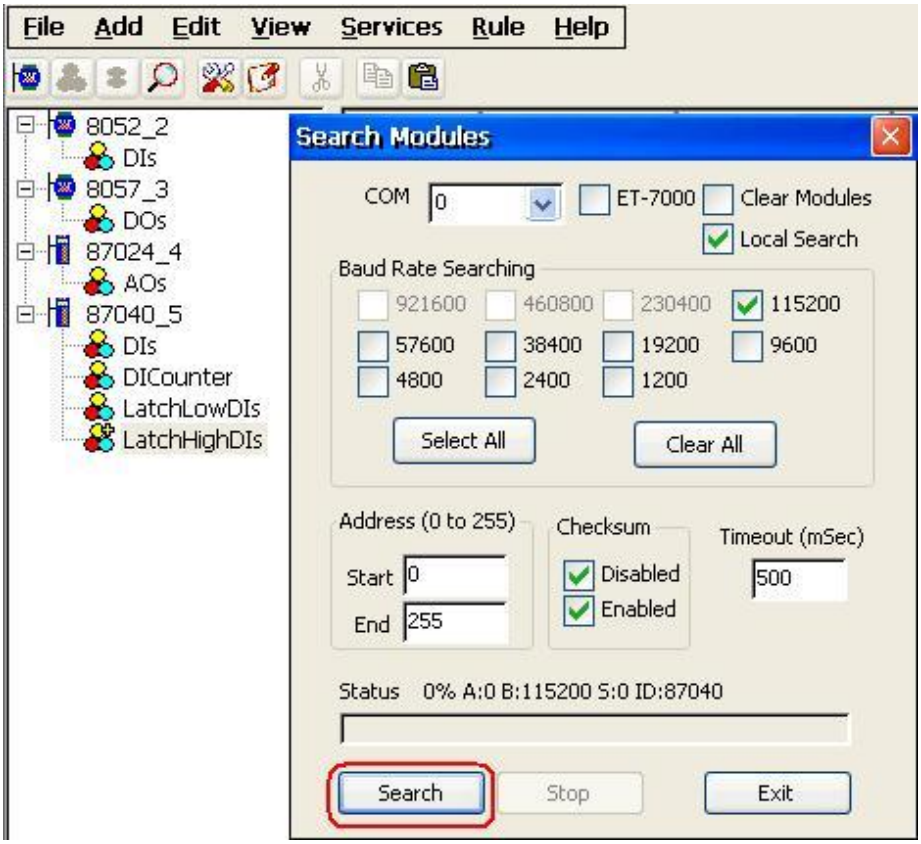


圖 1.2.1-2

步驟 3: 若您想搜尋 WinPAC-8000 控制器上的 I-8K I/O 模組，您必須勾選 “Local Search” 欄位。而 “COM 0” 是用來搜尋 WinPAC-8000 控制器上的 I-87K I/O 模組。

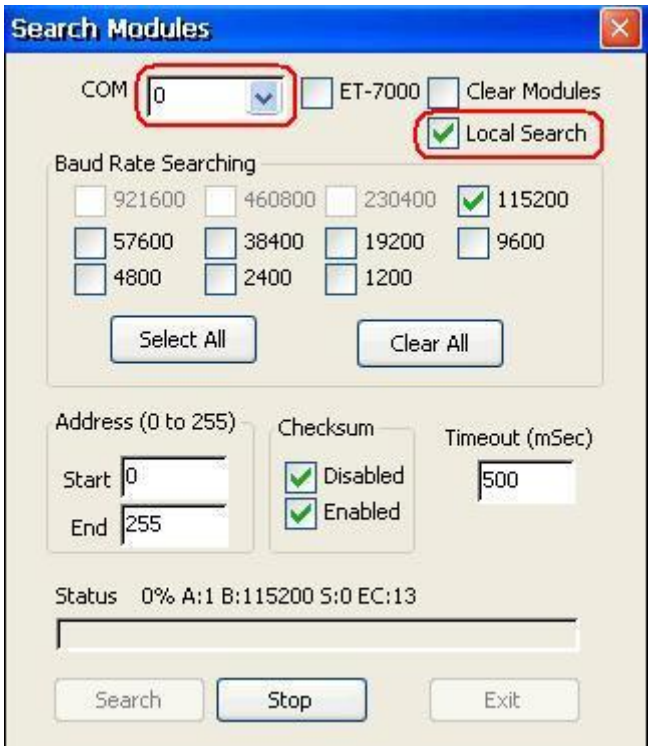


圖 1.2.1-3

步驟 4: 若您想透過 RS-232 來搜尋 I-7K/I-87K 遠程 I/O 模組，您並需選取 “COM 1” 並取消勾選 “Local Search”。

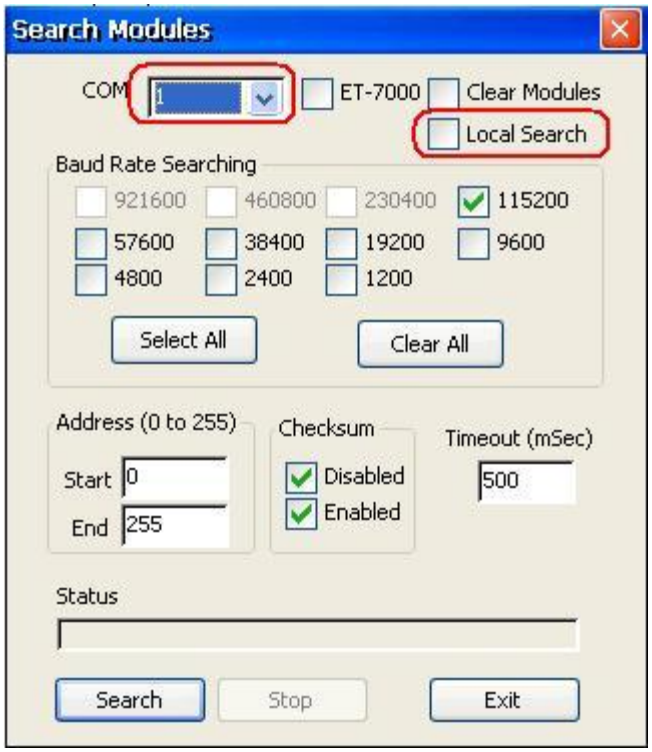


圖 1.2.1-4

步驟 5: 若您想透過 RS-485 來搜尋 I-7K/I-87K 遠程 I/O 模組，並透過 Ethernet 來搜尋 ET-7000 模組，您必需選取 “COM 2” 與勾選 “ET-7000”，並取消勾選 “Local Search”。

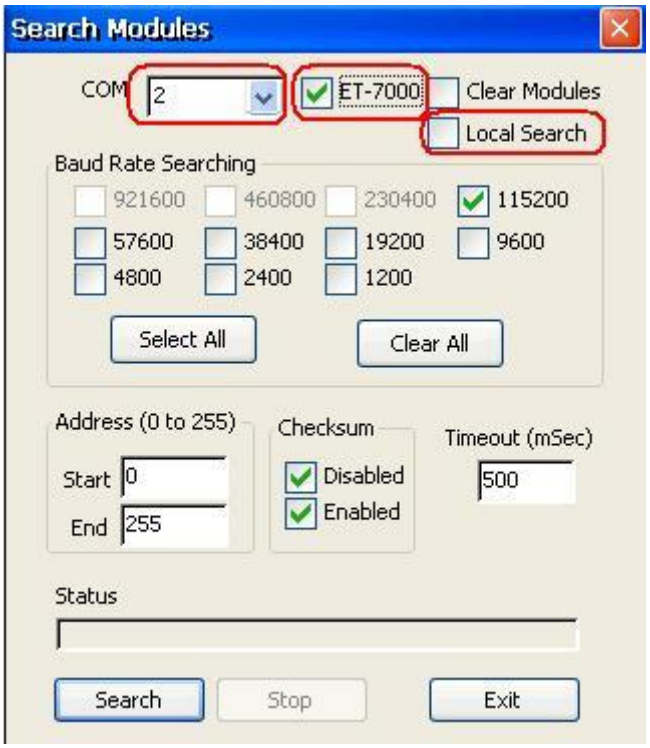


圖 1.2.1-5

COM :

指定欲搜尋的 "COM Port" 編號。預設值為 0，有效範圍為 0 ~ 255。請核對 RS-485 網路所連接的 "COM Port" 編號。

模組	COM 0	COM 1	COM 2	COM 3	COM 4
本機 I-87K	是	-	-	-	-
遠程 I-7K/I-87K (透過 RS-232)	-	是	-	-	-
遠程 I-7K/I-87K (透過 RS-485)	-	-	是	-	-
遠程 I-7K/I-87K (透過 RS-232/485)	-	-	-	是	-
遠程 I-7K/I-87K (透過 RS-232)	-	-	-	-	是

ET-7000:

若勾選此欄位，則 NAPOPC 不僅可自動搜尋那些透過 COM Port 進行通訊的模組，也可搜尋透過 Ethernet 通訊的 ET-7000 模組。

Clear Modules:

模組可被多次加入。若勾選此欄位，會在搜尋之前先刪除所有視窗中的模組列表。 您可勾選它來避免加入重覆的模組。預設為無勾選。

Local Search:

若勾選此欄位，會先搜尋 WinPAC-8000 控制器上的 I-8K 模組。

Baud Rate Searching:

指定欲搜尋的 "Baud Rate" 傳輸速率。預設值為 "9600"。

當然，若勾選多個傳輸速率，其搜尋時間也會變慢。當搜尋多個傳輸速率時，NAPOPC_CE5 必須關閉再重新開啟 COM Port 以與模組進行通訊。這也將降低通訊效能。 因此，強烈建議使用相同的傳輸速率與 COM Port 來搜尋模組。

Select All:

勾選所有的 "Baud Rate" 選項。請參考以上 "Baud Rate Searching" 選項。

Clear All:

取消勾選所有的 "Baud Rate" 選項。請參考以上 "Baud Rate Searching" 選項。

Address/Start:

指定起始位址。預設值為 0，有效範圍為 0 ~ 255。它不會搜尋小於此設定值的位址。

Address/End:

指定結束位址。預設值為 255，有效範圍為 0 ~ 255。它不會搜尋大於此設定值的位址。

Checksum/Disabled:

若勾選此欄位，表示 “無 checksum” 的模組搜尋方式。若 "Disabled" 與 "Enabled" 欄位皆無勾選，則此搜尋方式是未定義的。

Checksum/Enabled:

若勾選此欄位，表示 “有 checksum” 的模組搜尋方式。若 "Disabled" 與 "Enabled" 欄位皆無勾選，則此搜尋方式是未定義的。

Timeout:

指定每個模組通訊逾時的時間。預設值為 200 (相當於 0.2 秒)，以毫秒 (0.001 秒) 為量測標準。搜尋到模組後，此逾時值將被記錄下來以備進一步的使用。使用者可減低此設定值來縮短搜尋時間。但要小心，更短的搜尋時間可能會造成通訊失敗。

Status:

顯示搜尋狀態 (包含: 進度 - %，位址 - "A:??"，Baud-Rate - "B:????"，Checksum - "S:?" 與 錯誤代碼 - "EC:??") 。 逾時的錯誤代碼為 15。在大多數情況下，表示模組沒有對當前的命令做出回應。

Search:

完成上述設定後，點選此按鈕開始搜尋。此視窗將於完成後自動關閉。

Stop:

搜尋期間，使用者可點選此按鈕來停止搜尋。取消搜尋後，該視窗將停留在螢幕上。

Exit:

使用者可點選此按鈕來離開視窗。

步驟 6: 搜尋完成後，找到的模組會列於左邊的 “設備視窗”。使用者也可於右邊的 “Tag 視窗” 看見由 "Search Modules..." 功能自動產生的標籤 (Tag)。

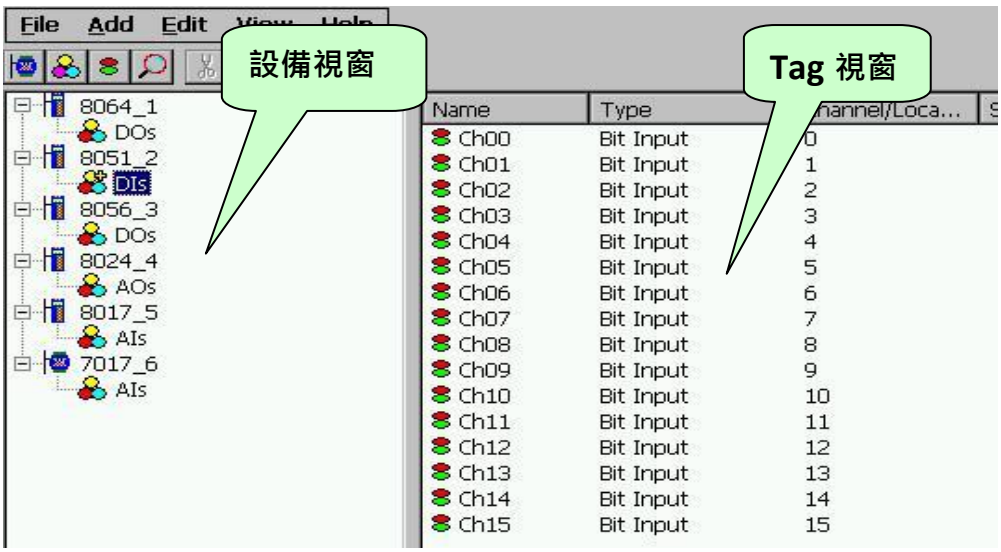


圖 1.2.1-6

此 "Search Modules..." 功能會產生 "Digital Input"，"Digital Output"，"Bit Input" 或 "Bit Output" 型別的標籤 (Tag)。

"Digital Input" 與 "Digital Output" 標籤是採取一次通訊即讀取全部通道 (channel) 狀態的方式，而 "Bit Input" 與 "Bit Output" 標籤則是採取一次通訊只讀取一個通道狀態的方式。相較之下，"Digital Input" 與 "Digital Output" 標籤 擁有更佳的效能，因此強烈建議使用 "Digital Input" 與 "Digital Output" 標籤來存取模組。

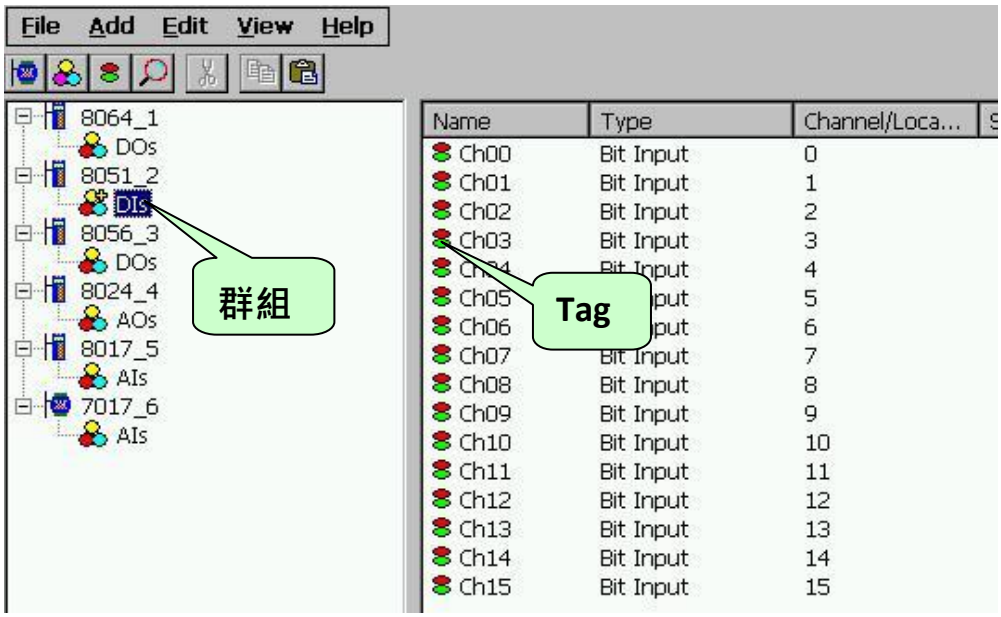


圖 1.2.1-7

1.2.2 監測設備

您可在功能表 "View/ Monitor" 中勾選 "Monitor" 功能，以查看各標籤 (Tag) 的數值。取消勾選則表示停止監測功能。

步驟 1: 點選功能表 "View/ Monitor" 來啟用監測功能。

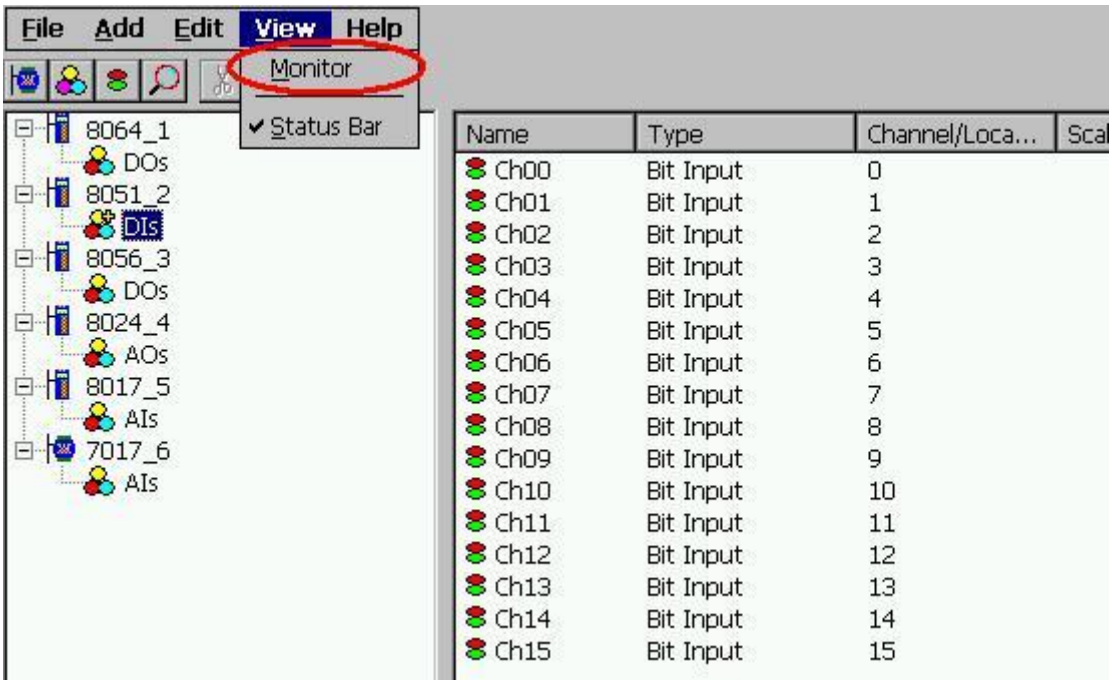


圖 1.2.2-1

步驟 2: 如下圖，點選左邊設備視窗的 "AIs" 群組來監測屬於該群組的 AI Tag。

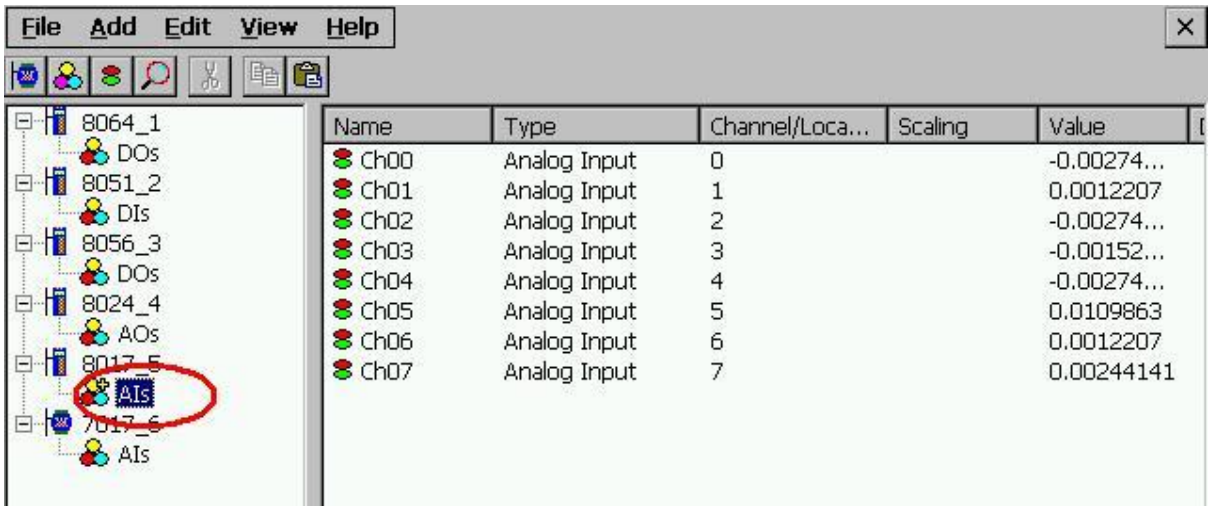


圖 1.2.2-2

步驟 3: 如下圖，點選設備視窗的 "8064" 模組來監測屬於該模組的 DO Tag。

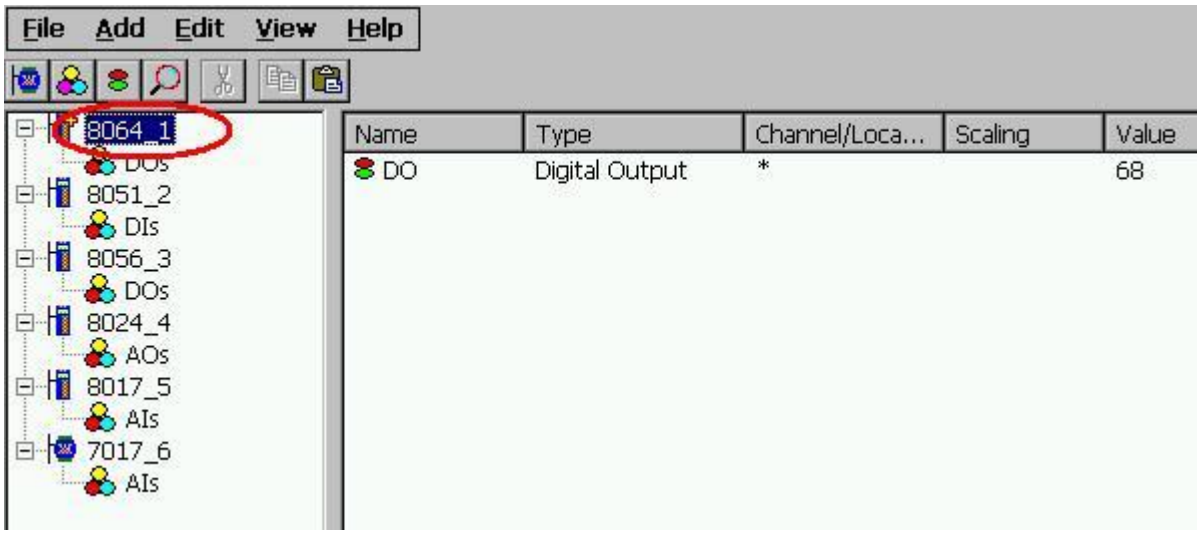


圖 1.2.2-3

1.2.3 新增設備

NAPOPC_CE5 提供了三種設備模式，您可新增 “DCON 設備”，”FRnet 設備” 與 “Modbus 設備”。 “DCON 設備” 包含了 “I-8K/87K 內嵌式模組”，”遠程 I/O 模組” 與 “內部設備”。“內部設備” 可以是介於多個使用者的應用程式之間的中介容器 (Container) 或是設計 “Rule Script” 的中介設備。“FRnet 設備” 支援泓格科技 (ICP DAS) 所生產的 FRnet 模組。”Modbus 設備” 支援 “Modbus RTU”，”Modbus ASCII” 與 “Modbus TCP” 通訊協定。

NAPOPC_CE5 提供了透過 COM Port 與 Ethernet 的多執行緒通訊，預設為最多 32 個 Modbus TCP Master 執行緒的通訊限制。

1.2.3.1 新增 I-8K/I-87K 內嵌式模組

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

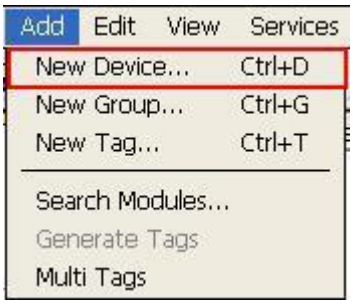


圖 1.2.3.1-1

步驟 2: 彈出 "Select Device" 視窗。

步驟 3: 點選 "DCON" 單選按鈕。

步驟 4: 點選 "I-8K/I-87K Embedded Modules" 單選按鈕。

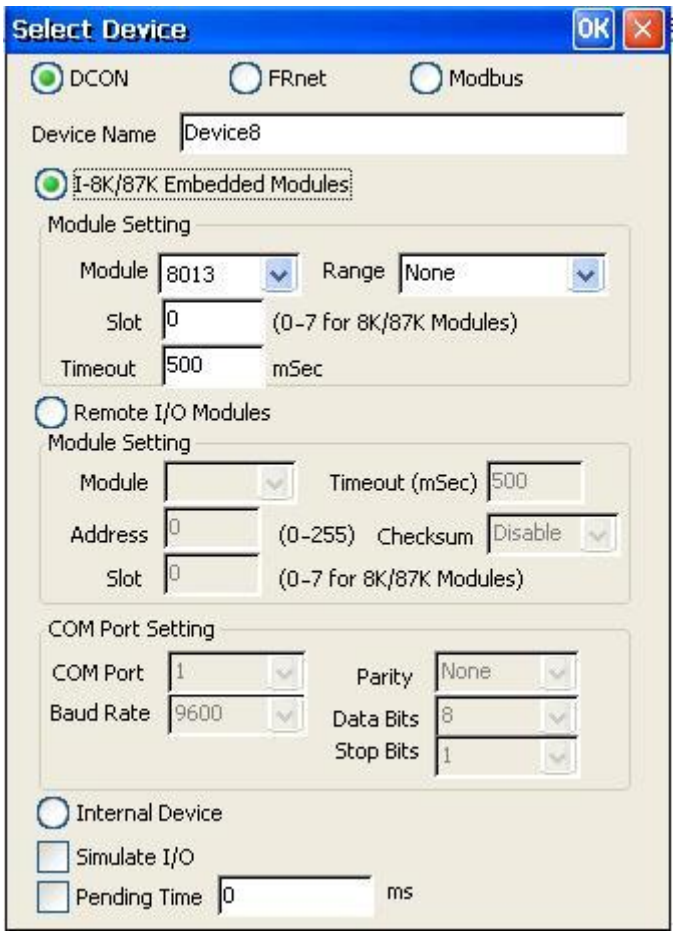


圖 1.2.3.1-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，而 "Device Name" 不可與其他模組相同。

Module:

使用者可點選下拉式選單來選取模組的型號 (ID)。

Timeout:

指定該模組的逾時值 (回應時間)。逾時值太小可能會造成通訊失敗，而逾時值太大可能會降低用戶端程式的效能。

Slot:

WinPAC-8000 擁有 4 或 8 個插槽可供使用。此 “slot” 欄位表示 I/O 模組所使用的插槽編號。有效範圍為 0 ~ 7。

Range:

此設定適用於 I-8017 與 I-8024 模組。請參閱該模組手冊來選擇正確的範圍。

Simulate I/O:

勾選 “Simulate I/O” 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟 TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。

Pending Time:

兩項存取動作的最小間隔時間。啟用此功能，則 NAPOPC_CE5 可在最佳通訊效能下運作。若此模組只需在每 5 秒存取一次，您可設定等待時間為 5000 ms，NAPOPC_CE5 會自動將時間資源分配給其他相連接的模組。

步驟 5: 點選 "OK" 按鈕，新增此模組 。

1.2.3.2 新增遠程 I/O 模組

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

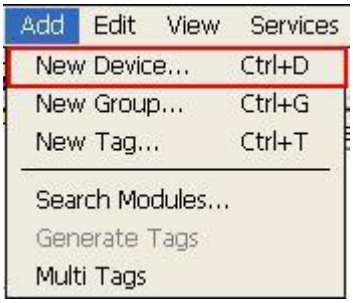


圖 1.2.3.2-1

步驟 2: 彈出 "Select Device" 視窗。

步驟 3: 點選 "DCON" 單選按鈕。

步驟 4: 點選 "Remote I/O Modules" 單選按鈕。

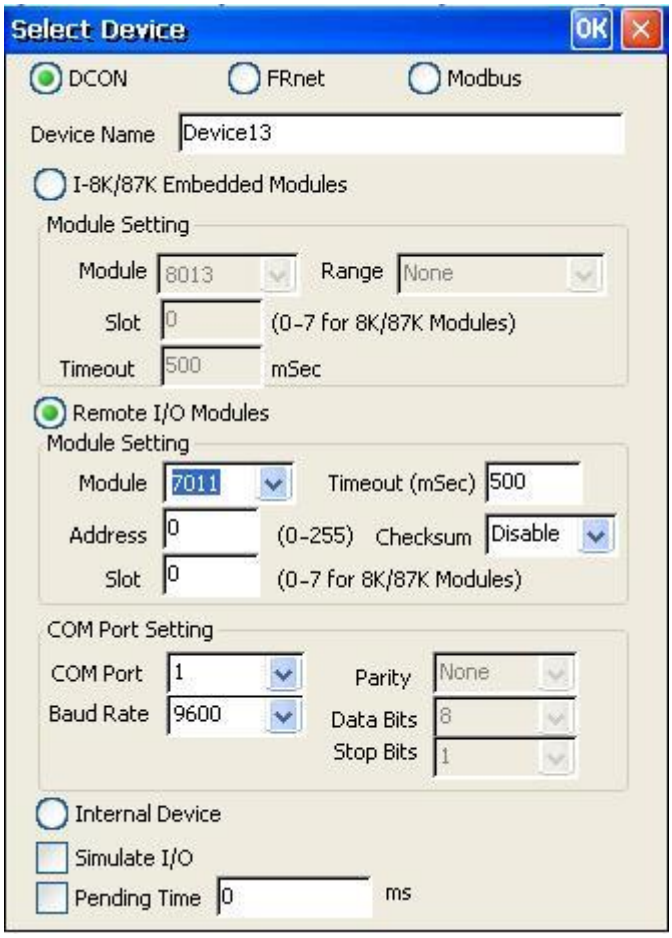


圖 1.2.3.2-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，而 "Device Name" 不可與其他模組相同。

Module:

使用者可點選下拉式選單來選取模組或 I/O 擴充單元的型號 (ID)。

Address:

指定每個模組 (例如: 7000 系列) 或 8000 系列主設備的位址。預設值為 0，有效範圍為 0 ~ 255。此欄位可用在設定 8000 系列 I/O 擴充單元 (main-device) 的位址，但是不提供給 8000 系列模組 (sub-devices) 設定。

Slot:

8000 系列主設備 擁有 4 或 8 個插槽可插上 8000 系列子設備。此 “slot” 欄位是指該 8000 系列子設備使用的插槽編號。有效範圍為 0 ~ 7。此欄位不提供給 8000 系列主設備與 7000 系列模組設定。

Timeout:

指定該模組的逾時值 (回應時間)。逾時值太小可能會造成通訊失敗，而逾時值太大可能會降低用戶端程式的效能。此欄位用在設定 8000 系列主設備的逾時值，但是不提供給 8000 系列子設備設定。

Checksum:

Checksum 欄位必須與硬體設定相符。若不相符，往往會造成該模組的通訊失敗。此欄位用在設定 8000 系列主設備的 checksum，但是不提供給 8000 系列子設備設定。

COM Port:

指定需使用的 COM Port。請檢視 RS-485 網路目前是使用哪個 COM Port 編號。而錯誤的設定將會造成模組通訊失敗。此欄位用在設定 8000 系列主設備的 COM Port，但是不提供給 8000 系列子設備設定。

Baud Rate:

指定需使用的傳輸速率。請檢視模組的傳輸速率是正確的。而錯誤的設定將會造成模組的通訊錯誤。此欄位用在設定 8000 系列主設備的 Baud Rate，但是不提供給 8000 系列子設備設定。

Simulate I/O:

勾選 “Simulate I/O” 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟 TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。但此欄位不提供給 8000 系列主設備設定。

Pending Time:

兩項存取動作的最小間隔時間。啟用此功能，NAPOPC_CE5 可在最佳通訊效能下運作。若此模組只需在每 5 秒存取一次，您可設定等待時間為 5000 ms，NAPOPC_CE5 會自動將時間資源分配給其他相連接的模組。

OK:

點選 "OK" 按鈕，新增此模組設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

步驟 5: 點選 "OK" 按鈕，新增此模組。

1.2.3.3 新增內部設備

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

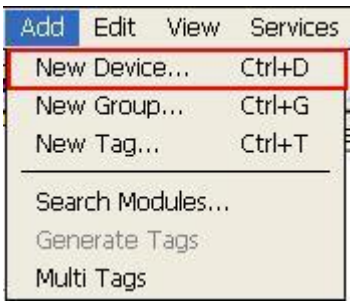


圖 1.2.3.3-1

步驟 2: 彈出 "Select Device" 視窗。

步驟 3: 點選 "DCON" 單選按鈕。

步驟 4: 點選 "Internal Device" 單選按鈕。

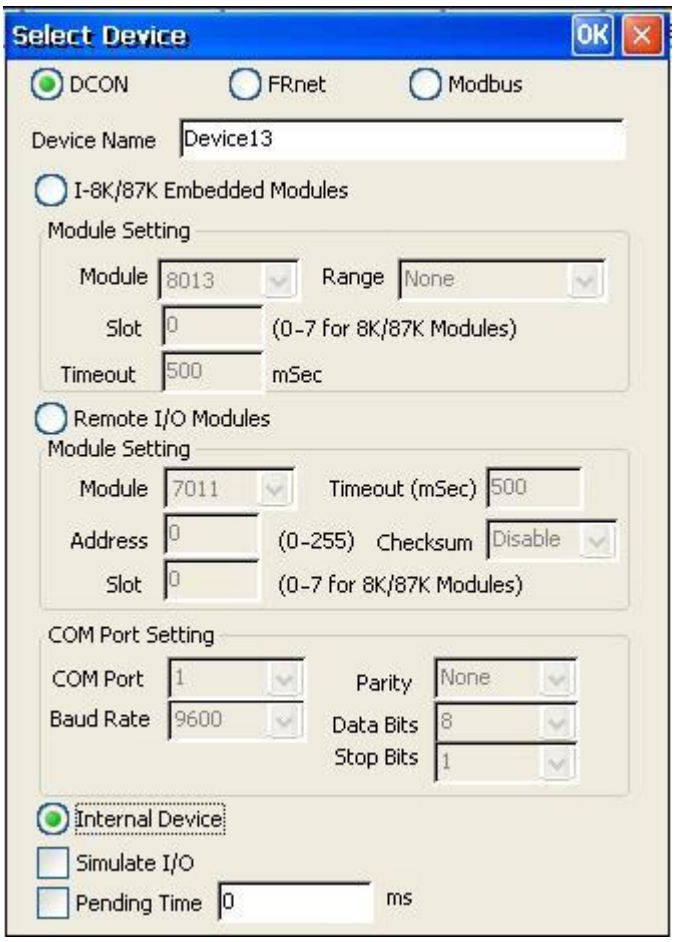


圖 1.2.3.3-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端可使用 “Device Name” 與 “Tags” 來存取其值，而 “Device Name” 不可與其他模組相同。

步驟 5: 點選 "OK" 按鈕，新增此模組。

1.2.3.4 新增 FRnet 設備

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

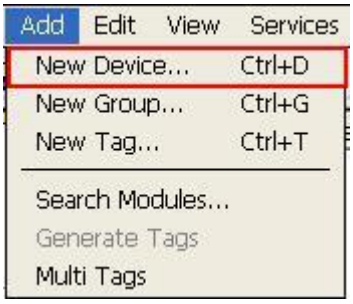


圖 1.2.3.4-1

步驟 2: 彈出 "Select Device" 視窗。

步驟 3: 點選 "FRnet" 單選按鈕。

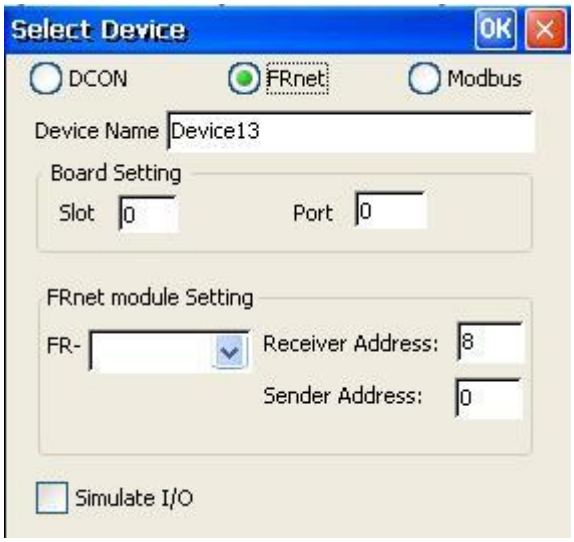


圖 1.2.3.4-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端可使用 “Device Name” 與 “Tags” 來存取其值，而 “Device Name” 不可與其他模組相同。

Slot:

WinPAC-8000 擁有 4 或 8 個插槽可供使用。此 “Slot” 欄位表示 I/O 模組所使用的插槽編號。有效範圍為 0 ~ 7。

Port:

"Port" 是指 I-8172 的 Port 編號 (Port 0 或 1)。 每個 FRnet I/O 模組必須使用 I-8172 為 FRnet 通訊模組。

FR-:

使用者可點選下拉式選單來選取 FRnet 模組的型號 (ID)。

Receiver Address:

FRnet 通訊必須擁有正確的硬體設置，包含設定主機控制器、網路中遠程模組的發送者位址 (SA)、接收者位址 (RA)。請參閱 FRnet 手冊取得更多資訊。

Sender Address:

FRnet 通訊必須擁有正確的硬體設置，包含設定主機控制器、網路中遠程模組的發送者位址 (SA)、接收者位址 (RA)。請參閱 FRnet 手冊取得更多資訊。

Simulate I/O:

勾選 "Simulate I/O" 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。

1.2.3.5 新增 Modbus RTU 控制器

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

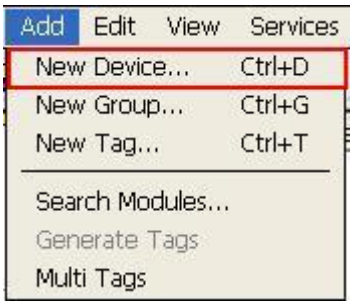


圖 1.2.3.5-1

步驟 2: 彈出 "Select Device" 視窗。

步驟 3: 點選 "Modbus" 單選按鈕。

步驟 4: 點選 "Modbus RTU" 單選按鈕。

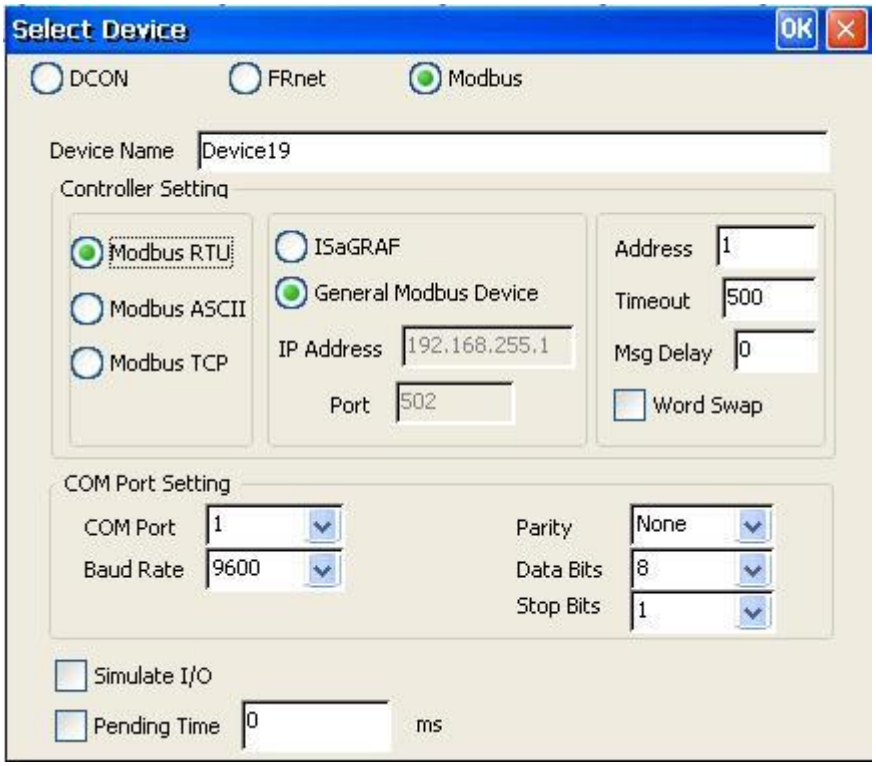


圖 1.2.3.5-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端可使用 “Device Name” 與 “Tags” 來存取其值，而 “Device Name” 不可與其他模組相同。

ISaGRAF:

連接至 ISaGRAF 控制器。

General Modbus Device:

連接至 一般的 Modbus 設備。

Address:

指定控制器的位址。預設值為 1，有效範圍為 1 ~ 255。

Timeout:

指定該控制器的逾時值 (回應時間)。逾時值太小可能會造成通訊失敗，而逾時值太大可能會降低用戶端程式的效能。

Msg Delay:

指定該控制器的訊息延遲時間。預設值為 0 ms，較小的訊息延遲時間會有較高的系統負載，但會有較快的資料交換率。

Word Swap:

勾選 “Word Swap” 可將資料轉譯為 4 Byte 的值，以完成 Lo-Hi/Hi-Lo 通訊。

COM Port:

指定需使用的 COM Port。請檢視 RS-485 網路目前是使用哪個 COM Port 編號。而錯誤的設定將會造成模組通訊失敗。

Baud Rate:

指定需使用的傳輸速率。請檢視模組的傳輸速率是正確的。而錯誤的設定將會造成模組的通訊錯誤。

Parity:

指定需使用的同位元檢查格式。下列為其選項：

選項	說明
None	無同位元
Even	偶同位元
Odd	奇同位元

Data Bits:

指定傳送與接收一位元組 (Bytes) 資料，所包含的位元 (Bits) 數。

Stop Bits:

指定需使用的停止位元數。下列為其選項：

選項	說明
1	1 個停止位元
2	2 個停止位元
1.5	1.5 個停止位元

Simulate I/O:

勾選 “Simulate I/O” 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟 TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。

Pending Time:

兩項存取動作的最小間隔時間。啟用此功能，則 NAPOPC_CE5 可在最佳通訊效能下運作。若此模組只需在每 5 秒存取一次，您可設定等待時間為 5000 ms，NAPOPC_CE5 會自動將時間資源分配給其他相連接的模組。

OK:

點選 "OK" 按鈕，新增此控制器的設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

步驟 5: 點選 "OK" 按鈕，新增此設備。

1.2.3.6 新增 Modbus ASCII 控制器

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

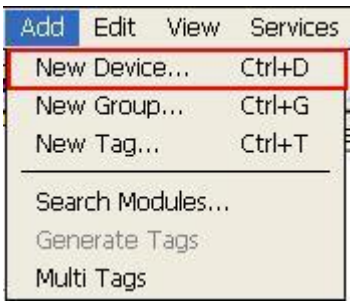


圖 1.2.3.6-1

步驟 2: 彈出 “Select Device” 視窗。

步驟 3: 點選 "Modbus" 單選按鈕。

步驟 4: 點選 "Modbus ASCII" 單選按鈕。

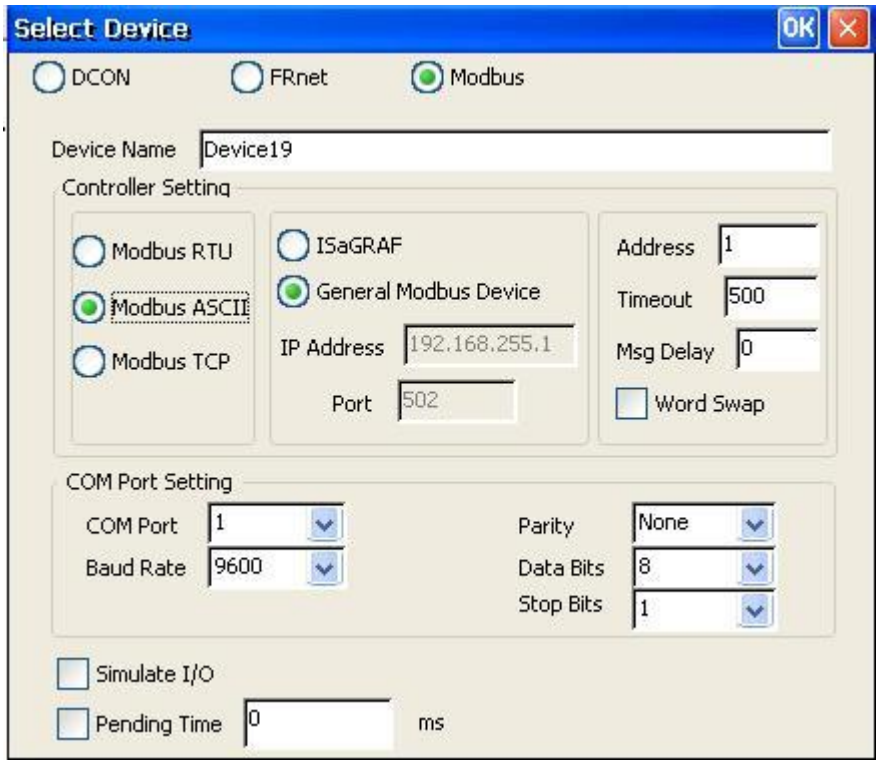


圖 1.2.3.6-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，而 “Device Name” 不可與其他模組相同。

ISaGRAF:

連接至 ISaGRAF 控制器。

General Modbus Device:

連接至 一般的 Modbus 設備。

Address:

指定控制器的位址。 預設值為 1，有效範圍為 1 ~ 255。

Timeout:

指定該控制器的逾時值 (回應時間)。逾時值太小可能會造成通訊失敗，而逾時值太大可能會降低用戶端程式的效能。

Msg Delay:

指定該控制器的訊息延遲時間。預設值為 0 ms，較小的訊息延遲時間會有較高的系統負載，但會有較快的資料交換率。

Word Swap:

勾選 “Word Swap” 可將資料轉譯為 4 Byte 的值，以完成 Lo-Hi/Hi-Lo 通訊。

COM Port:

指定需使用的 COM Port。請檢視 RS-485 網路目前是使用哪個 COM Port 編號。而錯誤的設定將會造成模組通訊失敗。

Baud Rate:

指定需使用的傳輸速率。請檢視模組的傳輸速率是正確的。而錯誤的設定將會造成模組的通訊錯誤。

Parity:

指定需使用的同位元檢查格式。下列為其選項：

選項	說明
None	無同位元
Even	偶同位元
Odd	奇同位元

Data Bits:

指定傳送與接收一位元組 (Bytes) 資料，所包含的位元 (Bits) 數。

Stop Bits:

指定需使用的停止位元數。下列為其選項：

選項	說明
1	1 個停止位元
2	2 個停止位元
1.5	1.5 個停止位元

Simulate I/O:

勾選 “Simulate I/O” 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟 TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。

Pending Time:

兩項存取動作的最小間隔時間。啟用此功能，則 NAPOPC_CE5 可在最佳通訊效能下運作。若此模組只需在每 5 秒存取一次，您可設定等待時間為 5000 ms，NAPOPC_CE5 會自動將時間資源分配給其他相連接的模組。

OK:

點選 "OK" 按鈕，新增此控制器的設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

步驟 5: 點選 "OK" 按鈕，新增此設備。

1.2.3.7 新增 Modbus TCP 控制器

步驟 1: 點選功能表 "Add/ New Device..." 或  圖示來新增模組。

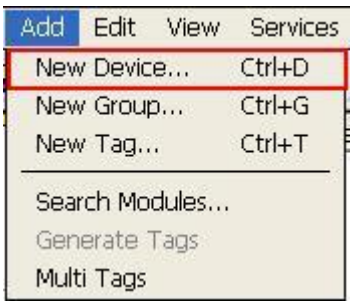


圖 1.2.3.7-1

步驟 2: 如下圖，彈出 “Select Device” 視窗。

步驟 3: 點選 "Modbus" 單選按鈕。

步驟 4: 點選 "Modbus TCP" 單選按鈕。

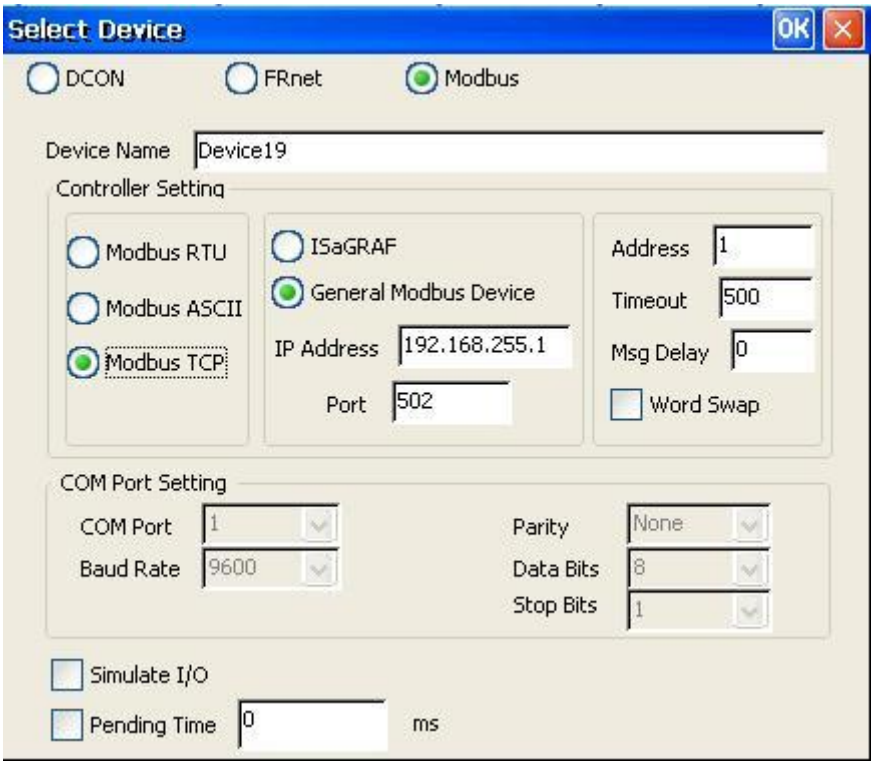


圖 1.2.3.7-2

Device Name:

名稱中有空格或標點符號，例如 “|!.,” 不可作為模組名稱。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，而 “Device Name” 不可與其他模組相同。

ISaGRAF:

連接至 ISaGRAF 控制器。

General Modbus Device:

連接至 一般的 Modbus 設備。

IP Address:

Modbus TCP 控制器的唯一 IP 位址。

Port: 您必須設定為 “502” 以與 ICP DAS Modbus TCP 控制器通訊。

Address:

指定控制器的位址。 預設值為 1，有效範圍為 1 ~ 255。

Timeout:

指定該控制器的逾時值 (回應時間)。逾時值太小可能會造成通訊失敗，而逾時值太大可能會降低用戶端程式的效能。

Msg Delay:

指定該控制器的訊息延遲時間。預設值為 0 ms，較小的訊息延遲時間會有較高的系統負載，但會有較快的資料交換率。

Word Swap:

勾選 “Word Swap” 可將資料轉譯為 4 Byte 的值，以完成 Lo-Hi/Hi-Lo 通訊。

Simulate I/O:

勾選 “Simulate I/O” 會切換至讀取 I/O 的模擬器。由於模擬器不會去開啟 TCP/IP 或 COM Port，它可更簡易地處理 Server 作業，配置標籤 或 連接至用戶端而無需任何硬體設備。

Pending Time:

兩項存取動作的最小間隔時間。啟用此功能，則 NAPOPC_CE5 可在最佳通訊效能下運作。若此模組只需在每 5 秒存取一次，您可設定等待時間為 5000 ms，NAPOPC_CE5 會自動將時間資源分配給其他相連接的模組。

OK:

點選 "OK" 按鈕，新增此控制器的設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

1.2.4 新增群組

步驟 1: 點選功能表 "Add/ New Group" 或  圖示來新增群組。

步驟 2: 彈出 "Group" 視窗。

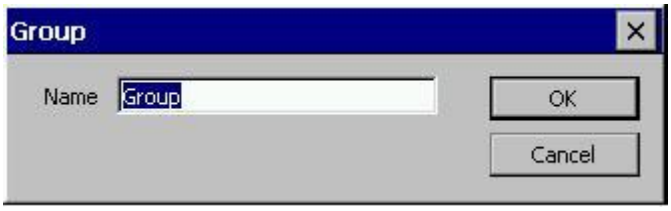


圖 1.2.4-1

Name:

"群組名稱" 可為任何名稱，但請避免使用空白或標點符號 (例如: “|!.,”) 且群組名稱不可重覆。”群組” 可被定義為含有一個或多個 Tag 的子目錄。一個模組可能有許多子群的 Tag。所有掃描出屬於各模組的 Tag，可用來執行 I/O 動作。

1.2.5 新增標籤 (Tag)

1.2.5.1 新增 I-7K/8K/87K/ZigBee/FRnet 模組用標籤

步驟 1: 點選功能表 "Add/ Generate Tags" 來新增 Tag。

圖 1.2.5.1-1

步驟 2: "Generate Tags" 功能將會為您所選取的設備產生 Tag。

步驟 3: 雙擊 Tag 來修改其屬性。

步驟 4: 點選 "Settings" 頁籤。 由於此 Tag 是屬於模組類型的設備，您可於下圖中見到 "I/O Module" 單選按鈕是啟用的狀態。

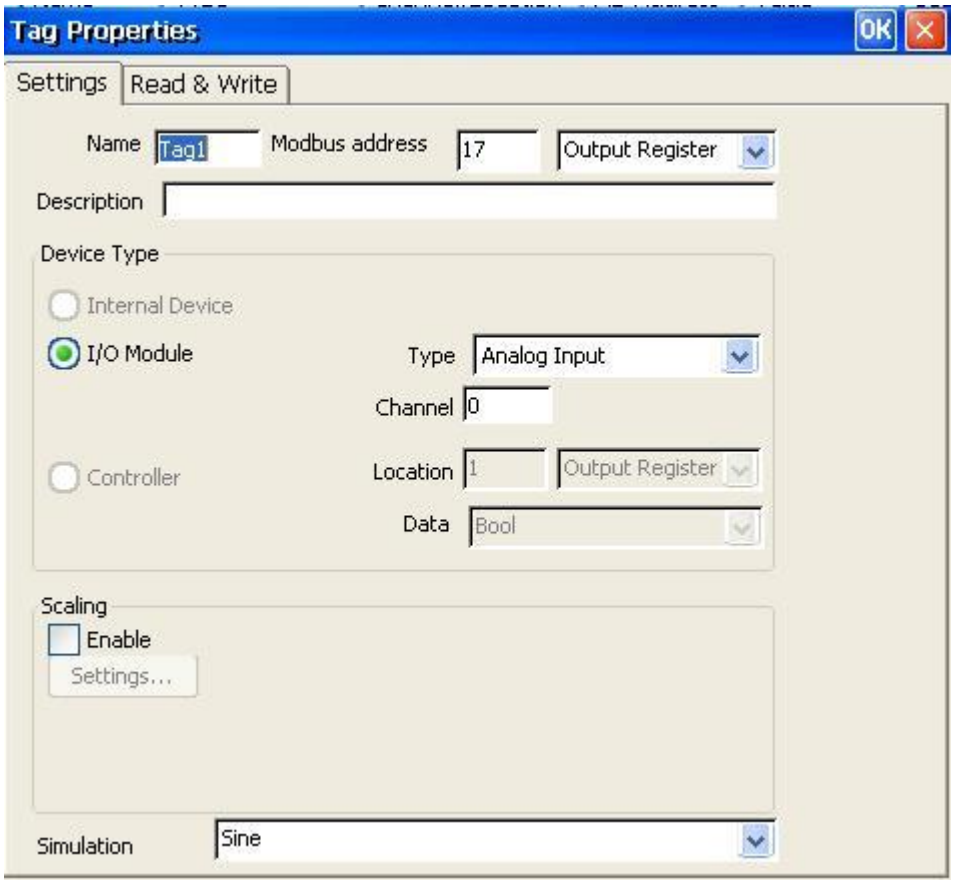


圖 1.2.5.1-2

Name:

"Tag Name" 可為任何名稱，但請避免使用空白或標點符號 (例如: “|!.,”)。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，因此同一群組的 "Tag Name" 不可重覆。

Modbus address:

為此 Tag 指定一個唯一的 Modbus 位址，以便與 Modbus 用戶端通訊。而預設值即是唯一的位址。此外，您還需選擇位址類型，以下共有 4 種位址類型可供選擇。依照標籤的屬性，分別是 “Input Coil”，”Output Coil”，”Input Register” 與 “Output Register” 。然而，指定適當的 Modbus 位址類型與位址是相當重要的。

位址類型	範圍
Output Coil	000001 - 001000
Input Coil	100001 - 101000
Input Register	300001 - 301000
Output Register	400001 - 401000

Description:

為此 Tag 加上文字描述。此處可為空白。

Type:

顯示用於此 Tag 的命令。不同的模組支援不同的命令。

Channel:

指定用於此 Tag 的通道編號。"數位輸入" 與 "數位輸出" Tag 無需採用此項設定，因為皆是一次通訊即讀取全部的通道。

Simulation:

其有效訊號為 SINE，RAMP 與 RANDOM。當模組使用 “Simulation I/O” 功能時，此欄位才會生效。請參閱 "1.2.3 新增設備" 章節。

OK:

點選 "OK" 按鈕，新增此 Tag 設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

Scaling:

Enable:

勾選此項目，啟用 "Settings..." 按鈕。

Settings:

點選此按鈕， 設定 “Scaling” 功能。請參閱 "1.2.5.4 比例設定" 章節，以取得詳細資訊。

1.2.5.2 新增 網路設備用標籤

步驟 1: 點選功能表 "Add/ New Tag" 或  圖示來新增 Tag。

步驟 2: 彈出 "Tag Properties" 視窗。

步驟 3: 點選 “Settings” 頁籤。由於此 Tag 是屬於網路類型設備，因此 “Internal Device” 單選按鈕為啟用狀態。

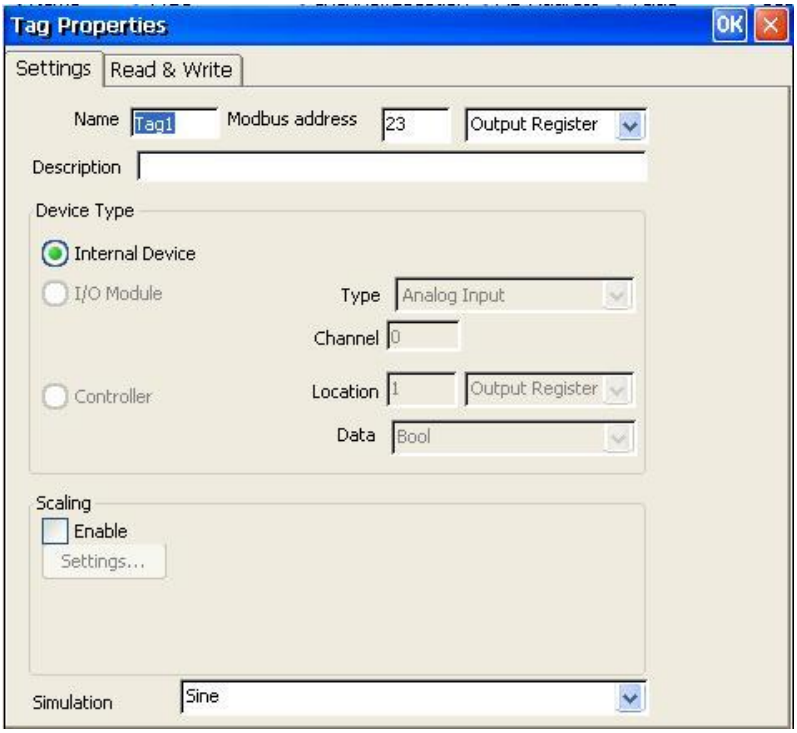


圖 1.2.5.2-1

Name:

"Tag Name" 可為任何名稱，但請避免使用空白或標點符號 (例如: “|!.,”)。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，因此同一群組的 “Tag Name” 不可重覆。

Modbus address:

為此 Tag 指定一個唯一的 Modbus 位址，以便與 Modbus 用戶端通訊。而預設值即是唯一的位址。此外，您還需選擇位址類型，以下共有 4 種位址類型可供選擇。依照標籤的屬性，分別是 “Input Coil”，”Output Coil”，”Input Register” 與 “Output Register”。然而，指定適當的 Modbus 位址類型與位址是相當重要的。

位址類型	範圍
Output Coil	001001 - 020999
Input Coil	101001 - 120999
Input Register	301001 - 320999
Output Register	401001 - 420999

Description:

為此 Tag 加上文字描述。此處可為空白。

1.2.5.3 新增 Modbus 設備用標籤

步驟 1: 點選功能表 "Add/ New Tag" 或  圖示來新增 Tag。

步驟 2: 彈出 "Tag Properties" 視窗。

步驟 3: 點選 “Settings” 頁籤。由於此 Tag 是屬於控制器類型設備，因此 “Controller” 單選按鈕為啟用狀態。

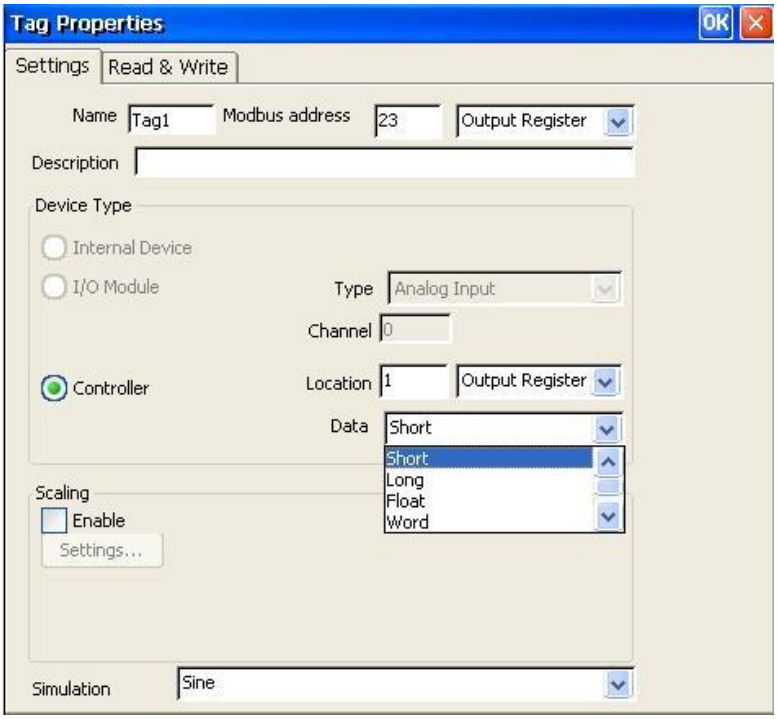


圖 1.2.5.3-1

Name:

“Tag Name” 可為任何名稱，但請避免使用空白或標點符號 (例如: “|!.,”)。用戶端會使用 “Device Name” 與 “Tags” 來存取其值，因此同一群組的 “Tag Name” 不可重覆。

Modbus address:

為此 Tag 指定一個唯一的 Modbus 位址，以便與 Modbus 用戶端通訊。而預設值即是唯一的位址。此外，您還需選擇位址類型，以下共有 4 種位址類型可供選擇。依照標籤的屬性，分別是 “Input Coil”，”Output Coil”，”Input Register” 與 “Output Register”。然而，指定適當的 Modbus 位址類型與位址是相當重要的。

位址類型	範圍
Output Coil	000001 - 001000
Input Coil	100001 - 101000
Input Register	300001 - 301000
Output Register	400001 - 401000

Description:

為此 Tag 加上文字描述。此處可為空白。

Location:

指定 Tag 位址。此位址必須與控制器中的變數位址相同。此外，您還必須選擇位置類型。當您選擇位置編號之後，以下有 4 種位置類型可供您選擇 - “Input Coil”，”Output Coil”，”Input Register” 與 “Output Register”。當您監測控制器設備時 (詳見 1.2.2 監測設備)，”Channel/Location” 欄位將會依據該位置與位置類型顯示其值 (如下表)。

位置類型	範圍
Output Coil	000001 - 065536
Input Coil	100001 - 165536
Input Register	300001 - 365536
Output Register	400001 - 465536

Data:

指定位置類型為 “Input Register” 或 “Output Register” 之 Tag 的資料型態。NAPOPC_CE5 共支援 5 種資料型態 - “Short”，”Long”，”Float”，”Word” 與 “DWord”。

資料類型	定義	範圍
Short	16 位元，有號數整數	-32768 ~ 32767
Long	32 位元，有號數整數	-2147483648 ~ 2147483647
Float	浮點數變數	-1.7E-308 ~ 1.7E+308
Word	16 位元，無號數整數	0 ~ 65535
DWord	32 位元，無號數整數	0 ~ 4294967295

“Input Coil” 或 “Output Coil” 的資料型態為 “Bool”。

Simulation:

有效訊號為 SINE，RAMP 與 RANDOM。當模組使用 “Simulation I/O” 功能時，此欄位才會生效。請參閱 "新增設備" 章節。

OK:

點選 "OK" 按鈕，新增此 Tag 設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

Scaling:

Enable:

勾選此項目，啟用 “Settings…” 按鈕。

Settings:

點選此按鈕， 設定 “Scaling” 功能。請參閱 “1.2.5.4 比例設定” 章節，以取得詳細資訊。

1.2.5.4 比例設定

一般而言， “Scaling” 功能僅供給 浮點數資料類型 使用。

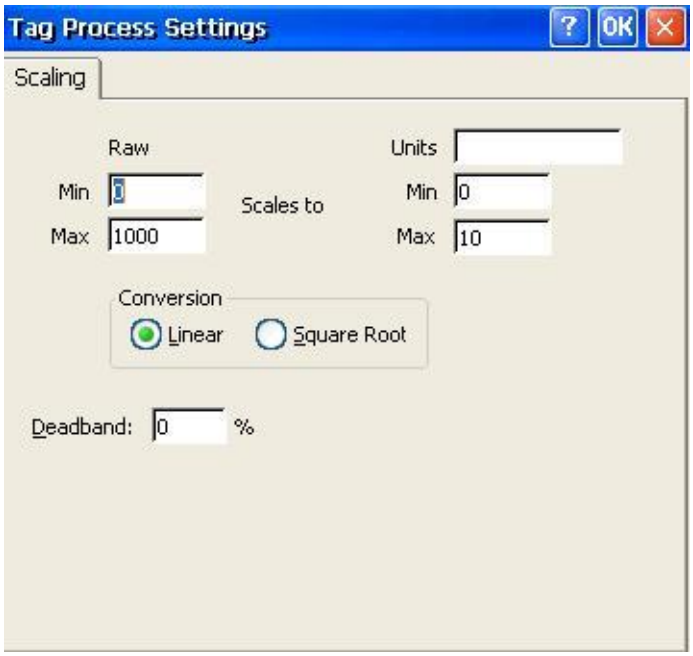


圖 1.2.5.4-1

Raw Data:

Min: 原始的最小值。([MinRaw])

Max: 原始的最大值。([MaxRaw])

Scales to:

Units: 縮放值的單位。 (僅為參考符號)

Min: 縮放比例的最小值。([MinScale])

Max: 縮放比例的最大值。([MaxScale])

Conversion:

Linear:

Scaled Value = ((Original Value – [MinRaw]) / ([MaxRaw] – [MinRaw]))
* ([MaxScale] – [MinScale]) + [MinScale]

Square Root:

Scaled Value = ((sqrt(Original Value) – [MinRaw]) * ([MaxScale] – [MinScale]))
/ sqrt([MaxRaw] – [MinRaw]) + [MinScale]

Deadband(%):

一般而言，此欄位為 "0"。固定區間只適用在群組中 dwEU 型態為 類比變數的項目。若 dwEU 型態是類比的，則此項目的 EU Low 與 EU High 可用來計算出其範圍。此範圍將依故定區間值成倍增加，並產生異常限制。異常狀態取決如下:

Exception if (absolute value of (last cached value - current value) >
PercentDeadband * (EU High –EU Low))

OK:

點選 "OK" 按鈕，儲存此設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

1.2.6 新增 Modbus 設備用的多重標籤

此功能僅供給 Modbus 設備使用。

步驟 1: 點選功能表 "Add/ Multi Tags" 。

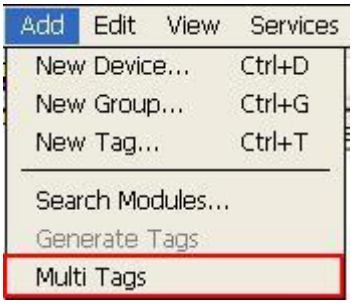


圖 1.2.6 -1

步驟 2: 彈出 "Add Multi Tags Dialog" 對話框。

步驟 3: 選取正確的 “Prototype” 與 “Data Type” 並輸入 Modbus 位址。

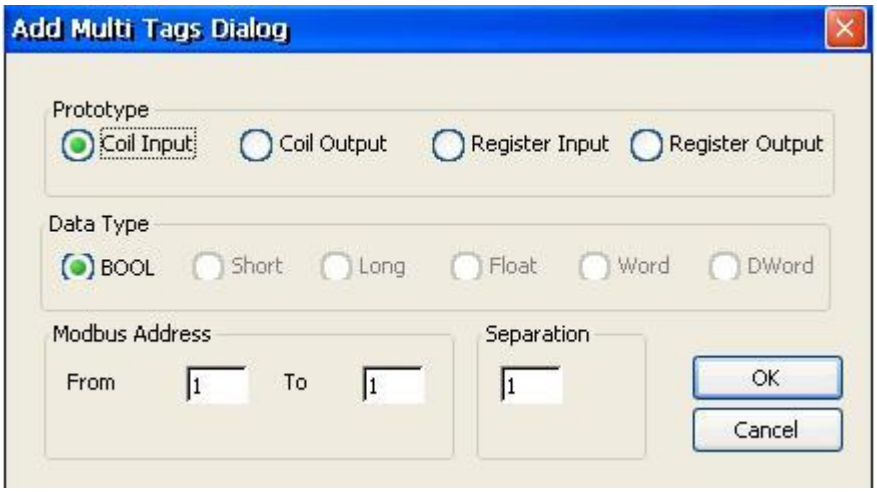


圖 1.2.6 -2

Prototype:

此 Modbus Tag 共有四種屬性 - “Coil Input”，“Coil Output”，“Register Input” 與 “Register Output” 。

Data Type:

“Bool” : 8 位元，True 或 False。

“Short” : 16 位元，-32768 ~ 32767。

“Long” : 32 位元，-2147483648 ~ 2147483647。

“Float” : 32 位元，浮點數。

“Word” : 16 位元，0 ~ 65535。

“DWORD” : 32 位元，0 ~ 4294967295。

Modbus Address:

“From” : 起始 Tag 的 Modbus 位址編號，1 ~ 65535。

“To” : 結束 Tag 的 Modbus 位址編號，1 ~ 65535。

Separation:

每個 Tag 的間隔數，1 ~ 100。

OK:

點選 "OK" 按鈕，新增此 Tag 設定。

Cancel:

點選 "Cancel" 按鈕，不做任何修改。

1.2.7 讀取/寫入 標籤

首先，您必須點選功能表 "View/ Monitor"，使用 "Monitor" 功能來監看 Tage 的值。選取一個 Tag 並按滑鼠右鍵，然後點選 "Properties.." 選項，再點選 “Read & Write” 頁籤來讀取或寫入 Tag。

步驟 1: 點選功能表 "View/ Monitor" 啟用監測功能。

步驟 2: 選取一個 Tag 並按滑鼠右鍵，然後點選 "Properties.." 選項。

步驟 3: 點選 “Read & Write” 頁籤。首先您會見到 “Tag name” 與 “Access right”。若 “Access right” 為 “Read only!”，表示寫入功能為關閉狀態。

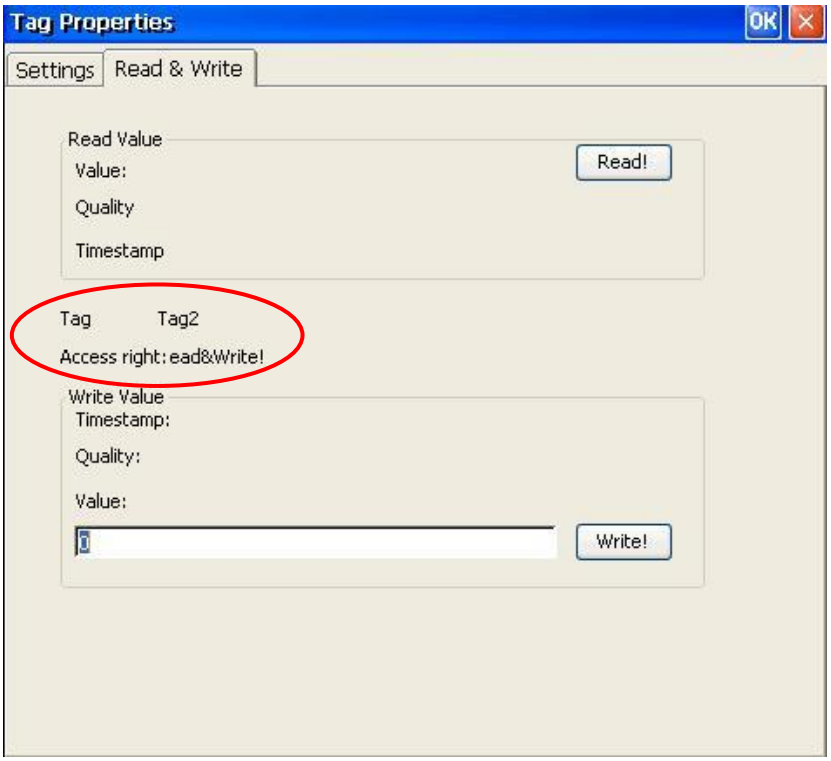


圖 1.2.7-1

Read Value/Value:

您可在 “Tag Properties” 視窗上點選 “Read!” 按鈕來讀取 Tag 的值。

Read Value/Quality:

此處將會顯示三種讀取品質 - “Good”，“Bad” 與 “Uncertain”。若通訊狀態良好，“Quality” 會顯示 “Good”。若通訊狀態有些錯誤，將顯示 “Bad”。而其它狀態則顯示 “Uncertain”。

Read Value/Timestamp:

此處將顯示讀取 Tag 的時間。

Tag name:

此處與 “Settings” 頁籤的 “Name” 相同。 您可在 “Settings” 頁籤修改名稱。

Access right:

共有兩種存取權限 - “Read Only!” 與 “Read&Write!” 。此存取權限取決於它是什麼類型的 Tag 屬性。請參閱 “1.6 新增標籤” 章節。

Write Value/Timestamp:

此處將顯示寫入 Tag 的時間。

Write Value/Quality:

此處將會顯示三種品質 - “Good”，”Bad” 與 “Uncertain”。若通訊狀態良好，“Quality” 會顯示 “Good”。若通訊狀態有些錯誤，將顯示 “Bad”。而其它狀態則顯示 “Uncertain”。

Write Value/Value:

您可點選 “Write!” 按鈕來將您輸入的值寫入 Tag。若 Tag 的資料型態是 “Boolean”。寫入值為 “0” 表示 “OFF”，寫入值 “非 0” 則表示 “ON”。

1.2.8 編輯 設備/群組/標籤 屬性

編輯原有的設備或群組，只需先選取該設備 (如下圖的 8024_4) 或群組 (如下圖的 AOs) 並選取功能表中的 "Properties..." 選項。

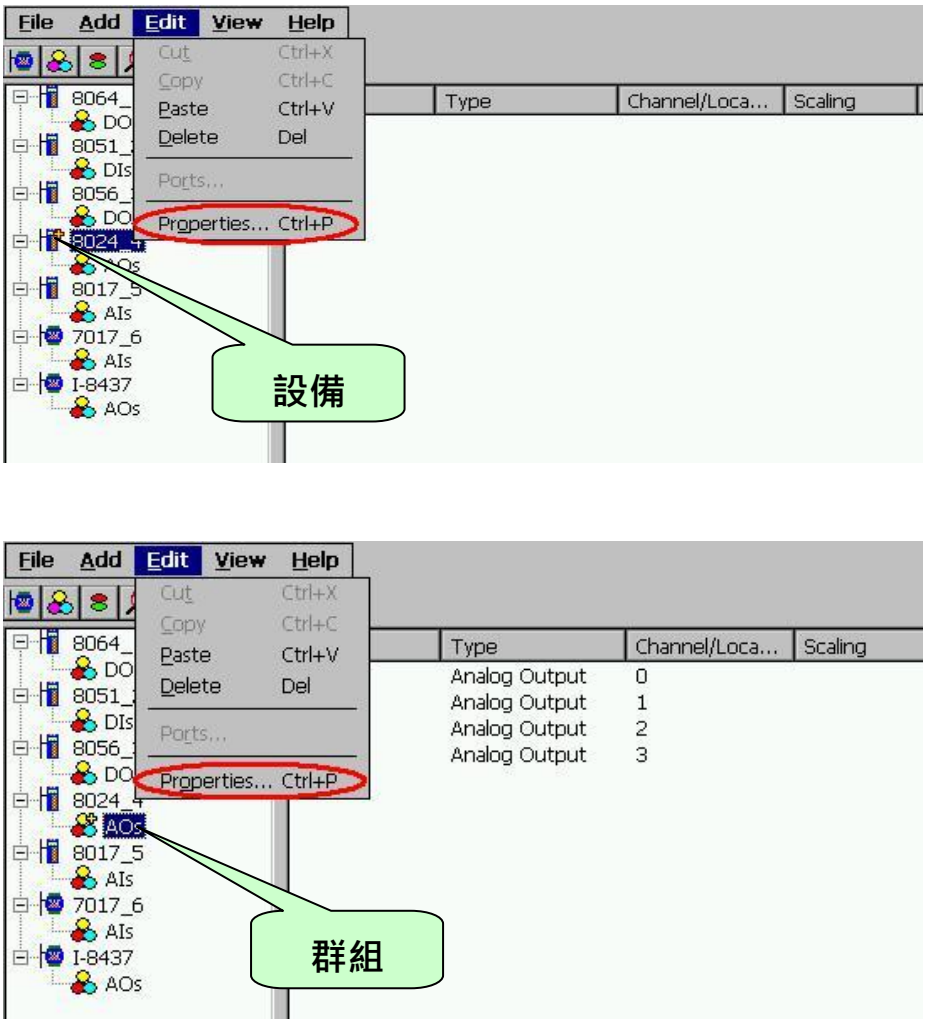


圖 1.2.8-1

編輯原有的 Tag，只需先選取該 Tag 並按滑鼠右鍵選取 "Properties..." 選項。

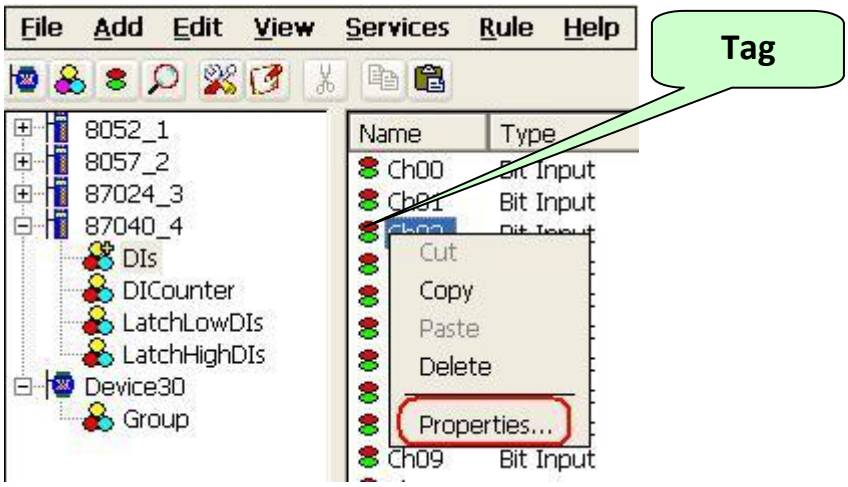


圖 1.2.8-2

1.2.9 刪除 設備/群組/標籤

欲刪除原有的設備/群組，只需先選取該設備/群組並選取 "Delete..." 選項。

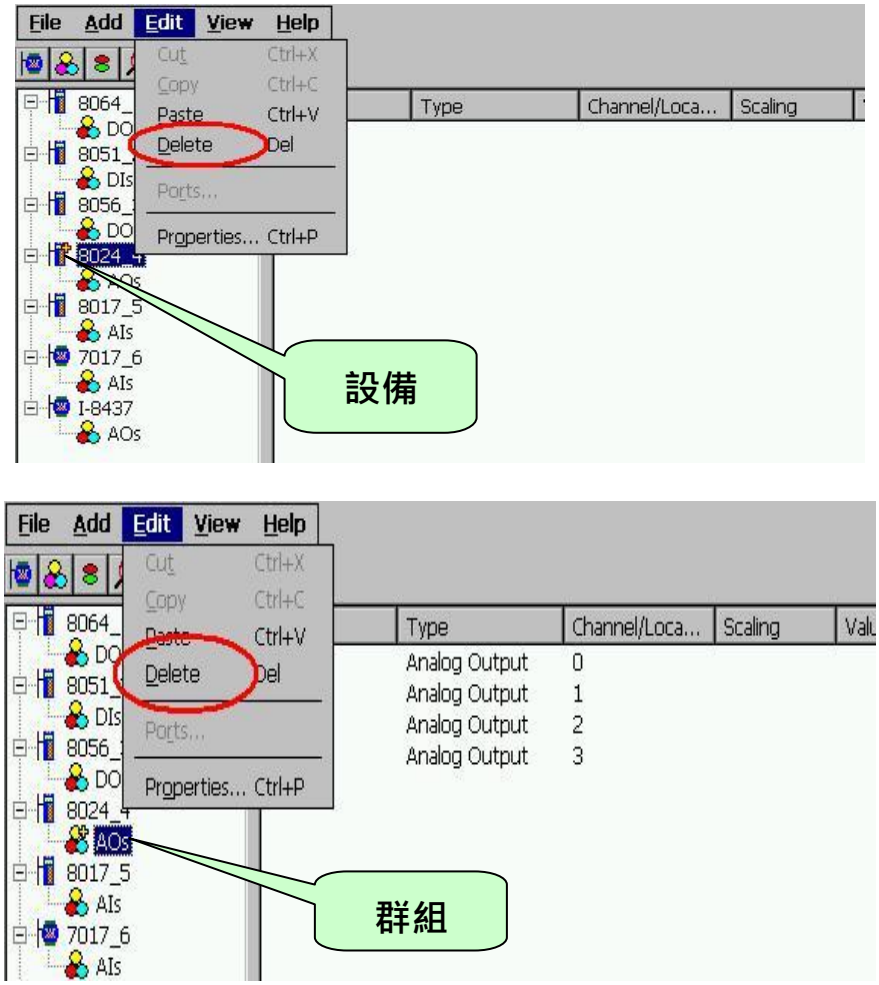


圖 1.2.9-1

欲刪除原有的 Tag，只需先選取該 Tag 並按滑鼠右鍵選取 "Delete..." 選項。

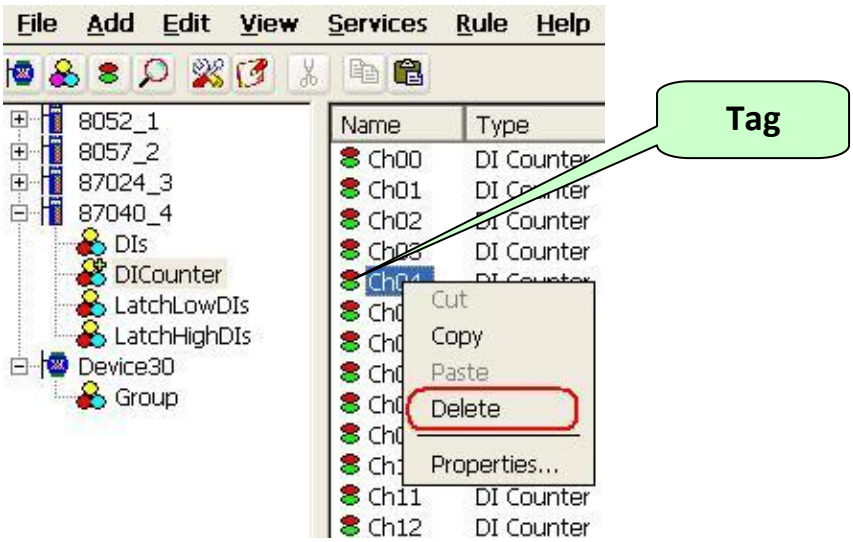


圖 1.2.9-2

1.2.10 建立 標籤

此功能可讓您輕鬆地在模擬模式下測試 NAPOPC_CE5。若所選取的模組類型設備沒有 附屬的 “Module”，"Group" 與 "Tag"，此功能才會生效。

步驟 1: 選取一個您欲產生 Tag 的模組類型設備。

步驟 2: 點選 "Add/ Generate Tags" 功能表。

圖 1.2.10-1

此處將會依據 Module-ID 產生 Tag，其可為 “Analog Input”，“Analog Output”，“Digital Input”， “Digital Output”， “Latched DI” 與 “Counter” 類型的 Tag。

1.2.11 服務設定

此功能可讓您定義，在與其它程式交換資料時需啟用哪些服務。NAPOPC_CE5 提供四種服務選擇 - “RPC Server”，”Modbus RTU”，”Modbus ASCII”，”Modbus TCP” 與 “Active ScanKernel”。其中，”RPC Server” 為一種允許 NAPOPC_ST/NAPOPC_XPE DA Server 透過 “遠端程序調用(RPC)” 的方式使用 NAPOPC_CE5 之機制。若您希望在 NAPOPC_ST/NAPOPC_XPE 端建立 “RPC” 設備，請在 NAPOPC_CE5 端勾選此選項。而 ”Modbus RTU”，”Modbus ASCII” 與 “Modbus TCP” 服務也都將在勾選後立即啟用。”Active ScanKernel” 服務將檢查所有的狀態，除了使用者應用程式之間的中介程式。

步驟 1: 點選功能表 "Services/Setup"。

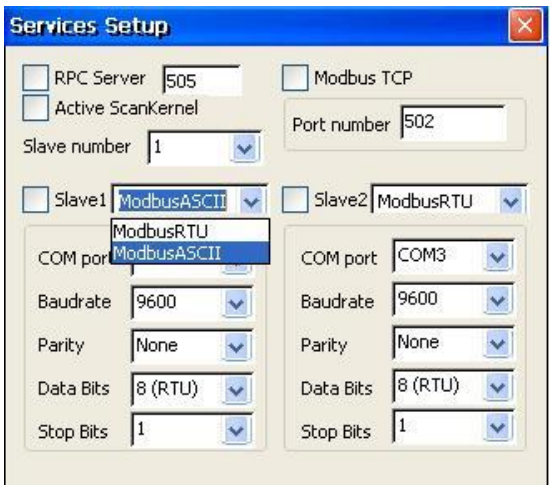


圖 1.2.11-1

步驟 2: 選取您所需的服務。

RPC Server Port:

此欄位必須設定為 “505” 以與 NAPOPC_ST 或 NAPOPC_XPE 進行通訊。

COM Port:

指定需使用的 COM Port。請檢視 RS-485 網路目前是使用哪個 COM Port 編號。而錯誤的設定將會造成模組通訊失敗。

Baudrate:

指定需使用的傳輸速率。請檢視模組的傳輸速率是正確的。而錯誤的設定將會造成模組通訊錯誤。

Parity:

指定需使用的同位元檢查格式。下列為其選項：

選項	說明
None	無同位元
Even	偶同位元
Odd	奇同位元

Data Bits:

指定傳送與接收一位元組 (Bytes) 資料，所包含的位元 (Bits) 數。

Stop Bits:

指定需使用的停止位元數。下列為其選項：

選項	說明
1	1 個停止位元
2	2 個停止位元
1.5	1.5 個停止位元

1.2.12 語法規則編輯器

此功能允許您透過 NAPOPC_CE5 來設計自己的基本規則，並讓您的 WinPAC-8000 成為一個 DCS 監控系統。NAPOPC_CE5 的基本規則，像是 “IF…THEN…”。在 “Rule Script Editor” 視窗左上角，”IF” 項目下有四個條件，其中變數是以 Modbus 位址來顯示並相互與 “AND/OR” 邏輯相結合。而在 “Rule Script Editor” 視窗右上角，“THEN” 項目下有四個輸出，其變數是以 Modbus 位址來顯示並相互與 “AND” 邏輯相結合。而 Timer 值與變數之間相關的邏輯為 “AND”。若 “IF” 下方的變數為 “0xxxxx” 或 “1xxxxx”，則 “Status”

會是 “0” 或 “1”。數值為 “0” 表示 “OFF”，“1” 表示 “ON”。若變數為 “3xxxxx” 或 “4xxxxx”，則 “Status” 將會依變數的資料型態而定。

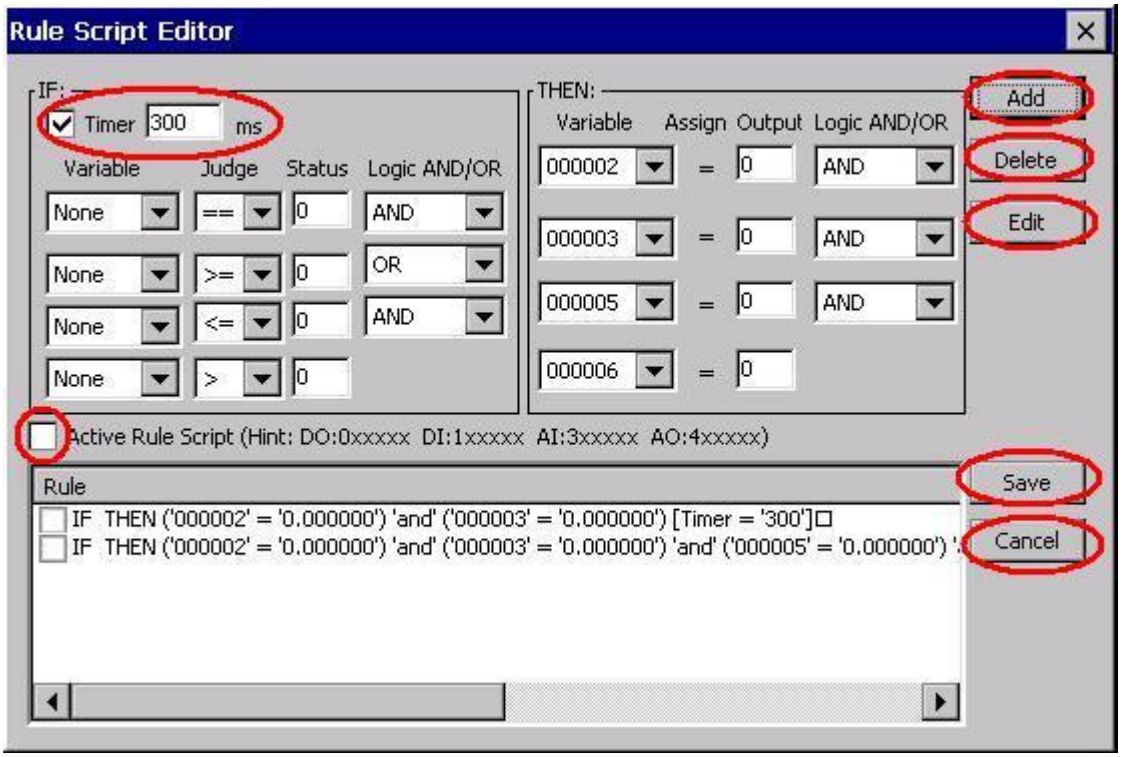


圖 1.2.12-1

Add:

編輯每項規則後，點選此按鈕將其加入規則清單 (Rule list)。

Delete:

於規則清單中檢視規則，並點選此按鈕將其刪除。

Edit:

於規則清單中點選該規則後，點選此按鈕更新。

Save:

完成編輯後，將規則清單存成 “Rule.txt” 。

Cancel:

離開此編輯畫面。

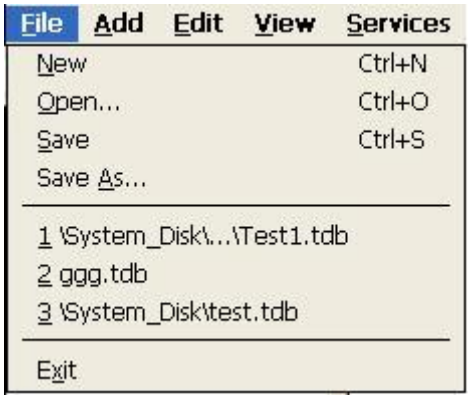
Active Rule Script:

勾選此選項後，將立即生效。若您希望於 NAPOPC_CE5 啟用後立即執行 “Rule Script”，您必須於功能表 “File/Save” 儲存檔案。

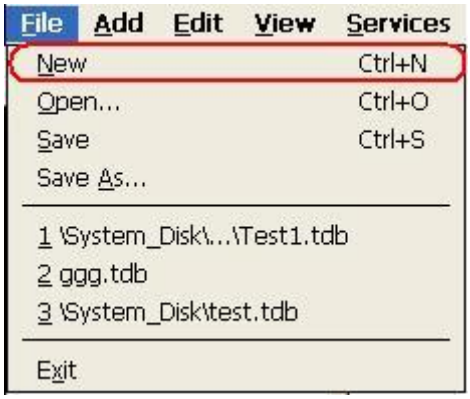
1.2.13 檔案

此功能可讓您儲存與載入 NAPOPC_CE5 設定。為了在 WinPAC-8000 啟動後可採用正確的 NAPOPC_CE5 設定檔 - “*.tdb”，您不僅需於 NAPOPC_CE5 中使用 “File/Save” 來儲存檔案，也需使用 “WinPAC Utility” 軟體中的 “Save and Reboot” 功能。

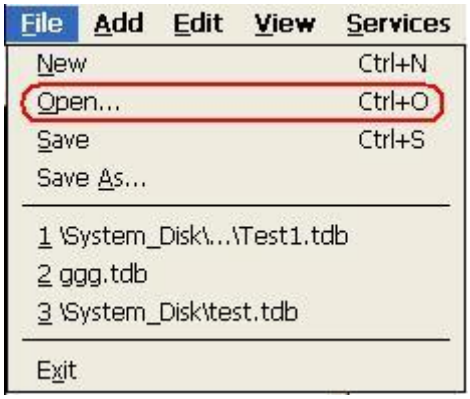
然後， NAPOPC_CE5 將會於每此啟動時，自動載入最後一個設定檔。



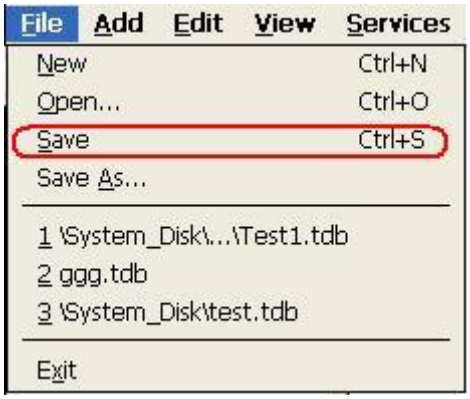
New: 清除目前專案並新建一個專案。



Open: 載入舊有 NAPOPC_CE5 專案。

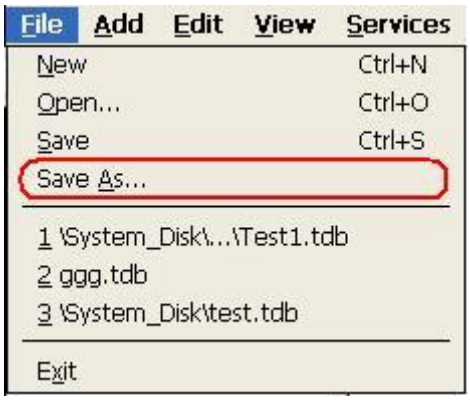


Save: 儲存目前的 NAPOPC_CE5 專案。



Save as...:

將 NAPOPC_CE5 專案另存新檔。



1.2.14 關於

點選功能表 "Help/ About NAPOPC_CE5" 可見到 "About NAPOPC_CE5" 視窗。它將顯示版本編號。

步驟 1: 點選 "Help/ About NAPOPC_CE5" 功能表。

步驟 2: 彈出 "About NAPOPC_CE5" 彈出視窗。

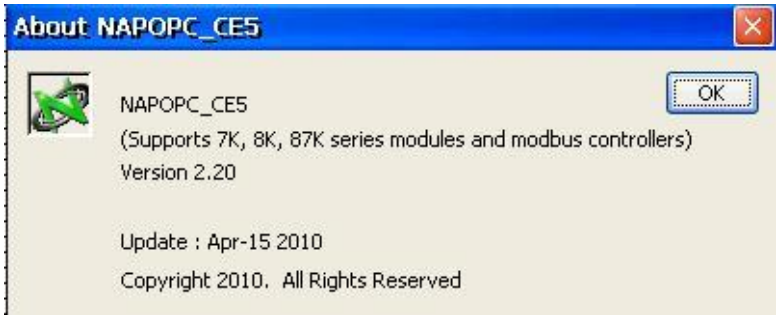


圖 1.2.14-1

1.2.15 最小化 NAPOPC_CE5

若您希望最小化 NAPOPC_CE5，請點選右上角的問號圖示。

圖 1.2.15-1

點選問號圖示後，NAPOPC_CE5 將會最小化至狀態列。您必需雙擊該圖示來回復它。

圖 1.2.15-2

2 快速上手

請依照下列步驟:

1. 連接模組 或 控制器。
   連接 RS-485 網路中的模組。
   連接控制器至 WinPAC-8000。
   (參閱 winpac8000_user_manual_v2.0.2.pdf)

2. 設定模組 或 控制器。
   使用 DCON Utility 軟體來設定模組。
   (參閱 winpac8000_user_manual_v2.0.2.pdf)

3. 運行 NAPOPC_CE5 軟體。
   執行 "NAPOPCSvr_CE5.exe" 或 "NAPCOP_CE5Boot.exe" 檔案來啟動 NAPOPC_CE5。

4. 搜尋模組。
   參閱 "1.2.1 搜尋模組" 章節來搜尋模組。

5. 新增控制器。
   參閱 “1.2.3 新增設備” 章節來新增 Modbus RTU 或 Modbus TCP 控制器。

6. 儲存設定。
   參閱 “1.2.13 檔案” 章節來儲存設定。

7. 關閉 NAPOPC_CE5。
   點選功能表 "File/Exit" 來關閉。

3 遠端存取

OPC Client 可有兩種方式來存取 OPC Server。一種稱為 “本區存取”，另一種稱為 “遠端存取”。若 OPC Client 與 OPC Server 同在一台電腦中，我們可稱此種架構為 “本區存取”。換言之，若 OPC Client 需透過網路來存取 OPC Server，我們稱此種架構為 “遠端存取”。

下圖中顯示了包含 “本區存取” 與 “遠端存取” 的整合架構。在實際的工業程序中，此兩種方式經常同時被使用。在程序管理層中，我們常使用 “本區存取” 的架構來監測與控制製造過程。在商業管理層中，我們只設立 OPC Client 來收集程序管理層的處理資訊。若您只需建立 “本區存取” 架構，則無需閱讀此章節。若您想建立 “遠端存取” 架構，您必須知道如何設定 OPC Client 與 OPC Server 之間的 DCOM 機制。

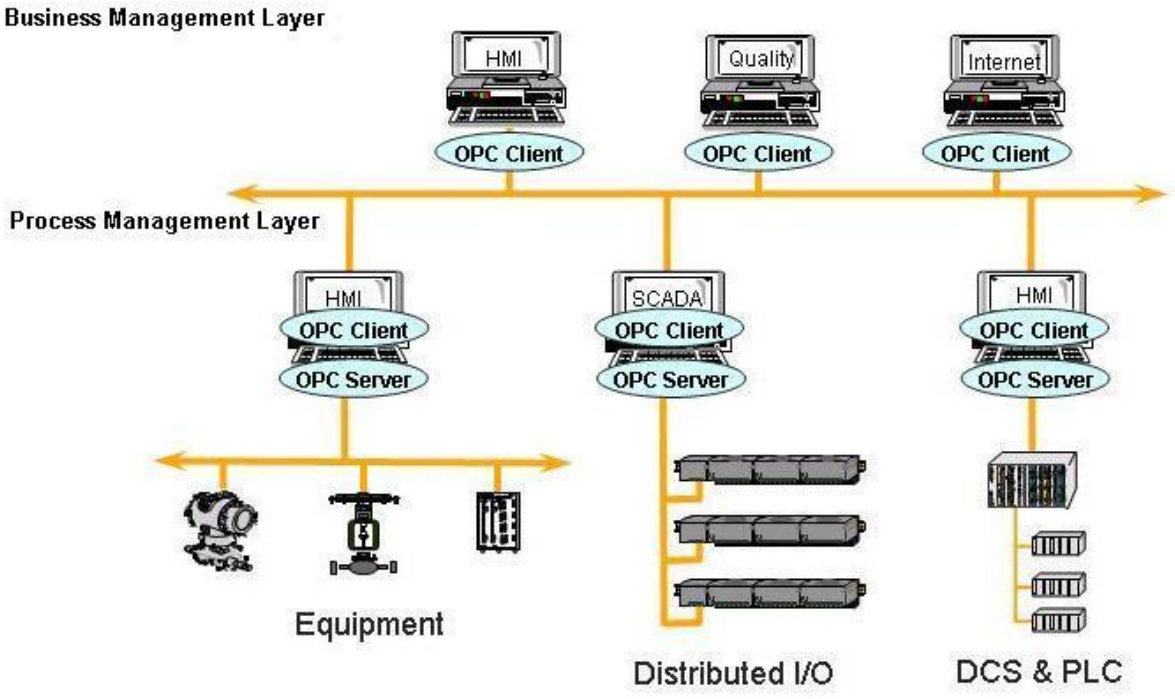


圖 3-0-1 本區存取 與 近端存取 架構

3.1 系統需求

為了透過網路來存取遠端的 OPC Server，您必須啟用位於 Client 與 Server 兩方的 DCOM 機制。

從客戶端的電腦無法啟動 Windows 95 電腦上的安全處理程序。在 Windows 95 中，其依循當前登入用戶的安全設定檔 (security context) 來執行所有的處理程序; 因此，Windows 95 上的 DCOM 機制並不支援遠端啟動功能。必需以手動方式來啟動 Windows 95 電腦上的 Server 應用程式 或 透過一些其它機制，由另一台電腦的 Client 應用程式來進行存取。因此，"DefaultLaunchPermissions" 與 "LaunchPermissions" 註冊值在 Windows 95 上也無效。

使用平台	此平台是否支援 DCOM?
Windows 95	否。 使用者必須由微軟網站下載與安裝 DCOM95.EXE 和 DCM95CFG.EXE 來啟用遠端存取功能。
Windows 98	是。 Windows 98 支援 DCOM 機制。建議更新 DCOM98 至最新版本。可至微軟網站取得最新版本。
Windows NT 4.0	是。 Windows NT 4.0 支援 DCOM 機制。 建議更新 Windows NT 4.0 的 Service Pack 至最新版本。 (Service Pack 3 或 更新版本)。
Windows 2000	是。 Windows 2000 支援 DCOM 機制。
Windows XP	是。 Windows XP 支援 DCOM 機制。

3.2 設定 DCOM

變更之前，需在 Client 與 Server 電腦的註冊表中註冊 Server 應用程式。這包含了執行 Server 端應用程式的安裝程式 或 執行 Server 端應用程式，然後關閉此兩台電腦上的應用程式。Server 端應用程式並不需駐留在 Client 端電腦上。

若 Server 端使用自定介面，編組處理碼 (Marshaling Code) 需安裝在 Client 與 Server 端電腦。而支援 "vtbl-binding" 的自動化 Server，需在 Client 與 Server 端電腦中安裝其型別程式庫。不支援 "vtbl-binding" 的自動化 Server，不需在 Client 端電腦中安裝型別程式庫。

更改註冊表後，執行 Client 端電腦上的 Client 應用程式。DCOM 機制會於 Client 端電腦上，檢視 Server 應用程式的註冊表項目並確定 Server 端電腦的名稱。然後，它將連線到 Server 端電腦，採用其註冊表來確定 Server 應用程式的位置並啟動該電腦的 Server 應用程式。

您可使用 DCOMCnfg.exe 工具 (一種 OLE 檢視器工具) 來更改註冊表，或以手動方式來更改。 關於使用 OLE 檢視器 或 手動更改之詳細資訊，請參閱微軟網站上，"Q158582, HOWTO: Configure a Non-DCOM Server and Client to Use DCOM" 的文章。關於使用 DCOMCnfg.exe 來設定 DCOM 之詳細資訊，請參閱微軟網站上，主題為 "Inside Distributed COM"，於 1998 年，由 Guy Eddon 與 Henry Eddon 所寫的文章。

此章節將說明如何在 Client 與 Server 端電腦上，使用 DCOMCnfg.exe 圖形驅動式工具 (可見於 Windows NT system32 目錄 或 Windows95/98 system 目錄) 來設定 DCOM 的狀態。

下表顯示了三種和 WinPAC 相關的 DCOM 設定組合。您可看到 WinPAC 可以是 Client 與 Server 端。但針對 XPAC 和 PC 而言，WinPAC 只能是 Server 端。此項限制是基於 DCOM 的安全性機制。由於 PC 上有太多種作業系統 (OS)，我們將選用 Windows XP 為範本來設定 DCOM。您也可以使用其它微軟的作業系統。

Client 端	Server 端
PC (NAPOPC_ST Server)	WinPAC (NAPOPC_CE5 Server)
XPAC (NAPOPC_XPE Server)	WinPAC (NAPOPC_CE5 Server)
WinPAC (NAPOPC_CE5 Server)	WinPAC (NAPOPC_CE5 Server)

3.2.1 設定 Server 端 (WinPAC)

系統需求

OS 版本:

WinPAC OS 1.3.04 或 更新版本。

程式:

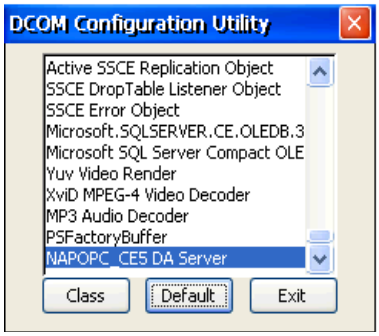
NAPOPC_CE5
DCOMCnfg.exe

WinPAC Utility 2.0.2.1 或 更新版本

設定 DCOM

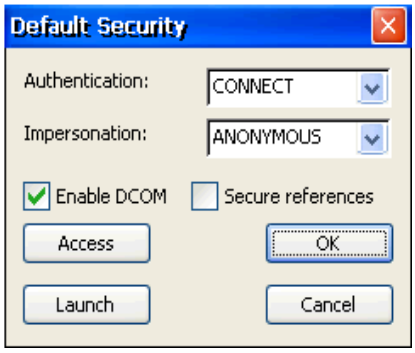
步驟 1: 執行 \\NAPOPC_CE5\napopc_ce5boot.exe 程式來註冊。

步驟 2: 執行 dcomcnfg.exe 程式並點選 “Default”。

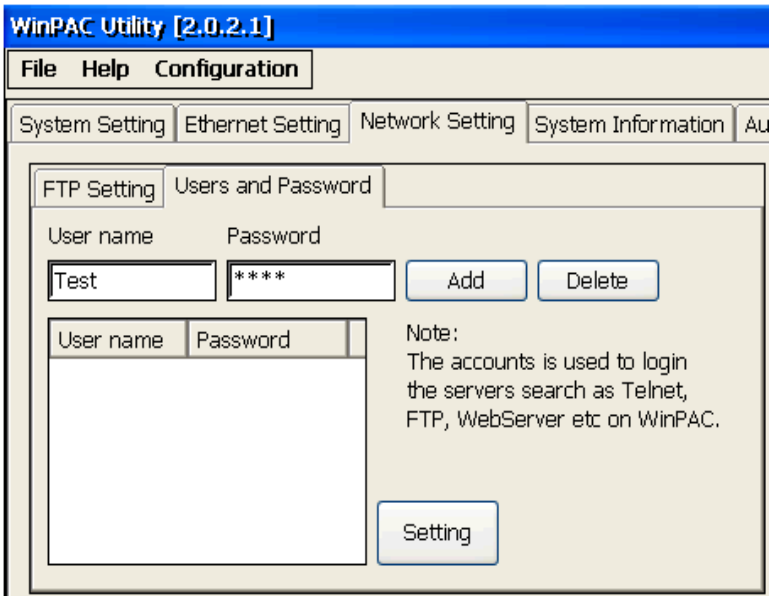


步驟 3: 點選 “Access” 按鈕，來加入目前已與 Client 網站連線的(存取權限)帳戶。

步驟 4: 點選 “Launch” 按鈕，來加入目前已與 Client 網站連線的(啟動權限)帳戶。



步驟 5: 執行 “WinPAC Utility->Network Setting->Users and Password”



步驟 6: 輸入 “User name”，“Password” 並點選 “Add”。“User name” 與 “Password” 必須是 **步驟 3** 所設定的帳戶名稱。最後，點選 “Setting” 來完成設定。

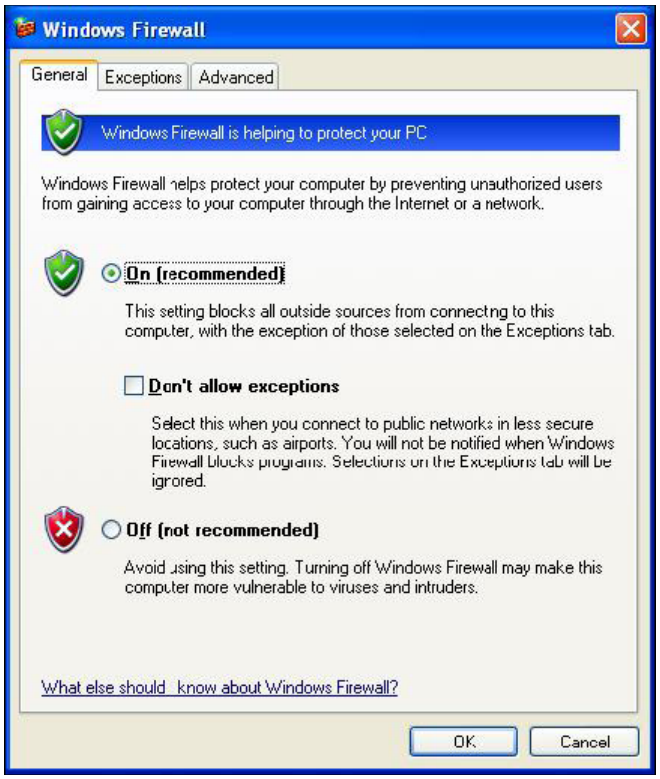
步驟 7: 執行 WinPAC Utility 中的 “save and reboot” 來儲存設定並重新啟動。

3.2.2 設定 Client 端 (PC)

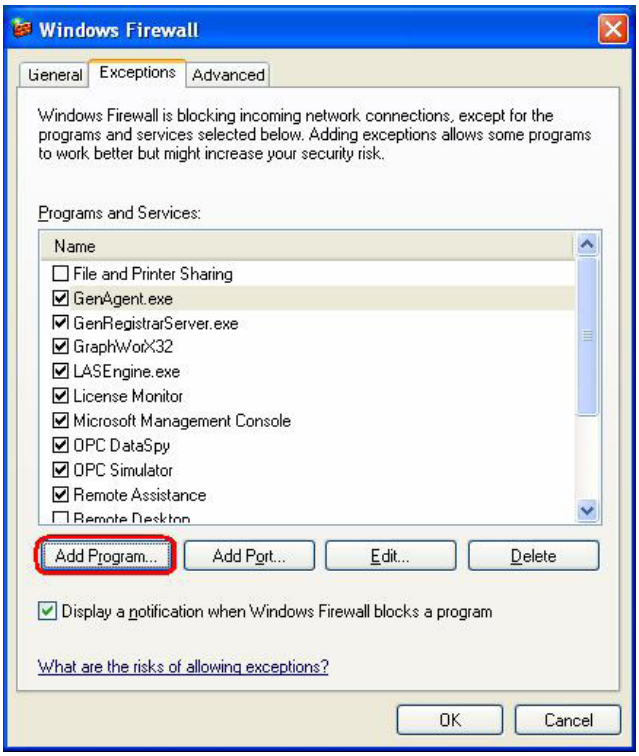
設定防火牆

步驟 1: Windows 中預設的防火牆設定為 “開啟”。此為微軟公司與 OPC 所建議的設置，以讓您的機器擁有最大可能的防護。就故障排除而言，您可能希望暫時地關閉防火牆來驗證 或 排除 任何通訊失敗是起因於防火牆設定之可能性。

注意: 若這台機器於企業防火牆之後，已獲得充分的防護，則永久關閉防火牆是適當的選擇。關閉後，則不需執行後續之個人防火牆設定來允許 OPC 通訊。

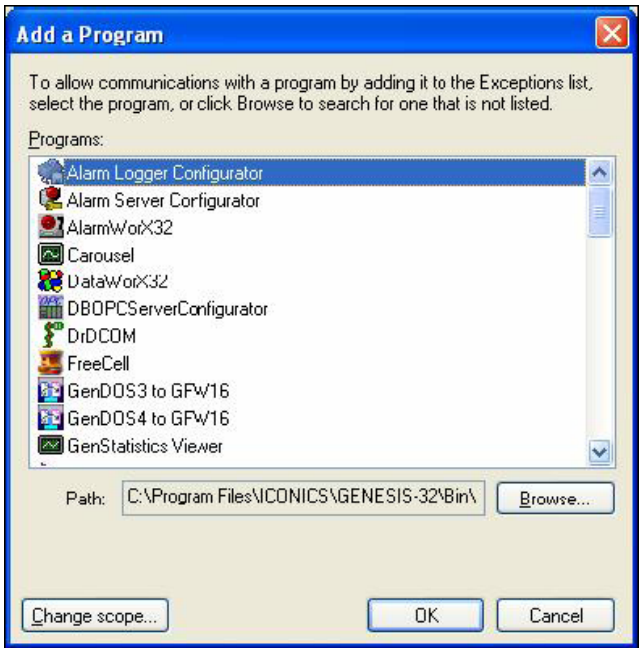


步驟 2: 如下圖，選取 “Exceptions” 頁籤並於例外清單中加入所有的 “OPC Client” 與 “OPC Server”，也加入 “Microsoft Management Console” (於下一章節中，用在 DCOM 設定工具) 與 OPC 工具 (OPCEnum.exe，位於 Windows\System32 目錄)。

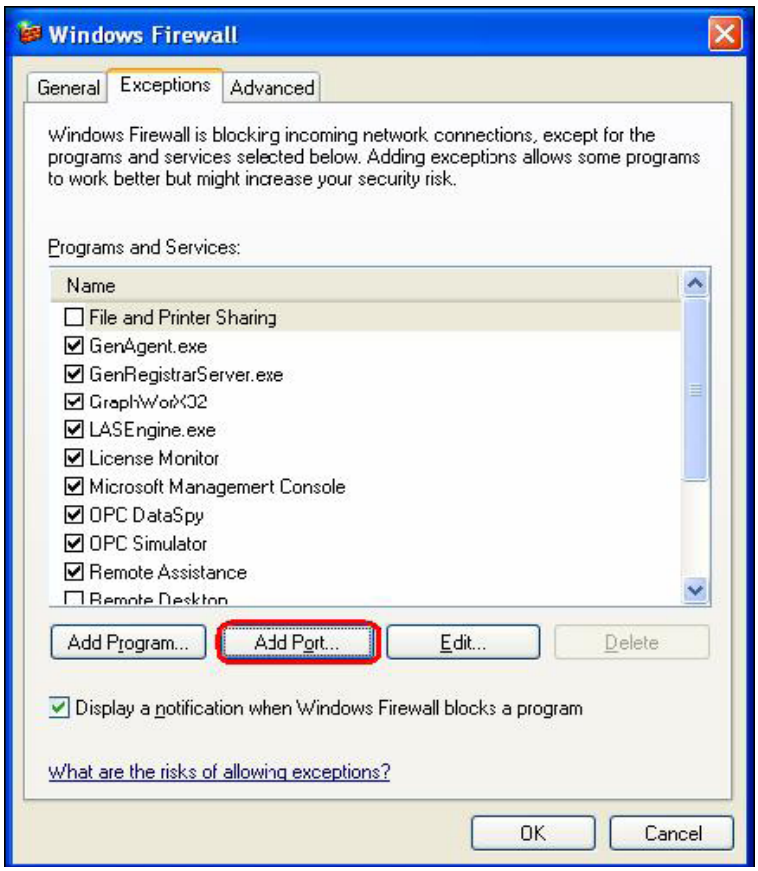


如下圖，在 “Add a Program” 對話框中，列出大多數電腦上的應用程式，但請注意並非所有的應用程式皆顯示於此，點選 “Browse” 按鈕來找到其它安裝於電腦的執行檔。

注意: 只有執行檔 (.exe) 需加入至例外清單。對於行程內的 OPC Server 與 Client (DLLs 與 OCXs)，您需要加入該應用程式執行檔，進而調用這些程式於清單中。

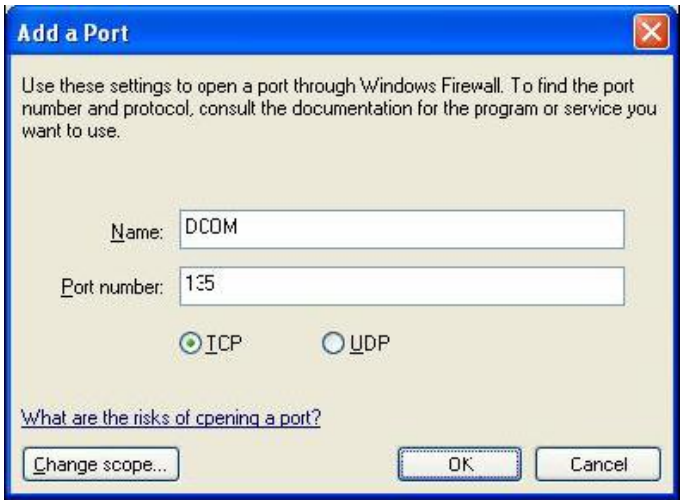


步驟 3: 加入TCP Port: 135 以允許啟動 DCOM 通訊，並傳入回應需求。於 Windows 防火牆設定的 “Exceptions” 頁籤，點選 “Add Port” 按鈕。



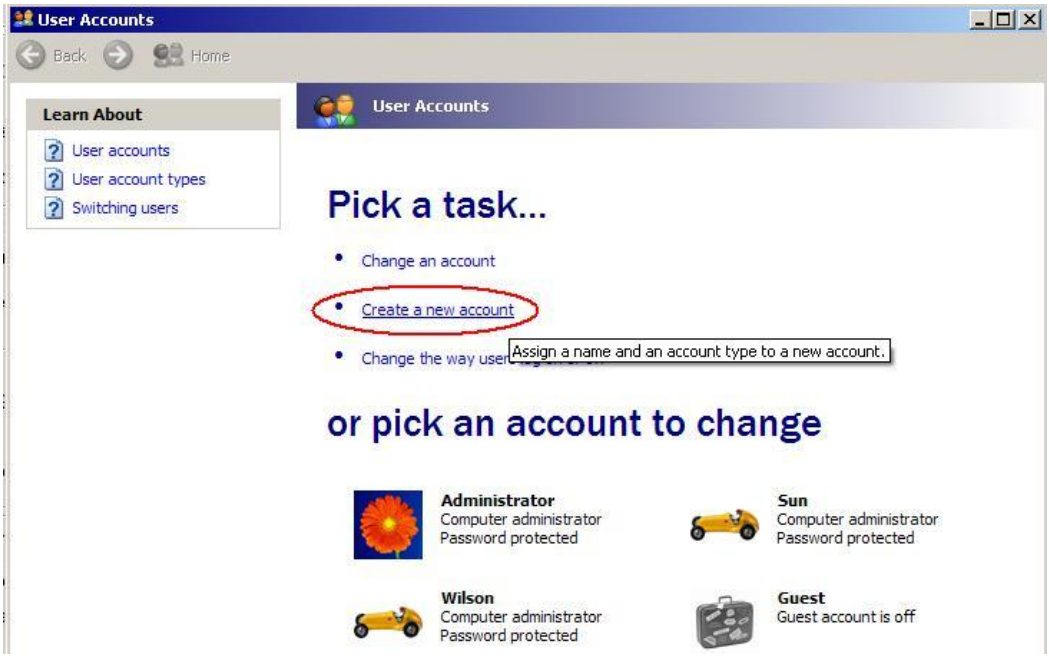
在 “Add a Port” 對話框中，輸入以下欄位後，再點選 TCP 單選按鈕:

Name: DCOM
Port number: 135



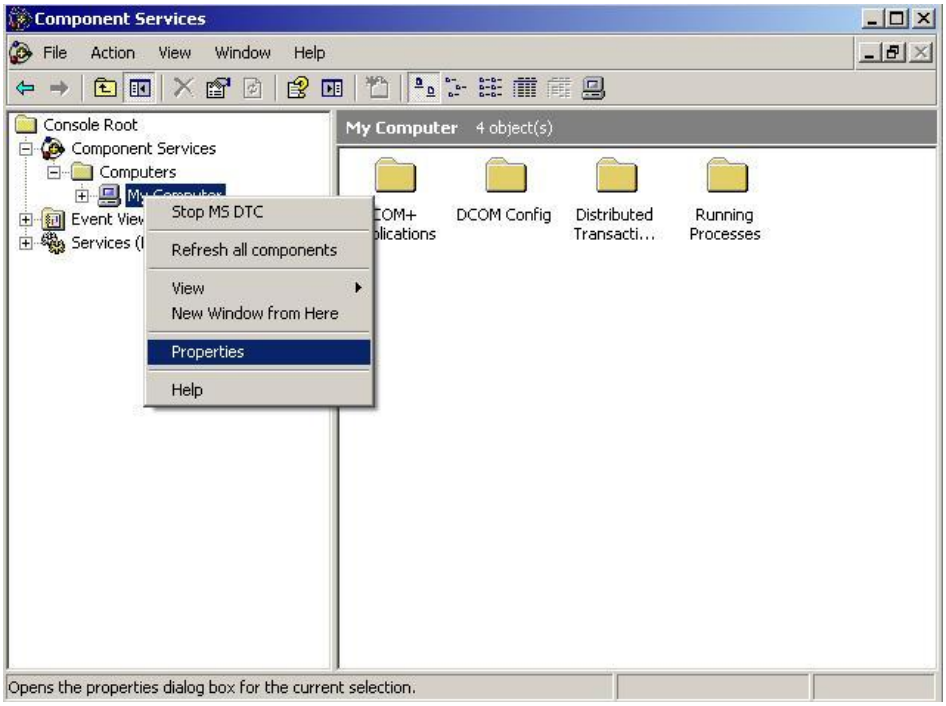
建立使用者帳戶

步驟 1: 建立一個帳戶，此帳戶必須與 Server 端的帳戶名稱相同。



設定 DCOM

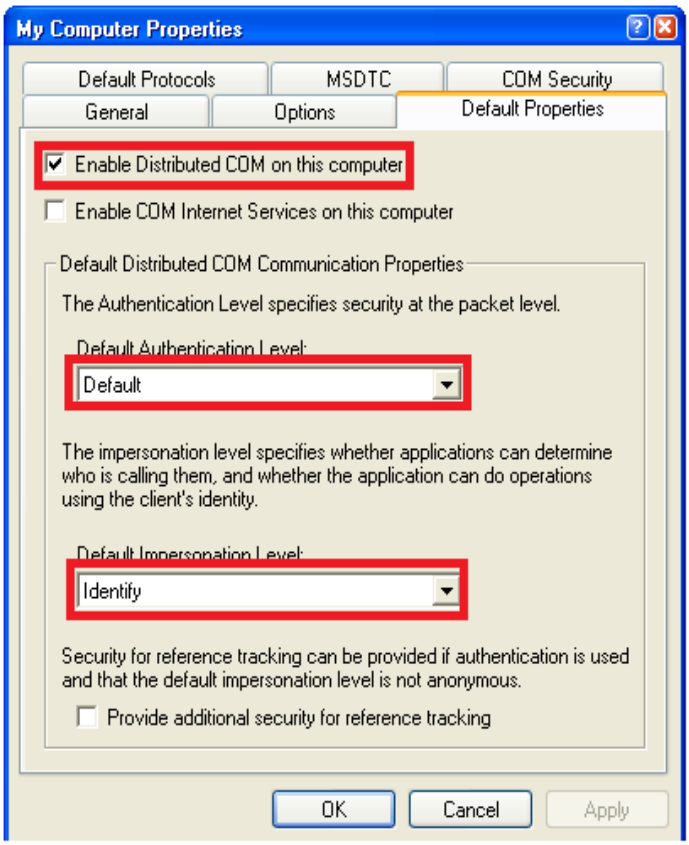
步驟 1: 執行 dcomcnfg.exe 程式來啟動元件服務。於 “My Computer” 按滑鼠右鍵並點選 “Properties”。



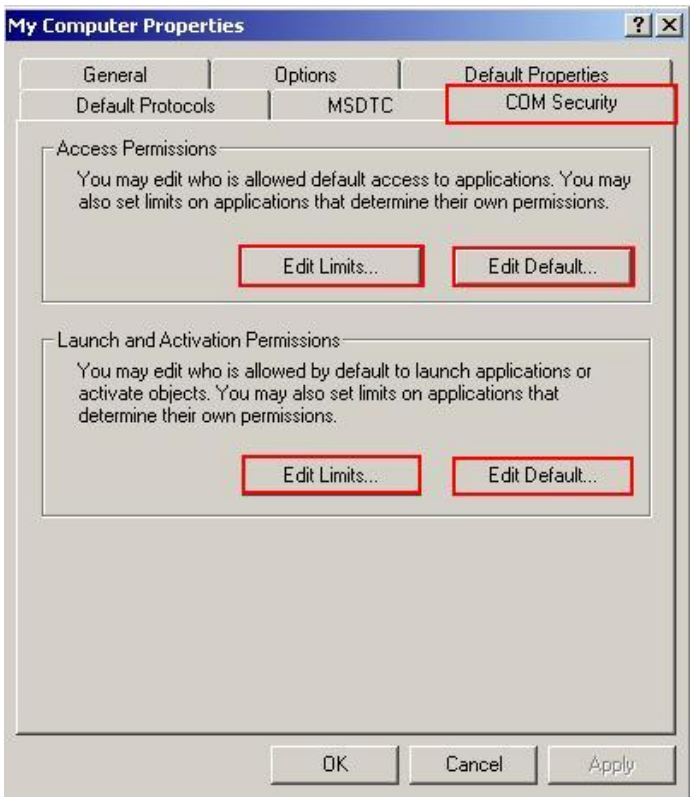
步驟 2: 點選 "Default Properties" 頁籤。

步驟 3: 使用以下設定:

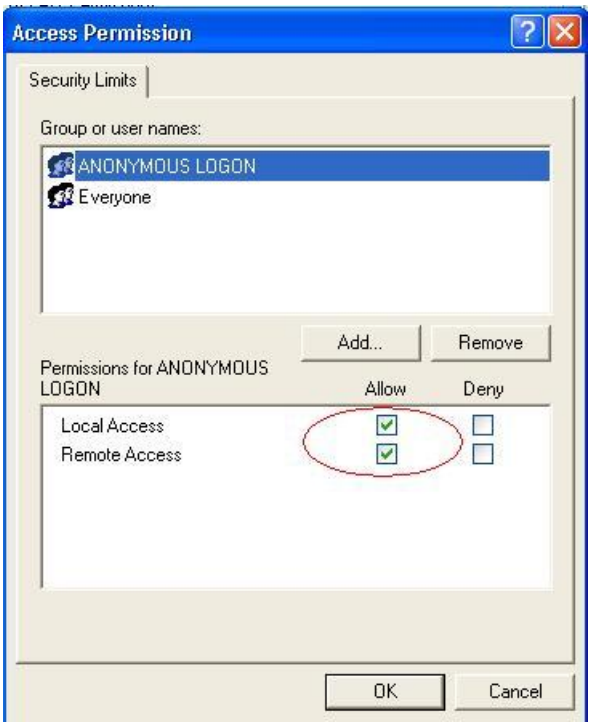
欄位名稱	設置
Enable Distributed COM on this computer	勾選
Default Authentication Level:	Default
Default Impersonation Level:	Identify

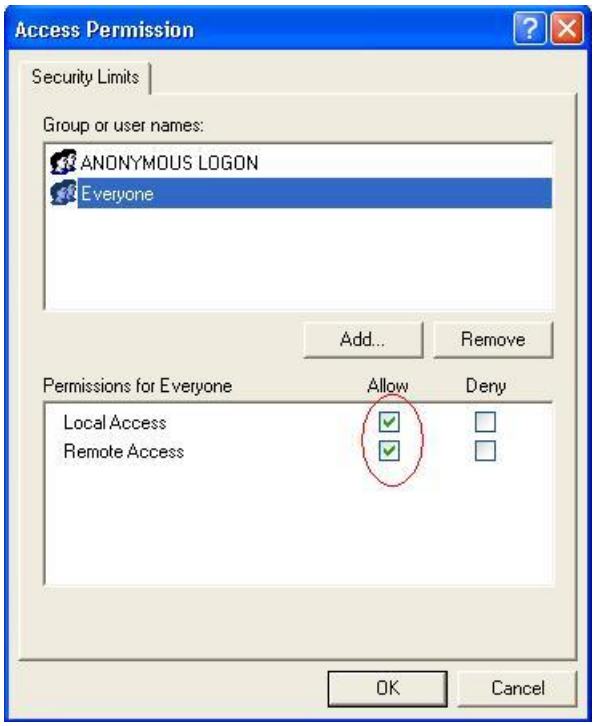


步驟 4: 點選 "COM Security" 頁籤。

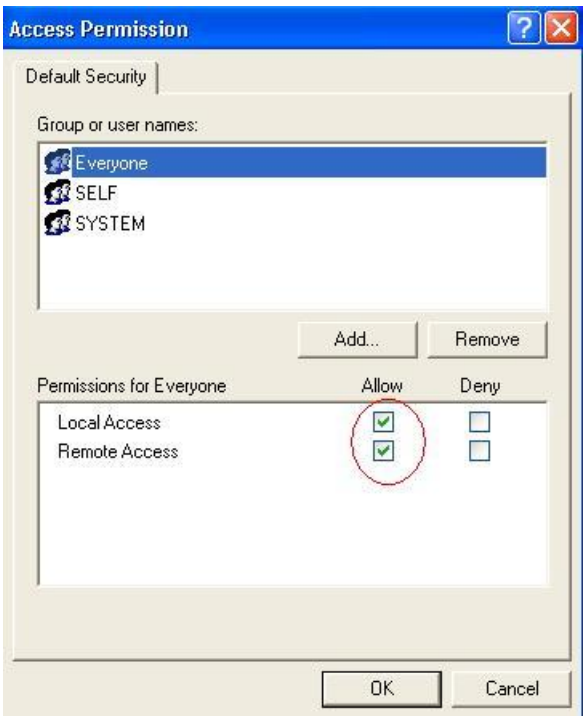


步驟 5: 如上圖，點選 “Access Permissions” 項目的 “Edit Limits…” 按鈕進入設定。

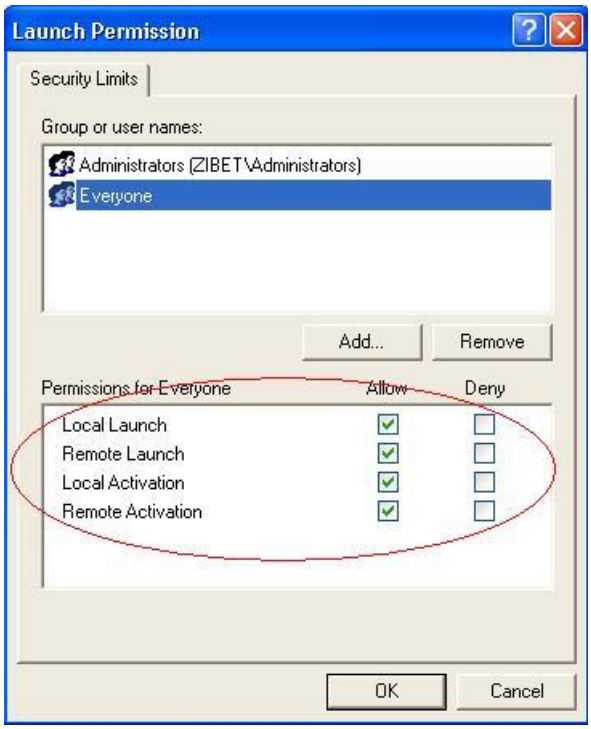




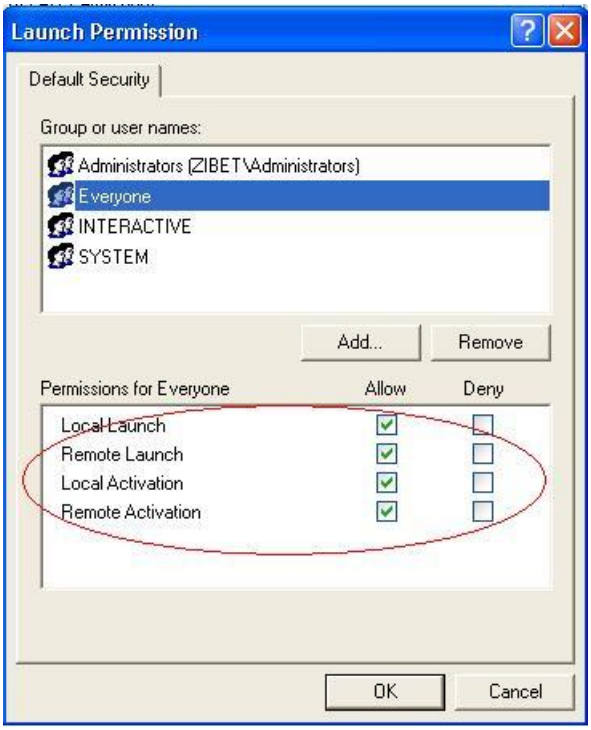
步驟 6: 如上述，點選 “Access Permissions” 項目的 “Edit Default…” 按鈕進入設定。



步驟 7: 如上述，點選 “Launch and Activation Permissions” 項目的 “Edit Limits…” 按鈕進入設定。

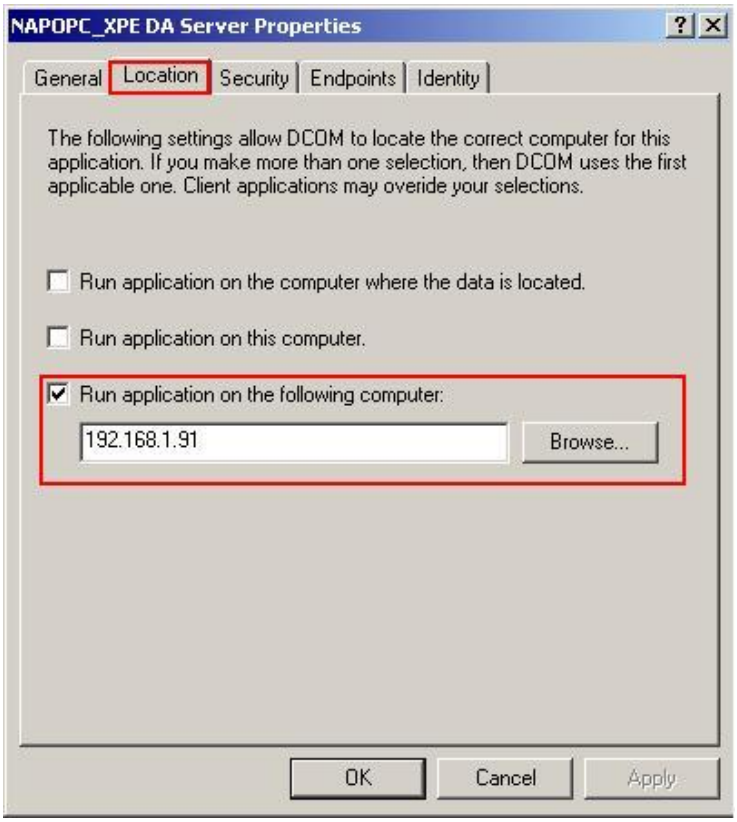


步驟 8: 如上述，點選 “Launch and Activation Permissions” 項目的 “Edit Default…” 按鈕進入設定。

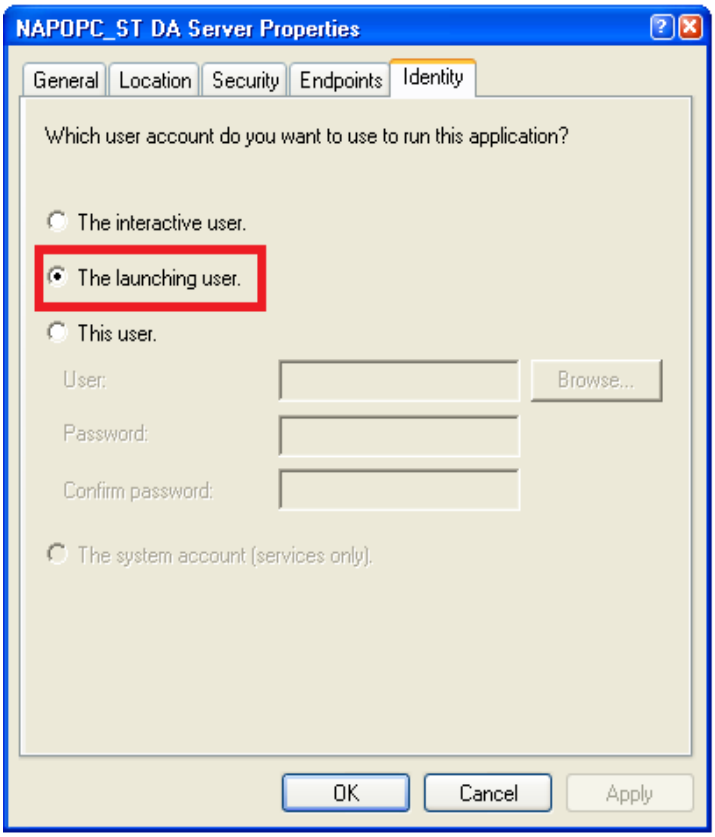


步驟 9: 如下圖，於 “DCOM Config” 中的 “NAPOPC DA Server” 按滑鼠右鍵並點選 “Properties”。

步驟 10: 點選 "Location" 頁籤，勾選 “Run application on the following computer” 並輸入 Server 的 IP 位址。



步驟 11: 點選 "Identity" 頁籤並選取 “The launching user” 項目。



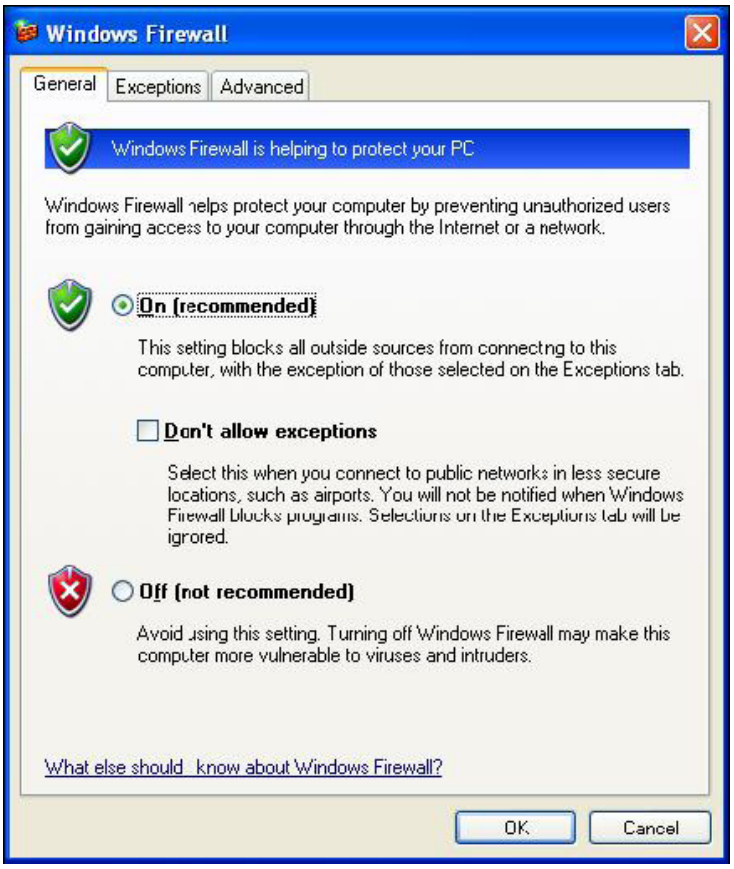
步驟 12: 重新啟動電腦。

3.2.3 設定 Client 端 (XPAC)

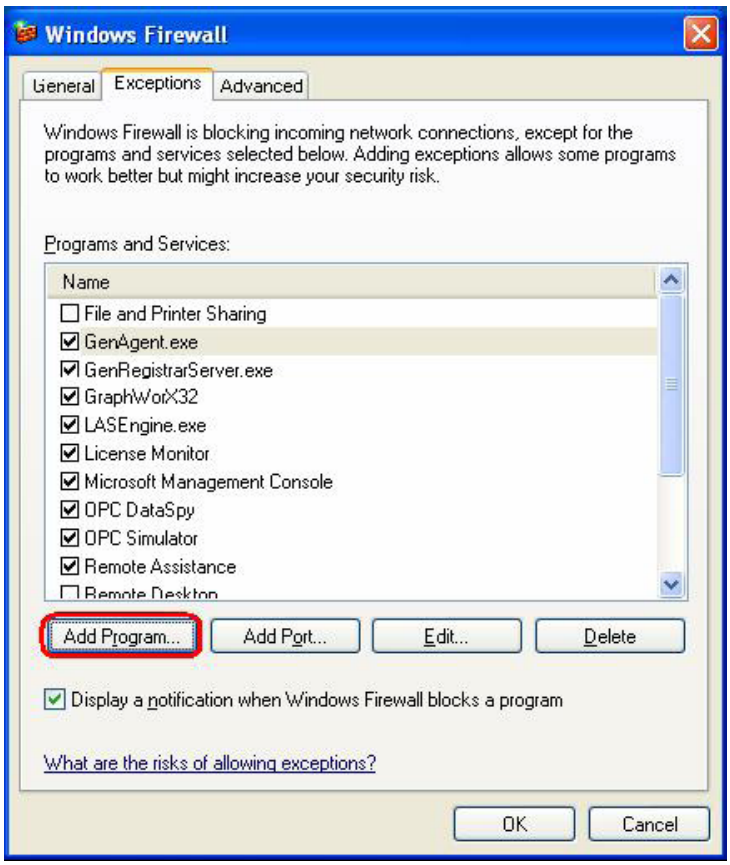
設定防火牆

步驟 1: Windows 中預設的防火牆設定為 “開啟”。此為微軟公司與 OPC 所建議的設置，以讓您的機器擁有最大可能的防護。就故障排除而言，您可能希望暫時地關閉防火牆來驗證 或 排除 任何通訊失敗是起因於防火牆設定之可能性。

注意: 若這台機器於企業防火牆之後，已獲得充分的防護，則永久關閉防火牆是適當的選擇。關閉後，則不需執行後續之個人防火牆設定來允許 OPC 通訊。

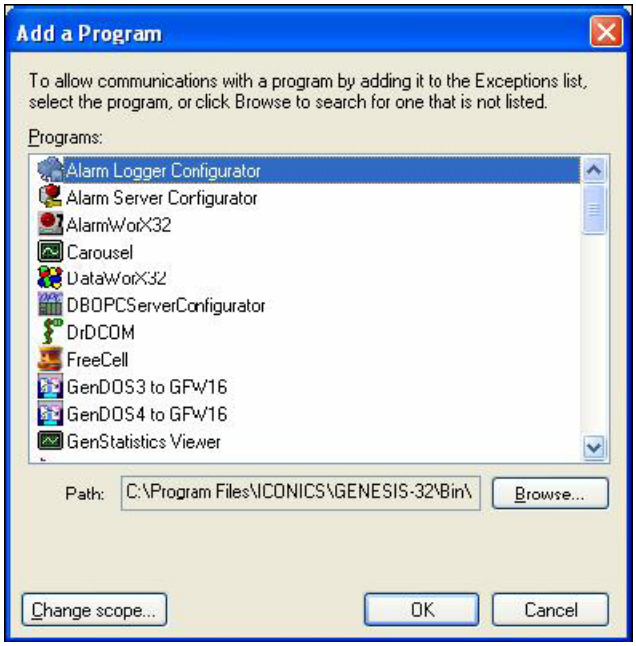


步驟 2: 選取 “Exceptions” 頁籤並於例外清單中加入所有的 “OPC Client” 與 “OPC Server”，也加入 “Microsoft Management Console” (下一章節中，用在 DCOM 設定工具) 與 OPC 工具 (OPCEnum.exe，位於 Windows\System32 目錄中)。

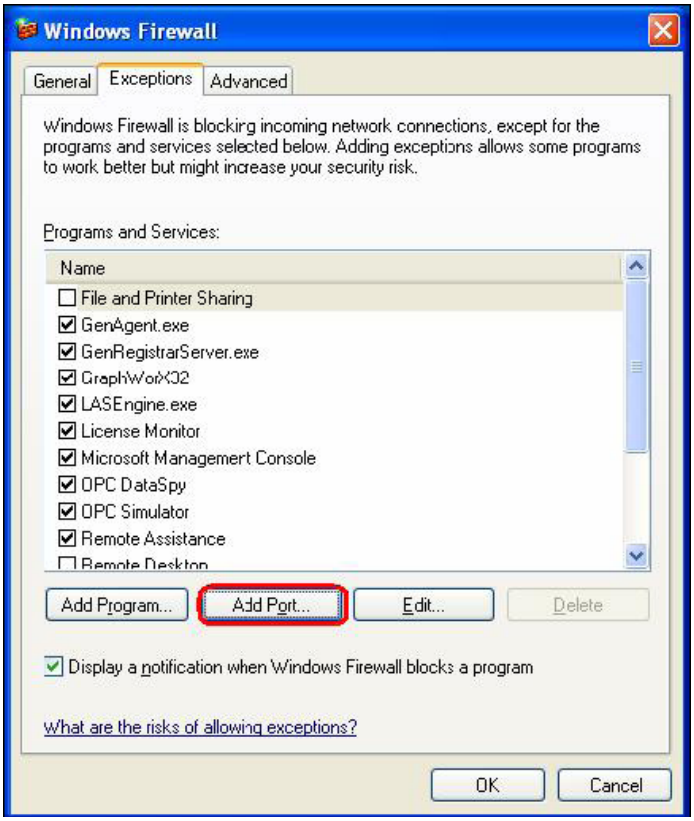


如下圖，在 “Add a Program” 對話框中，列出大多數電腦上的應用程式，但請注意並非所有的應用程式皆顯示於此，點選 “Browse” 按鈕來找到其它安裝於電腦的執行檔。

注意: 只有執行檔 (.exe) 需加入例外清單。對於行程內的 OPC Server 與 Client (DLLs 與 OCXs)，您需要加入該應用程式執行檔，進而調用這些程式於清單中。

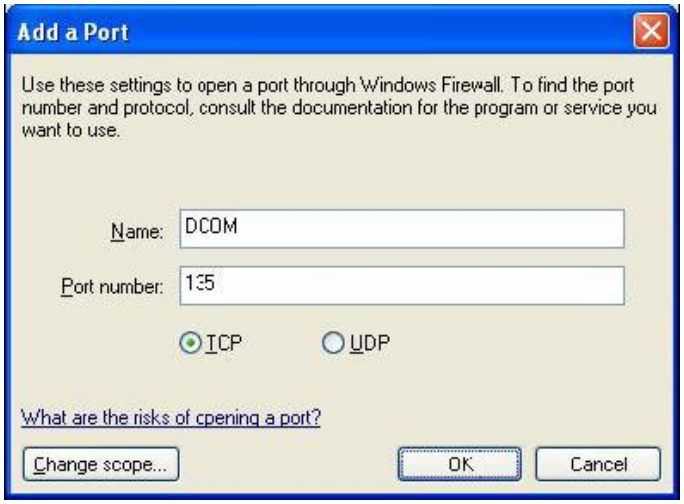


步驟 3: 加入TCP Port: 135 以允許啟動 DCOM 通訊，並傳入回應需求。於 Windows 防火牆 設定的 “Exceptions” 頁籤，點選 “Add Port” 按鈕。



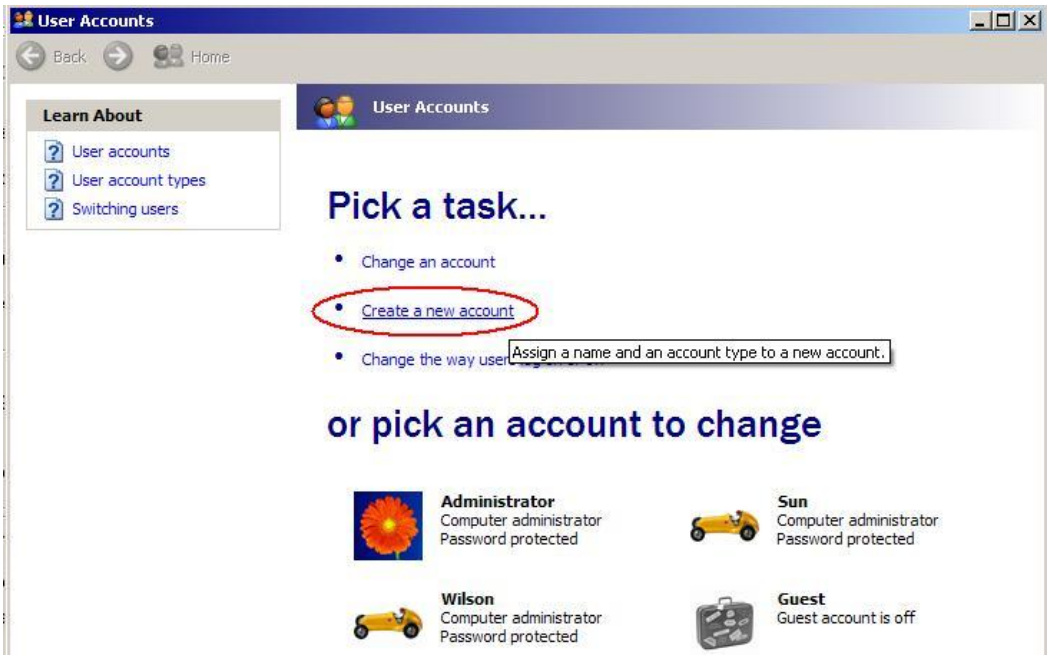
在 “Add a Port” 對話框中，輸入以下欄位後，再點選 TCP 單選按鈕:

Name: DCOM
Port number: 135



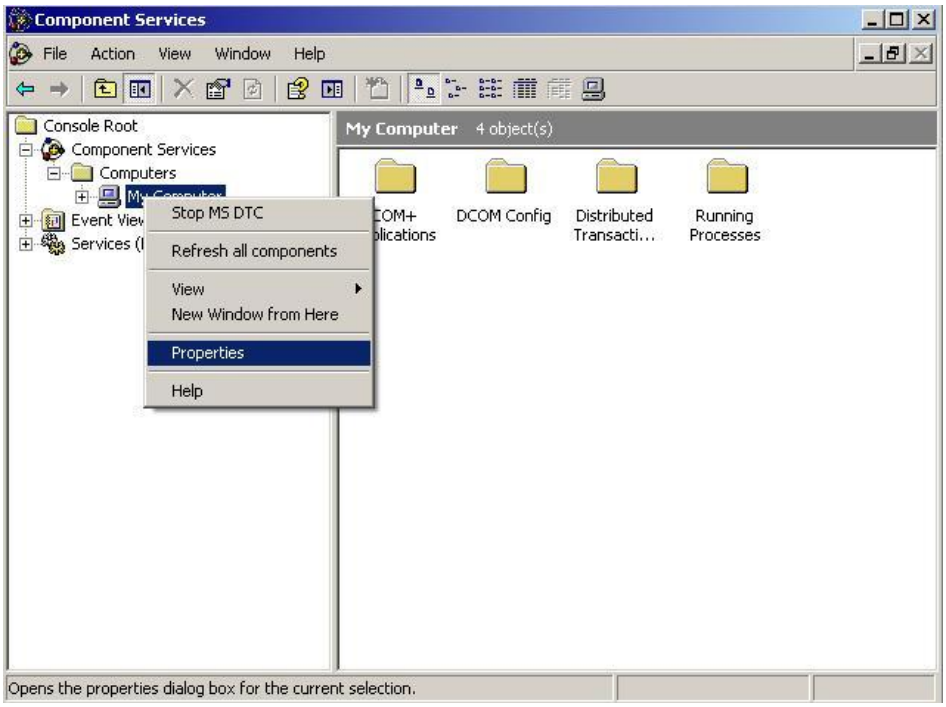
建立使用者帳戶

步驟 1: 建立一個帳戶，此帳戶必須與 Server 端的帳戶名稱相同。



設定 DCOM

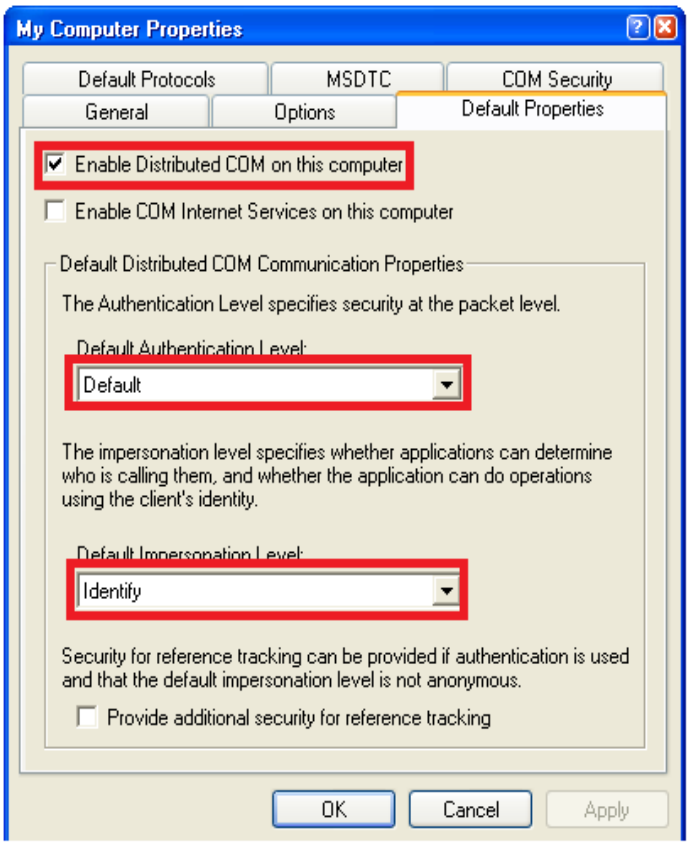
步驟 1: 執行 dcomcnfg.exe 程式來啟動元件服務。於 “My Computer” 按滑鼠右鍵並點選 “Properties”。



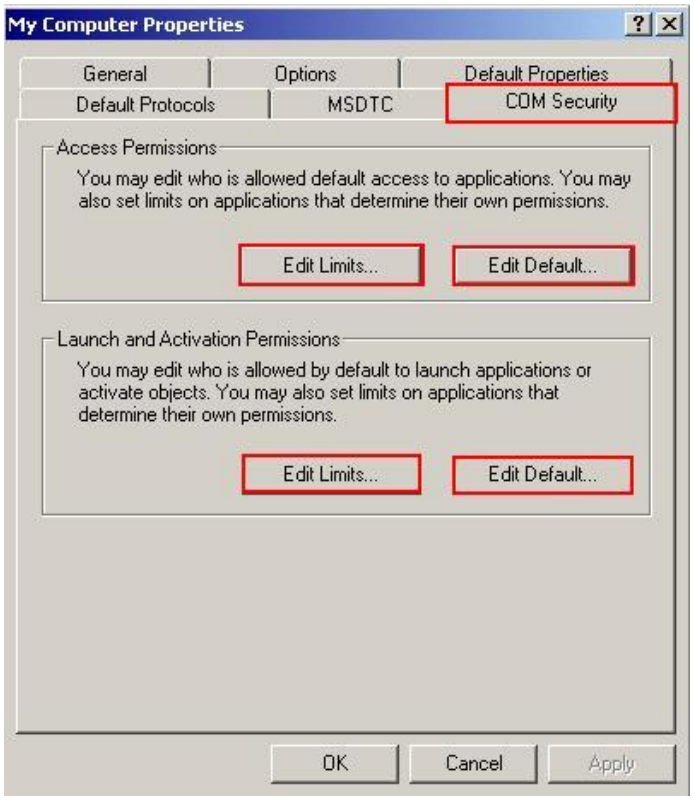
步驟 2: 點選 "Default Properties" 頁籤。

步驟 3: 使用以下設定:

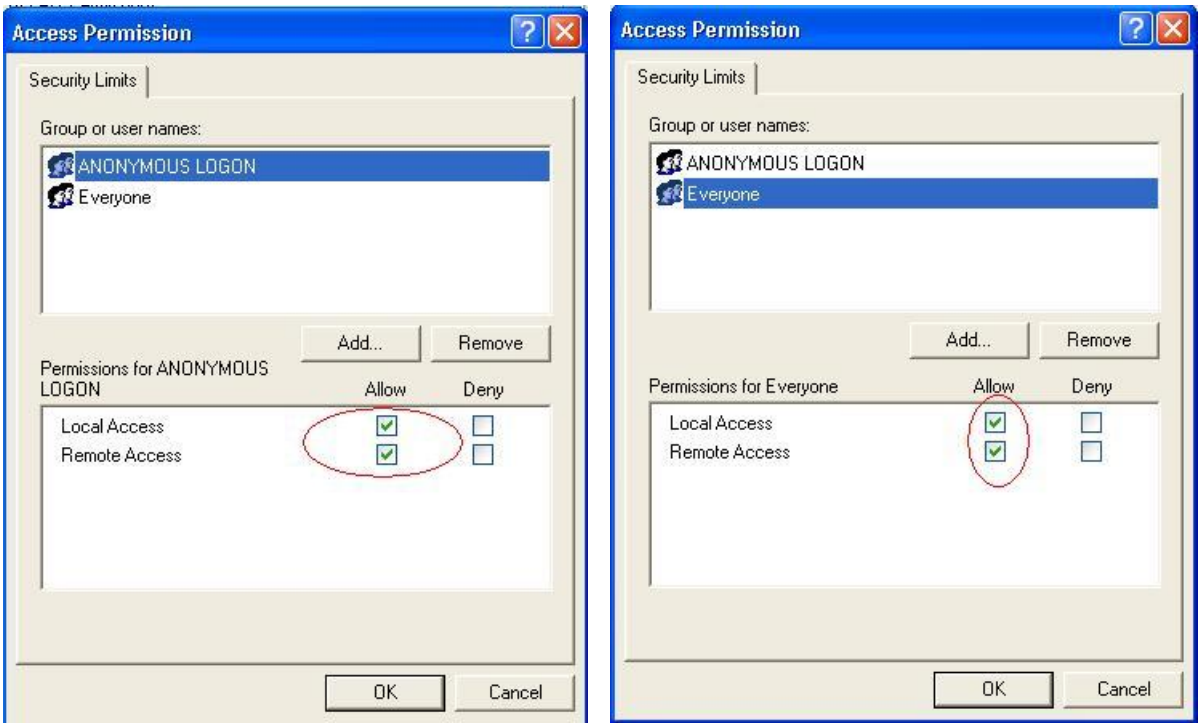
欄位名稱	設置
Enable Distributed COM on this computer	勾選
Default Authentication Level:	Default
Default Impersonation Level:	Identify



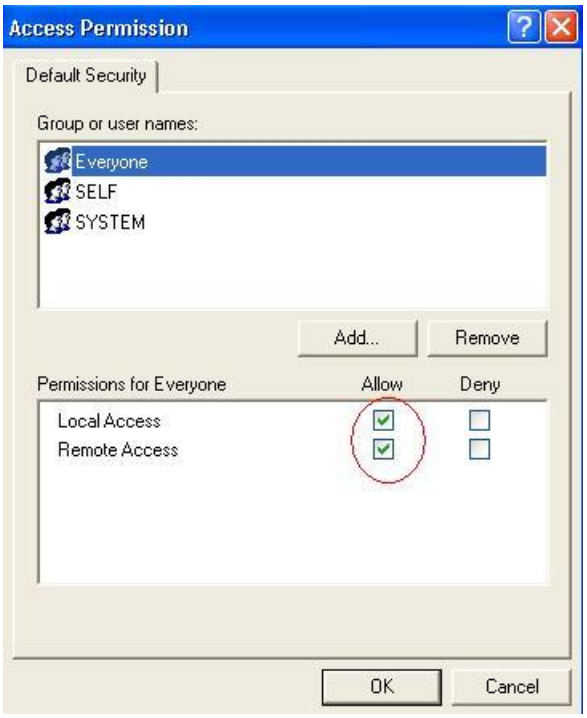
步驟 4: 點選 "COM Security" 頁籤。



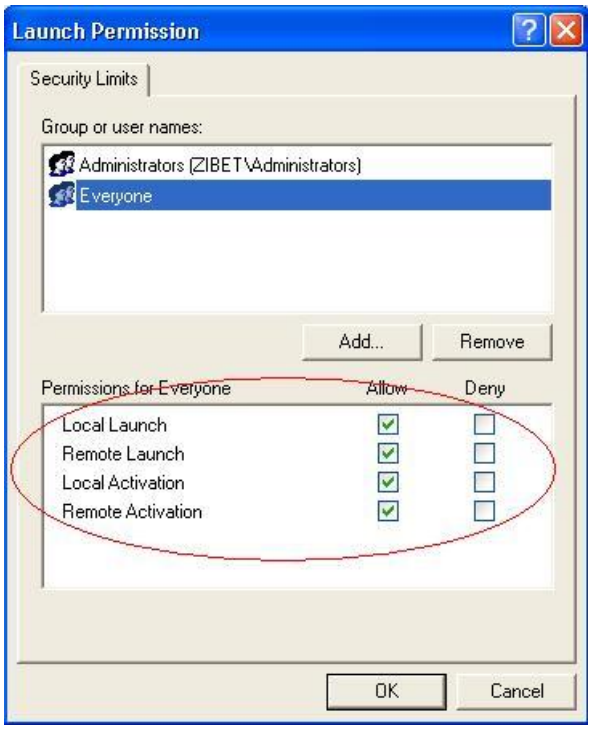
步驟 5: 如上圖，點選 “Access Permissions” 項目的 “Edit Limits…” 按鈕進入設定。



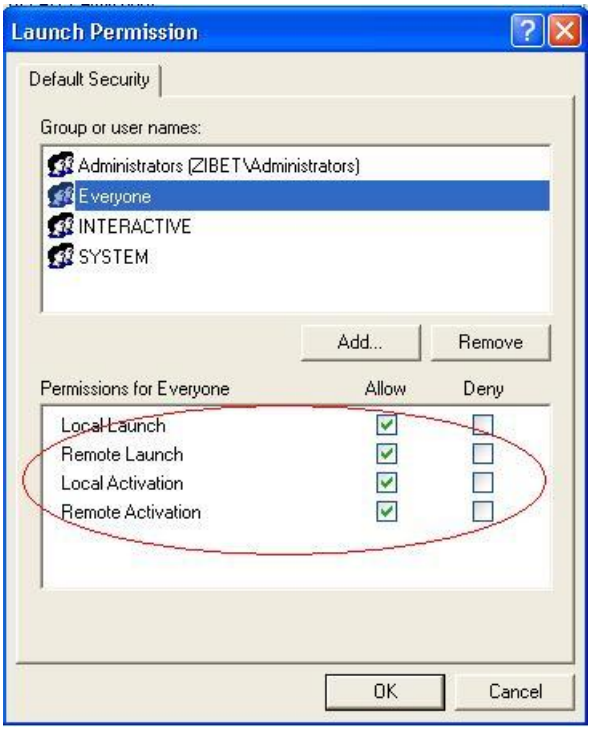
步驟 6: 如上述，點選 “Access Permissions” 項目的 “Edit Default…” 按鈕進入設定。



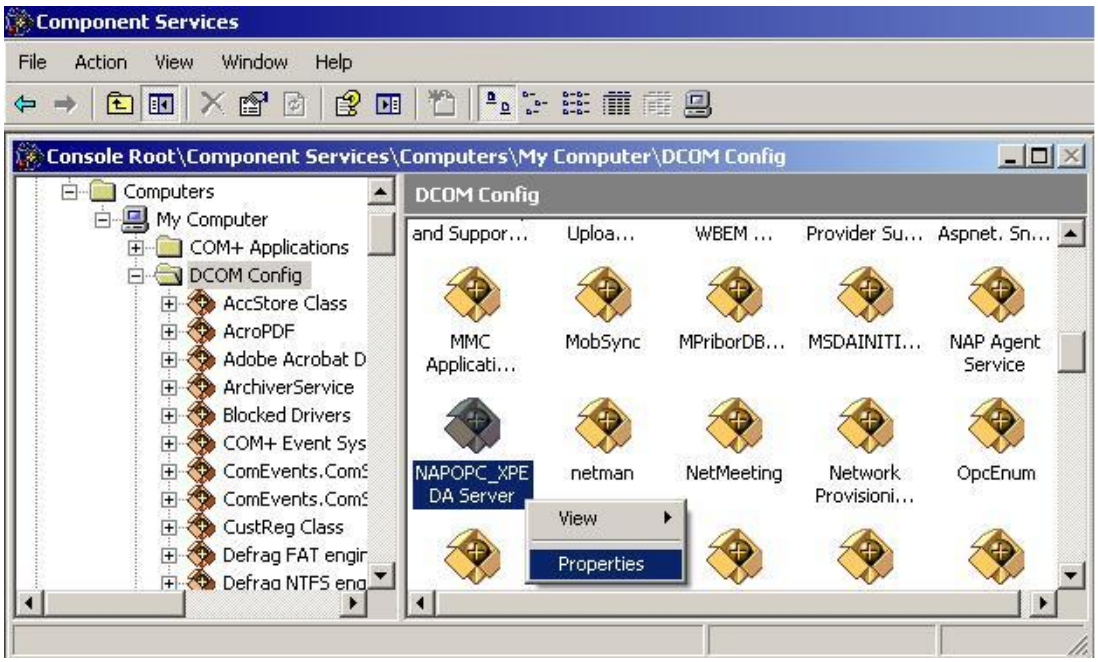
步驟 7: 如上述，點選 “Launch and Activation Permissions” 項目的 “Edit Limits…” 按鈕進入設定。



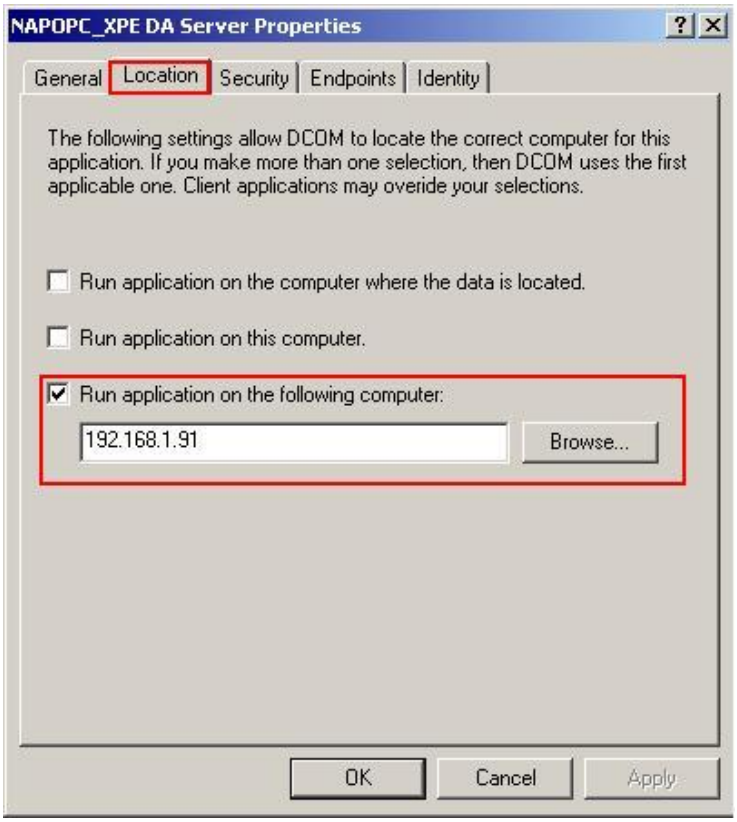
步驟 8: 如上述，點選 “Launch and Activation Permissions” 項目的 “Edit Default…” 按鈕進入設定。



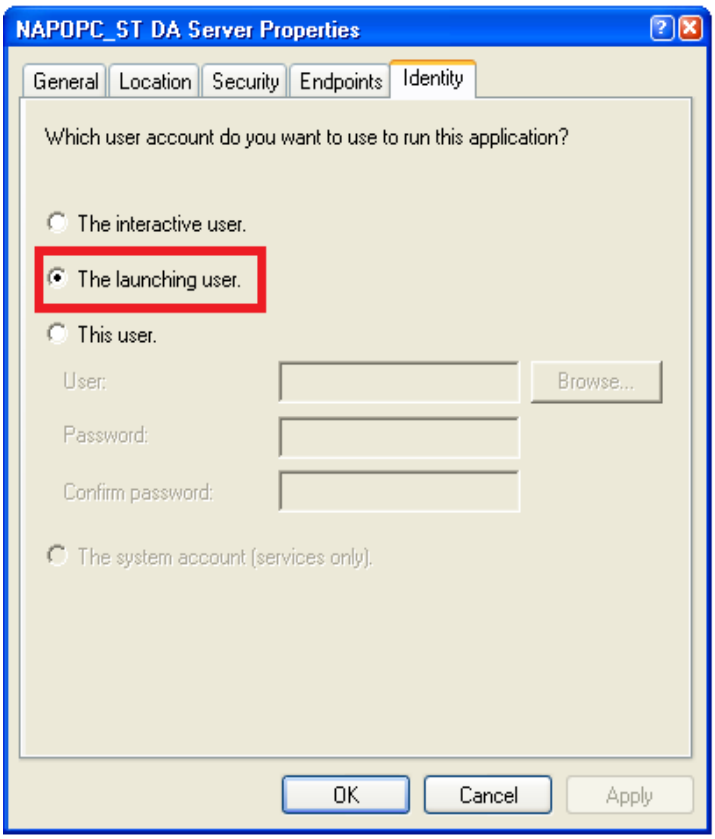
步驟 9: 如下圖，於 “DCOM Config” 中的 “NAPOPC_XPE DA Server” 按滑鼠右鍵並點選 “Properties”。



步驟 10: 點選 "Location" 頁籤，勾選 “Run application on the following computer” 並輸入 Server 的 IP 位址。



步驟 11: 點選 "Identity" 頁籤並選取 “The launching user” 項目。



步驟 12: 重新啟動 XPC。

3.2.4 設定 Client 端 (WinPAC)

系統需求

OS 版本:

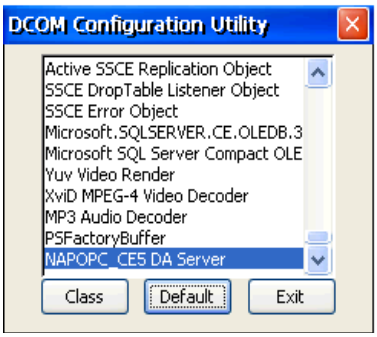
WinPAC OS 1.3.04 或 更新版本。

程式:

NAPOPC_CE5
DCOMCnfg.exe
WinPAC Utility 2.0.2.1 或 更新版本

設定 DCOM

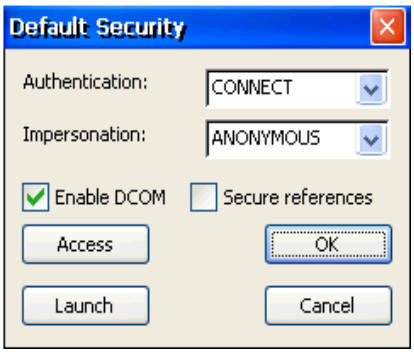
步驟 1: 執行 \\NAPOPC_CE5\napopc_ce5boot.exe 程式來註冊。



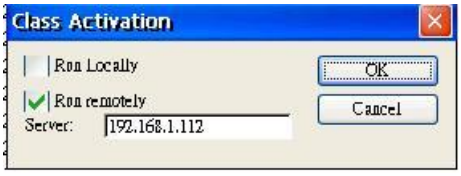
步驟 2: 執行 dcomcnfg.exe 程式並點選 “Default”。

步驟 3: 點選 “Access” 按鈕，加入(存取權限) 帳戶 (與 Server 端相同) 。

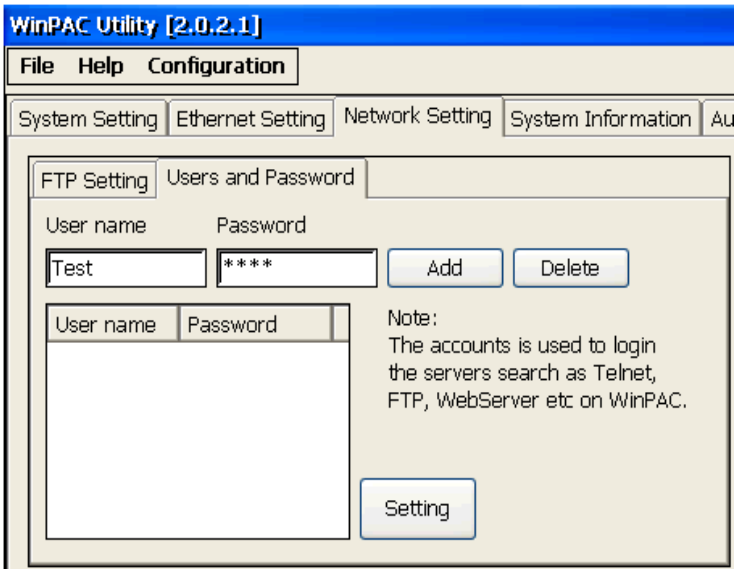
步驟 4: 點選 “Launch” 按鈕，加入(啟動權限) 帳戶 (與 Server 端相同) 。



步驟 5: 如上述，點選 “DCOM Configuration Utility” 對話框的 “Class” 按鈕來設定 “Class Activation”。取消勾選 “Run Locally” 並勾選 “Run remotelly”，再輸入 Server 端的 IP 位址。

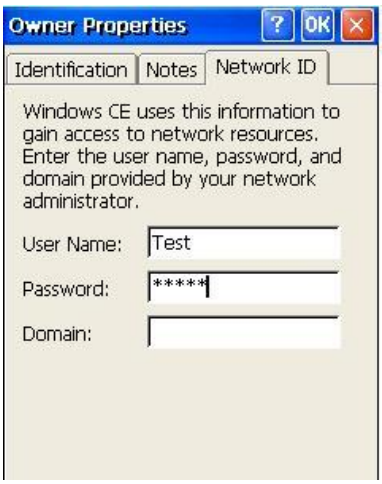


步驟 6: 執行 “WinPAC Utility->Network Setting->Users and Password”



步驟 7: 輸入 “User name”，“Password” 並點選 “Add”。“User name” 與 “Password” 必須是 **步驟 3** 所設定的帳戶名稱。最後，點選 “Setting” 來完成設定。

步驟 8: 點選 WinPAC 上的 “控制台” → “擁有者內容” → “網路識別碼” 並輸入與 Server 端帳戶相同的 User name 與 Password。



步驟 9: 執行 WinPAC Utility 中的 “save and reboot” 來儲存設定並重新啟動。

步驟 10: 您可執行 OPC Client 來進行測試。

4 NAPOPC_CE5 應用程式

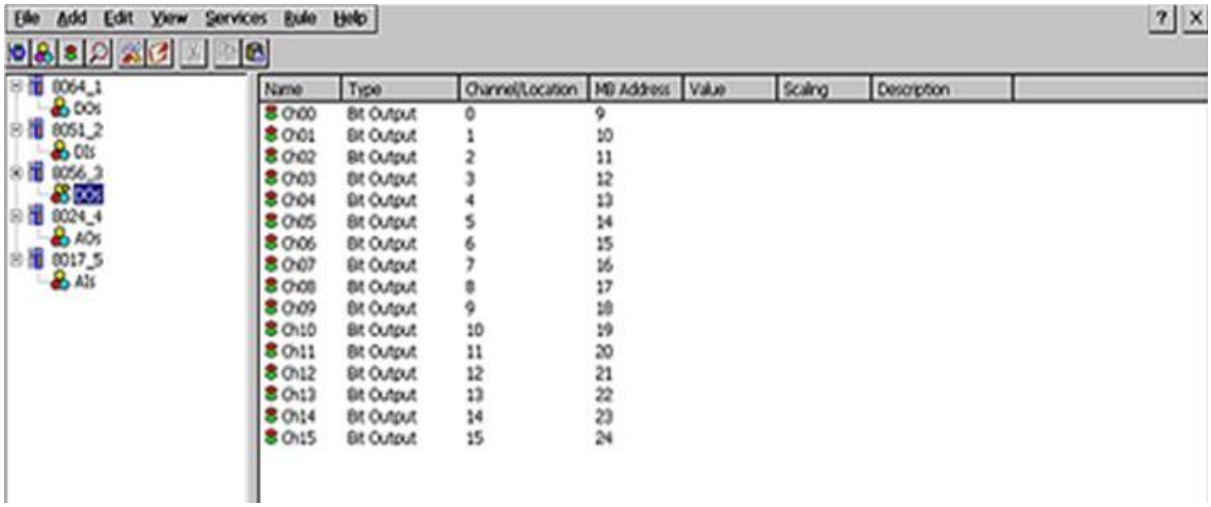
結合 OPC Client，Modbus RTU/TCP Client，NAPOPC_ST 與 NAPOPC_XPE 軟體，使用者可開發出一套令人驚艷的應用程式。若採用 NAPOPC_CE5 中的 “Rule Script” 功能，使用者不僅可節省許多時間於開發系統上，並可建立更具穩定性與安全性的系統。以下五個章節將介紹，應用於不同情況的時機與方法。

4.1 NAPOPC_CE5 搭配 OPC Client

NAPOPC_CE5 被設計為基於 OPC 的架構，因此自然支援 OPC Client。它適用於世界上許多基於 WinCE 的 OPC Client。請參閱其使用手冊以取得更多資訊。以下章節將說明如何連接 “InduSoft Web Studio Version 6.0” 至 Quicker。

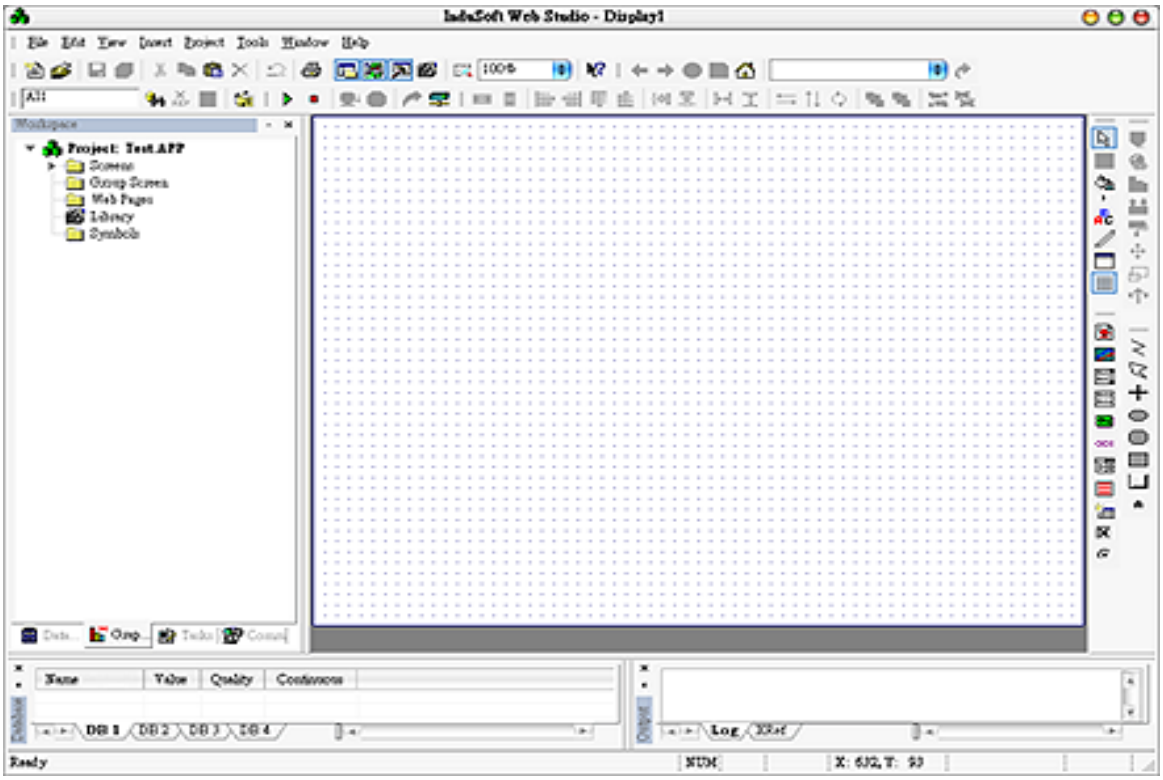
“InduSoft Web Studio” 是一個強大的，整合性的自動化工具集 – 包含了所有運用在開發人機介面 (HMIs)，監督控制 和 資料擷取 (SCADA) 系統，內嵌式檢測設備 與 控制應用程式所需的建構單元 (building blocks)。 Web Studio 運行在本機的 Windows NT，2000，XP，CE.Net 5.0 作業環境且符合工業標準 - 像是 Microsoft DNA，OPC，DDE，ODBC，XML，SOAP 與 ActiveX。請參訪網站: http://www.indusoft.com/ 取得詳細資訊。

步驟 1: 使用 InduSoft OPC Client 模組之前，您必須先在 WinPAC-8000 控制器上設定 NAPOPC_CE5。

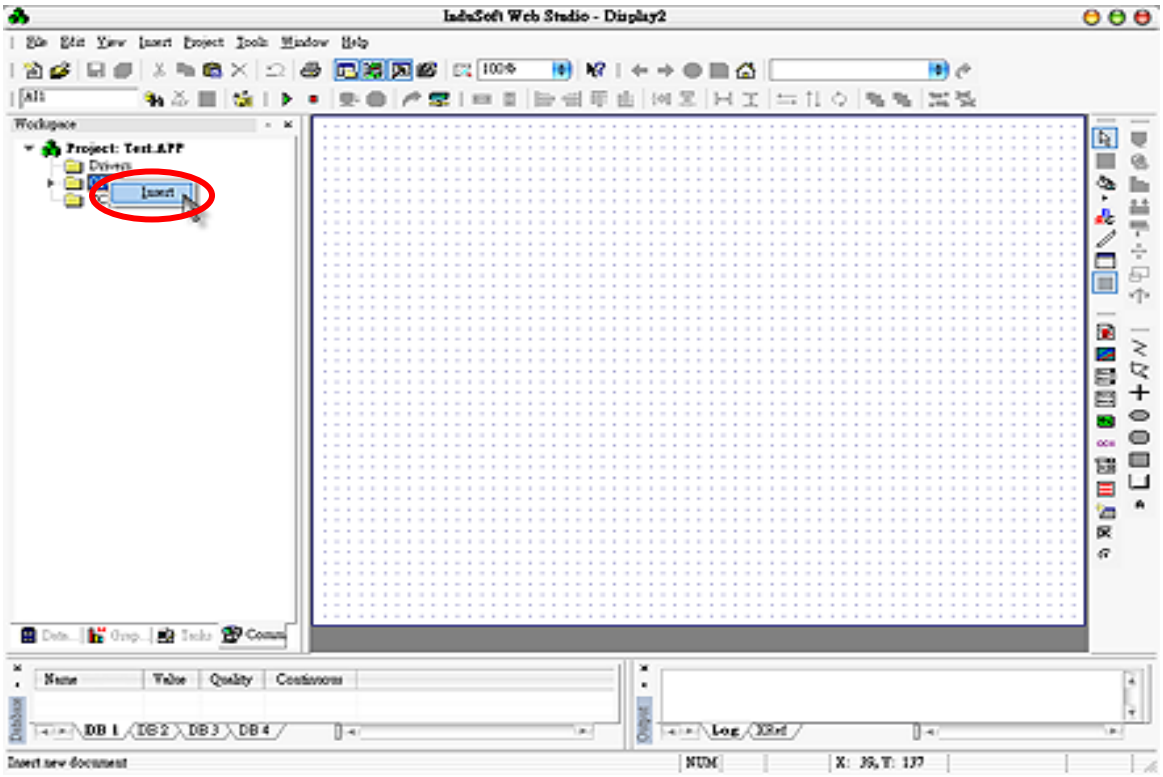


步驟 2: 執行 InduSoft Web Studio ver. 6.0。

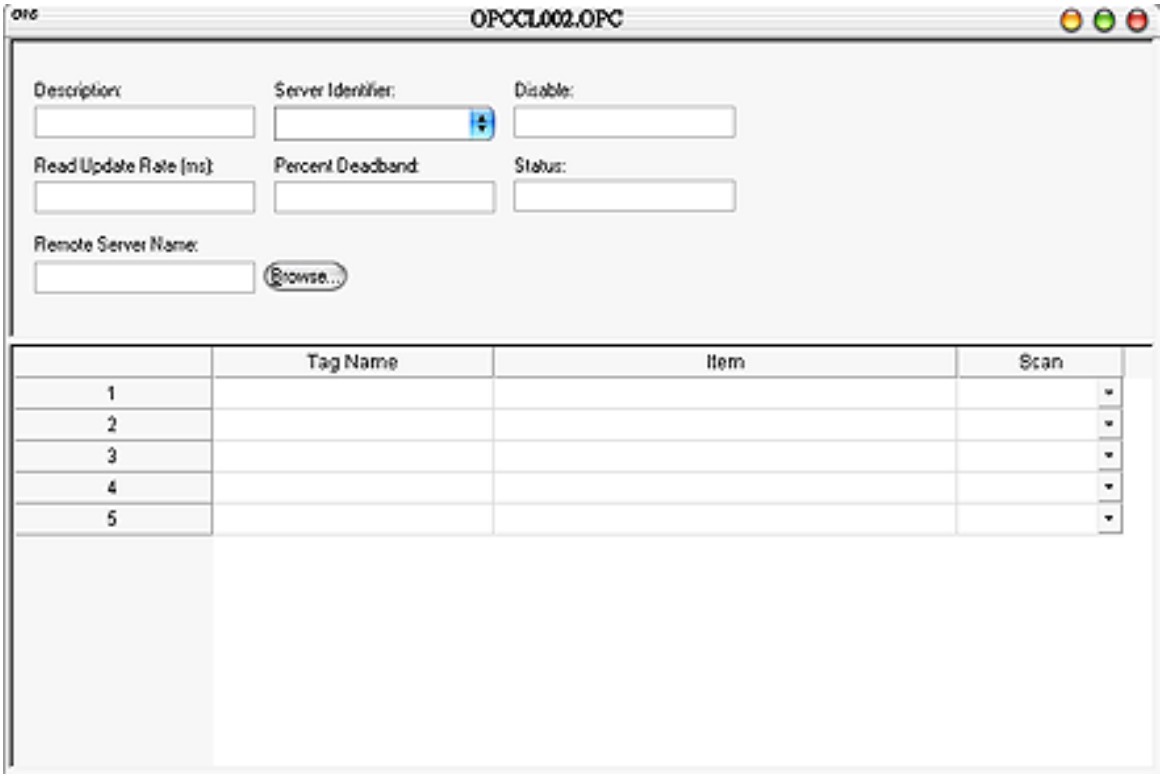
步驟 3: 新建一個專案。



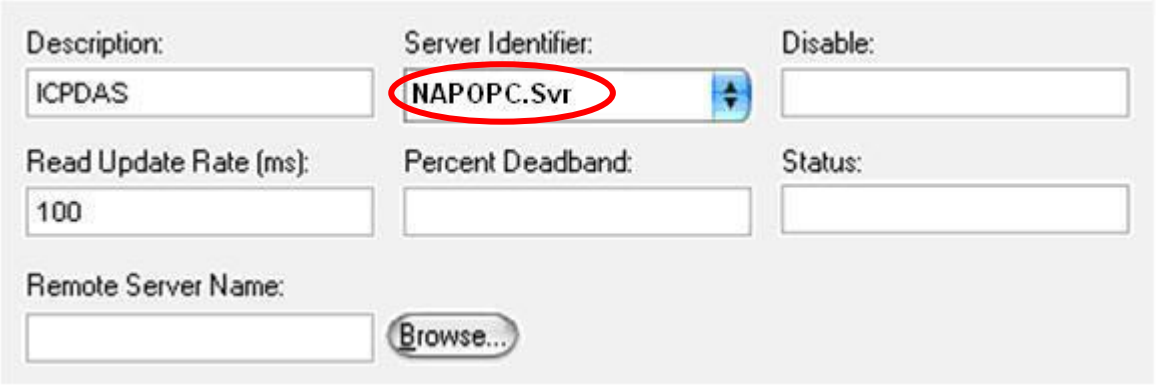
步驟 4: 於 Studio “Workspace” 視窗，點選 OPC 頁籤，在 OPC 目錄上按滑鼠右鍵，並點選 “Insert”。



步驟 5: 彈出 OPC 屬性視窗。



步驟 6: 點選 “Server Identifier”： 寫入 “NAPOPC.Svr” 檔案。

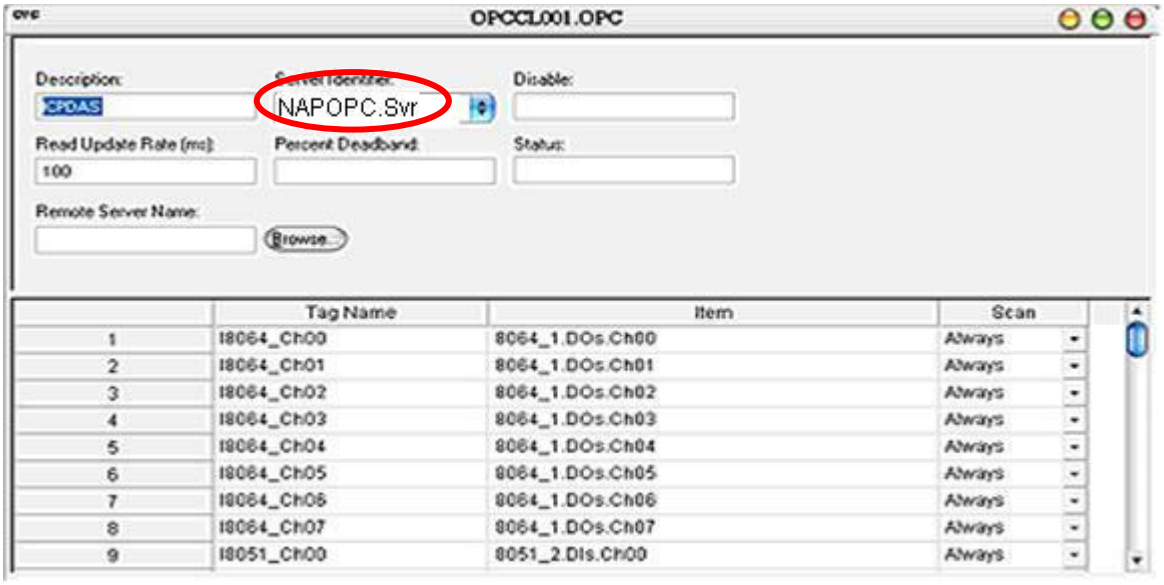


OPC 的設定表格有以下項目:

- Description: 此欄位僅供註記用。OPC Client 模組會略過此欄位。
- Server Identifier: 此欄位包含您欲連接的 Server 名稱。若 Server 安裝於電腦，其名稱可於列表框選取。
- Disable: 此欄位應包含一個 Tag 或常數。若其值為 “非 0”，表示與 OPC Server 之間的通訊為關閉狀態。
- Update Rate: 此欄位表示， Server 多久更新此群組 (單位: ms)。若為 “0” 表示 Server 會採用最快可行速度。
- Percent Deadband: 此欄位表示，該項目數值會導致 Server 發出通知的百分比變化。其只適用於類比項目。
- Tag Name: 此欄位包含了連接至 Server 項目的 Tag。
- Item: 此欄位包含了 Server 的項目名稱。

步驟 7: 於 “Tag Name” 欄位的第一格，輸入資料庫所建立的 Tag 名稱。

步驟 8: 於 “item” 欄位的第一格，您必須填入與 NAPOPC_CE5 中相同的設定。請參閱光碟中的 Demo。(“CD:\Compact Flash\ NAPOPC_CE5\Demo\InduSoft\Full”)

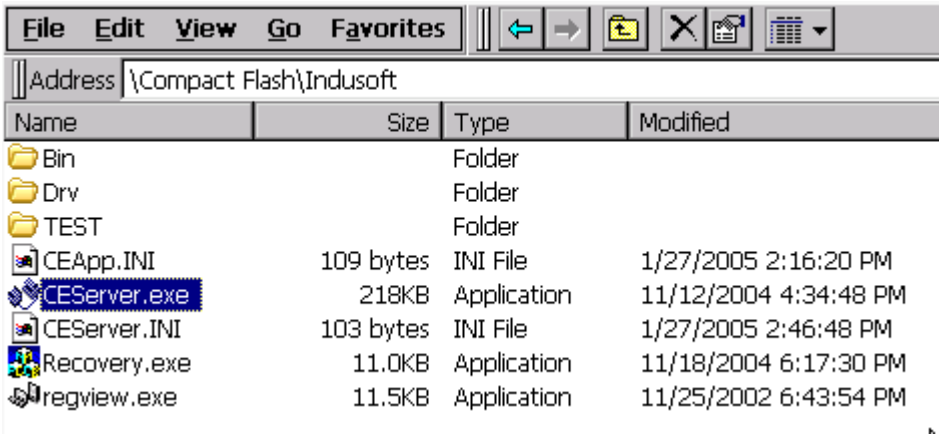


步驟 9: 重覆步驟 7，步驟 8 來新增更多的 Tag。

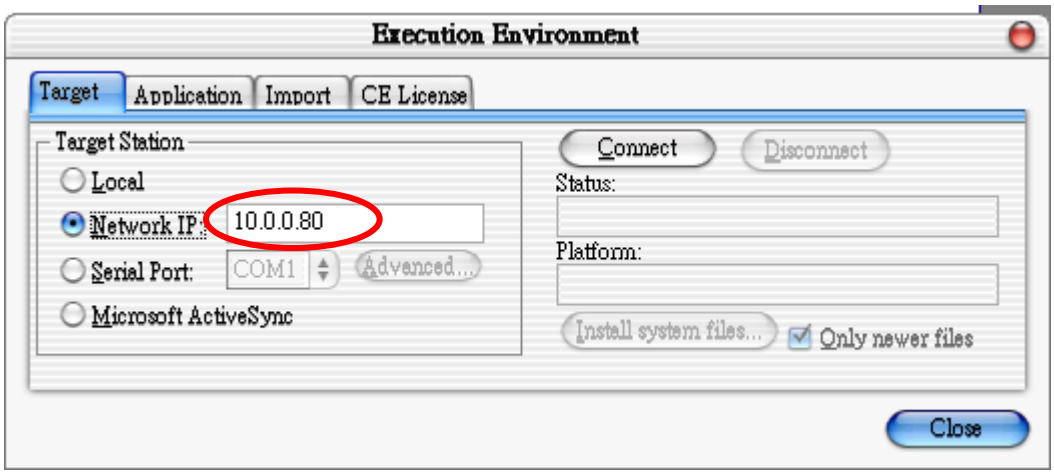
步驟 10: 為動態的輸入/輸出 建立一個文字字串。點選元件編輯工具列上的文字 (Text) 圖示，於主畫面的滑鼠十字標上按 “#” 鍵三次以在灰色方框中顯示 “###”。

步驟 11: 於元件編輯工具列，點選 輸入/輸出 文字的屬性圖示。物件屬性視窗的下拉選單將會顯示 輸入/輸出 文字。最後，於 Tag/Expression 欄位中輸入您欲連結的 Tag。

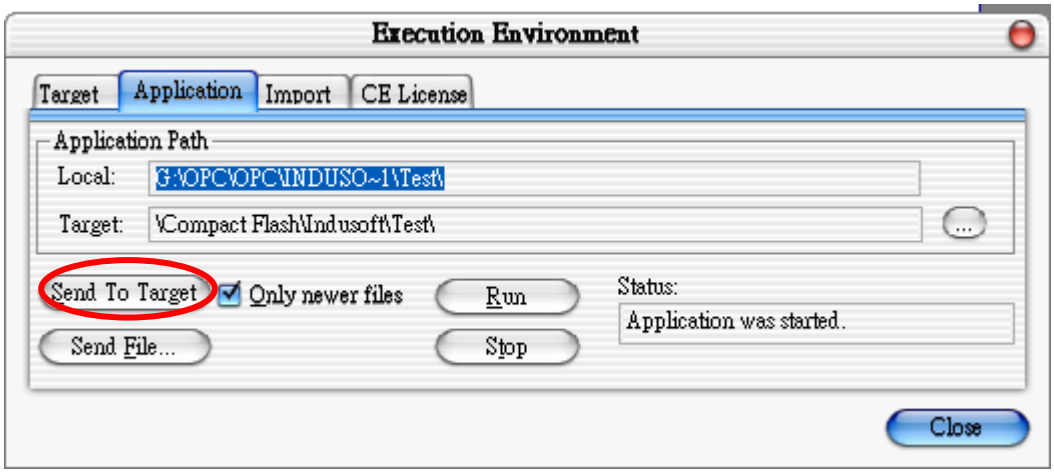
步驟 12: 完成設定後，點選以下路徑檔案 “Compact Flash\Indusoft\CEServer.exe” 來執行 InduSoft 遠端代理程式。



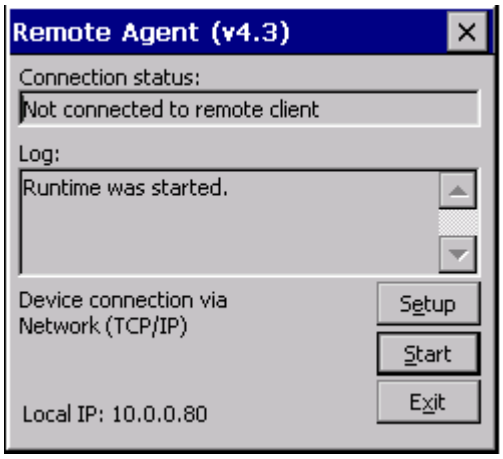
步驟 13: 點選 “Project → Execution Environment”，選取 “Network IP” 來輸入 WinPAC-8000 的 IP 位址，再點選 “Connect”。

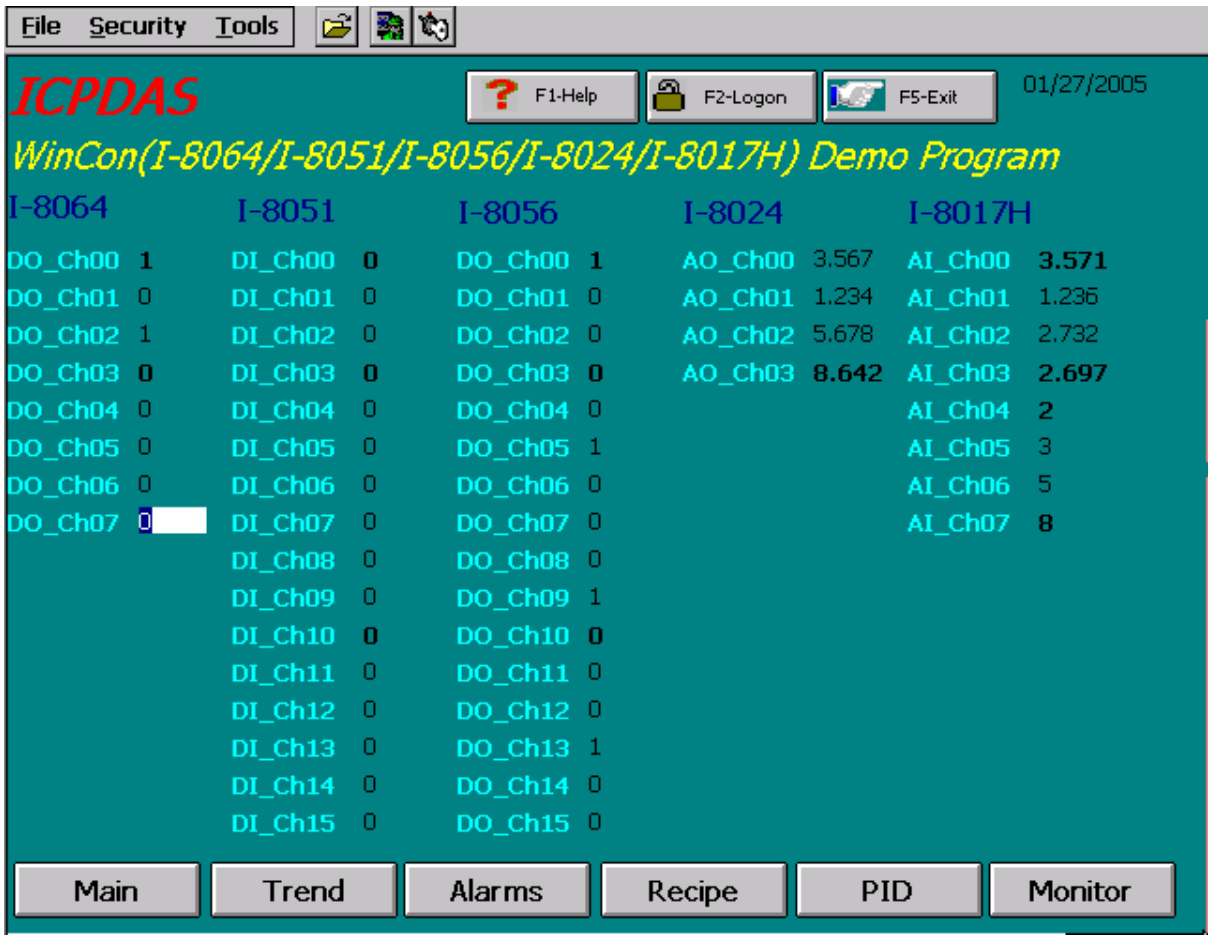


步驟 14: 選取 “Application → Send to Target”。



步驟 15: 點選 “Start” 來執行您的應用程式，之後，您將會見到運行之 HMI。





4.2 NAPOPC_CE5 搭配 Modbus RTU/TCP Client

若支援 Modbus RTU/TCP Client 的協力廠商軟體想要連到 NAPOPC_CE5，只需記得勾選 “Modbus RTU” 與 “Modbus TCP” 服務。請參閱協力廠商軟體 的設定使用手冊。關於 NAPOPC_CE5，請參閱 “1.2.11 服務設定” 章節。

4.2.1 支援 Modbus 命令

Modbus 通訊協定藉由置入設備 (或 廣播) 位址、定義了請求動作、需發送的任何資料、錯誤檢測範圍的功能碼，建立了 Master 的查詢格式。而 Slave 的回應訊息也是依據 Modbus 通訊協定所建構。它涵蓋了確認動作的執行、需回傳的任何資料與錯誤檢測範圍。若接收訊息時發生錯誤，或是 Slave 端無法執行請求動作，則 Slave 端會建立一個錯誤訊息來做為回應。

格式					
功能碼	說明	I/O 動作	單位	最小值	最大值
01(0x01)	Read Coil	Status In	Bit	1	2000(0x7D0)
02(0x02)	Read Discrete Inputs	Status In	Bit	1	2000(0x7D0)
03(0x03)	Read Holding Registers	Registers In	Word	1	125(0x7D)
04(0x04)	Read Input Registers	Registers In	Word	1	125(0x7D)
05(0x05)	Write Single Coil	Coil Out	Bit	1	1
06(0x06)	Write Single Register	Register Out	Word	1	1
15(0x0F)	Write Multiple Coils	Coils Out Bit	Bit	1	800
16(0x10)	Write Multiple registers	Registers Out Word	Word	1	100

4.3 NAPOPC_CE5 搭配 NAPOPC_ST/NAPOPC_XPE

您可透過使用 NAPOPC_CE5 並結合 NAPOPC_ST/NAPOPC_XPE 與 SCADA 軟體徹底地建構出一套完整的控制系統。請參閱 “1.2.11 服務設定” 章節，並依據 NAPOPC_ST/NAPOPC_XPE 所採用的通訊方式來設定 NAPOPC_CE5 服務。NAPOPC_CE5 提供了三種與 NAPOPC_ST/NAPOPC_XPE 通訊的方式 - ”Modbus TCP”，”Modbus RTU” 與 “RPC Server”。於 NAPOPC_ST/NAPOPC_XPE 端，請參閱 NAPOPC_ST/NAPOPC_XPE 手冊的 “新增 Modbus TCP 控制器”，” 新增 Modbus RTU 控制器” 與 “新增 RPC 控制器” 章節。

4.4 NAPOPC_CE5 搭配使用者的應用程式

使用者可使用 eVC++，VB.NET 或 VC#.NET 來開發自己的應用程式，並可透過 Quicker 的 API 與 NAPOPC_CE5 分享資料。使用者可採用 Modbus RTU/TCP 服務，或只採用 NAPOPC_CE5 內的共用記憶體來交換不同應用程式之間的資料。以下內容，我們不著重在 eVC++/VB.NET/VC#.NET 的編程技巧，而是著重在 Quicker 的 API。

4.4.1 適用於 eVC++ 開發者的 Quicker API

步驟 1:

安裝 pac270_sdk_2008xxxx.msi。

步驟 2:

選取 “Win32[WCE ARMV4I] CPU” 選項來新建一個 eVC++ 專案。

步驟 3:

#include "WinConAgent.h"。

步驟 4:

參閱下列函數來設計您自己的程式。

步驟 5:

以發行模式 (release mode) 來建立您的專案。

注意: Quicker.dll 與 eVC++ 應用程式必須複製到 WinPAC-8000 裡相同的目錄中。

系統函數
unsigned char StartQuicker(unsigned char iMode) unsigned char StopQuicker(void) unsigned char GetVersion()

QuickerIO 函數

unsigned char GetDIO(unsigned short iMBAddr, unsigned char *iRecv, unsigned char iAttribute);
unsigned char GetAIO_Short(unsigned short iMBAddr, short *iRecv, unsigned char iAttribute);
unsigned char GetAIO_Long(unsigned short iMBAddr, flong *iRecv, unsigned char iAttribute);
unsigned char GetAIO_Float(unsigned short iMBAddr, float *iRecv, unsigned char iAttribute);
unsigned char GetAIO_Word(unsigned short iMBAddr, unsigned short *iRecv, unsigned char iAttribute);
unsigned char GetAIO_DWord(unsigned short iMBAddr, unsigned long *iRecv, unsigned char iAttribute);
unsigned char SetDO(unsigned short iMBAddr, unsigned char iSend);
unsigned char SetAO_Short(unsigned short iMBAddr, short *iSend);
unsigned char SetAO_Long(unsigned short iMBAddr, long *iSend);
unsigned char SetAO_Float(unsigned short iMBAddr, float *iSend);
unsigned char SetAO_Word(unsigned short iMBAddr, unsigned short *iSend);
unsigned char SetAO_DWord(unsigned short iMBAddr, unsigned long *iSend);

Modbus 函數
unsigned char MBSetCoil(unsigned short iMBAddress, unsigned char iStatus, unsigned char iAttr) unsigned char MBGetCoil(unsigned short iMBAddress, unsigned char *iStatus, unsigned char iAttr) unsigned char MBSetReg(unsigned short iMBAddress, short iStatus, unsigned char iAttr) unsigned char MBGetReg(unsigned short iMBAddress, short *iStatus, unsigned char iAttr) unsigned char MBSetReg_Long(unsigned short iMBAddress, long iStatus, unsigned char iAttr) unsigned char MBGetReg_Long(unsigned short iMBAddress, long *iStatus, unsigned char iAttr) unsigned char MBSetReg_DWord(unsigned short iMBAddress, unsigned long iStatus, unsigned char iAttr) unsigned char MBGetReg_DWord(unsigned short iMBAddress, unsigned long *iStatus, unsigned char iAttr)

使用者共用函數

unsigned char UserSetCoil(unsigned short iUserAddress, unsigned char iStatus);
unsigned char UserGetCoil(unsigned short iUserAddress, unsigned char *iStatus);
unsigned char UserSetReg_Str(unsigned short iUserAddress, char *iStatus);
unsigned char UserGetReg_Str(unsigned short iUserAddress, char *iStatus);
unsigned char UserSetReg_Float(unsigned short iUserAddress, float *iStatus);
unsigned char UserGetReg_Float(unsigned short iUserAddress, float *iStatus);
unsigned char UserSetReg_Short(unsigned short iUserAddress, short *iStatus);
unsigned char UserGetReg_Short(unsigned short iUserAddress, short *iStatus);
unsigned char UserSetReg_Long(unsigned short iUserAddress, long *iStatus);
unsigned char UserGetReg_Long(unsigned short iUserAddress, long *iStatus);

4.4.1.1 系統函數

此群組提供了三種函數，可讓使用者啟動與停用 "NAPOPCSvr_CE5.exe" 軟體並在使用 "QuickerIO 函數" 與 "Modbus 函數" 之前取得 NAPOPC_CE5 軟體版本。

StartQuicker

此函數用來以不同模式來啟動 NAPOPC_CE5。

語法

```
[eVC++]
unsigned char StartQuicker(unsigned char iMode)
```

```
[VB.NET/VC#.NET]
byte Quicker.System.StartQuicker(byte iMode)
```

參數

iMode

[in] 核心模式的十進位編號。目前永遠為 “1”，將來會用來提供另一種模式。

回傳值

0 表示成功。若 NAPOPC_CE5 已經運行，此函數將回傳模式編號。

(請參閱 附錄 2.1)

附註

您必須在使用 QuickerIO 與 Modbus 函數之前，調用此函數來啟動 NAPOPC_CE5。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//以模式 1 來啟動 NAPOPC_CE5
if (StartQuicker(1) == 0){
AfxMessageBox(_T("Start NAPOPC_CE5 successfully!"));
}
else{
AfxMessageBox(_T("NAPOPC_CE5 has been started!"));
}
```

[VB.NET]

```
Quicker.System.StartQuicker(1)
```

[VC#.NET]

```
Quicker.System.StartQuicker(1)
```

StopQuicker

此函數用來停用 NAPOPC_CE5。

語法

[eVC++]

```
unsigned char StopQuicker(void)
```

[VB.NET/VC#.NET]

```
byte Quicker.System.StopQuicker()
```

參數

無。

回傳值

0 表示成功。 **WCA_Stop** 表示 NAPOPC_CE5 已經停用。**WCA_NOT_MASTER** 表示並非調用 NAPOPC_CE5 的主要 AP。(請參閱 附錄 2.1)

附註

NAPOPC_CE5 只能由啟動它的 AP 來將它停止。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//停用 NAPOPC_CE5
if(StopQuicker() == 0){
  AfxMessageBox(_T("Stop NAPOPC_CE5 successfully!"));
}
else if(StopQuicker() == WCA_Stop){
  AfxMessageBox(_T("NAPOPC_CE5 has been stopped!"));
}
else{
  AfxMessageBox(_T("Can not terminate the NAPOPC_CE5!"));
}
```

[VB.NET]

```
Quicker.System.StopQuicker()
```

[VC#.NET]

```
Quicker.System.StopQuicker()
```

GetVersion

此函數用來取得 NAPOPC_CE5 的版本。

語法

[eVC++]

```
unsigned char GetVersion(void)
```

[VB.NET/VC#.NET]

```
byte Quicker.System.GetVersion()
```

參數

無。

回傳值

回傳值即為版本號。 例如 209 表示 v2.09。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得 NAPOPC_CE5 版本
unsigned char iQversion;
iQversion = GetVersion();
```

[VB.NET]

```
Dim iQversion As Byte
iQversion = Quicker.System.GetVerison()
```

[VC#.NET]

```
byte iQversion = 0;
iQversion = Quicker.System.GetVersion();
```

4.4.1.2 QuickerIO 函數

此群組提供了 12 個函數讓使用者取得或設定，該 Modbus 位址對應到 NAPOPCSvr_CE5 中位址為 1 ~ 1000 的資料。透過 NAPOPC_CE5 可讓 OPC Client 與 Modbus Master 存取 Modbus 位址是對應至 1 ~ 1000 的資料。

GetDIO

此函數可從特定的 Modbus 位址來取得單個數位 I/O 狀態。

語法

```
[eVC++]
unsigned char GetDIO(unsigned short iMBAddr, unsigned char *iRecv,
                     unsigned char iAttribute
```

```
[VB.NET/VC#.NET]
byte GetDIO(ushort iMBAddr, out byte iRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的數位狀態。1 表示 ON，0 表示 OFF。

iAttribute

[in] 指定您需取得哪種數位狀態。 1 表示數位輸入，0 表示數位輸出。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得數位 I/O 狀態
//從 Modbus 位址 1 取得 DI 的狀態
unsigned char iRecvIn;
GetDIO(1,&iRecvIn,1);
//從 Modbus 位址 2 取得 DO 的狀態
unsigned char iRecvOut;
GetDIO(2,&iRecvOut,0);
```

[VB.NET]

```
Dim m_GetDIOVal As Byte
  Quicker.QuickerIO.GetDIO(7, m_GetDIOVal, 0)
```

[VC#.NET]

```
byte m_GetDIOVal;
Quicker.QuickerIO.GetDIO(7,out m_GetDIOVal, 0);
```

GetAIO_Short

此函數可從特定的 Modbus 位址來取得單個類比 I/O 數值。

語法

```
[eVC++]
unsigned char GetAIO_Short(unsigned short iMBAddr, short *iRecv,
                           unsigned char iAttribute)
```

```
[VB.NET/VC#.NET]
byte GetAIO_Short(ushort iMBAddr, out short fRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的類比數值。

iAttribute

[in] 指定您需取得的類比數值。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得類比 I/O 的數值
//從 Modbus 位址 1 取得 AI 的數值
short sRecvIn;
GetAIO_Short(1,&sRecvIn,1);
//從 Modbus 位址 2 取得 AO 的數值
short sRecvOut;
GetAIO_Short(2,&sRecvOut,0);
```

[VB.NET]

```
Dim m_GetAIOVal As short
  Quicker.QuickerIO.GetAIO_Short(7, m_GetAIOVal, 0)
```

[VC#.NET]

```
short m_GetAIOVal;
Quicker.QuickerIO.GetAIO_Short(7,out m_GetAIOVal, 0);
```

GetAIO_Long

此函數可從特定的 Modbus 位址來取得單個類比 I/O 的數值。

語法

```
[eVC++]
unsigned char GetAIO_Long(unsigned short iMBAddr, long *iRecv,
                          unsigned char iAttribute)
```

```
[VB.NET/VC#.NET]
byte GetAIO_Long(ushort iMBAddr, out long fRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的類比數值。

iAttribute

[in] 指定您需取得的類比數值。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得類比 I/O 的數值
//從 Modbus 位址 1 取得 AI 的數值
long lRecvIn;
GetAIO_Long(1,&fRecvIn,1);
//從 Modbus 位址 2 取得 AO 的數值
long lRecvOut;
GetAIO_Long(2,&fRecvOut,0);
```

[VB.NET]

```
Dim m_GetAIOVal As long
  Quicker.QuickerIO.GetAIO_Long(7, m_GetAIOVal, 0)
```

[VC#.NET]

```
long m_GetAIOVal;
Quicker.QuickerIO.GetAIO_Long(7,out m_GetAIOVal, 0);
```

GetAIO_Float

此函數可從特定的 Modbus 位址來取得單個類比 I/O 的數值。

語法

[eVC++]

```
unsigned char GetAIO_Float(unsigned short iMBAddr, float *iRecv,
                           unsigned char iAttribute)
```

[VB.NET/VC#.NET]

```
byte GetAIO_Float(ushort iMBAddr, out float fRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的類比數值。

iAttribute

[in] 指定您需取得的類比數值。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得類比 I/O 的數值
//從 Modbus 位址 1 取得 AI 的數值
float fRecvIn;
GetAIO_Float(1,&fRecvIn,1);
```

```
//從 Modbus 位址 2 取得 AO 的數值
float fRecvOut;
GetAIO_Float(2,&fRecvOut,0);
```

[VB.NET]

```
Dim m_GetAIOVal As Single
Quicker.QuickerIO.GetAIO_Float(7, m_GetAIOVal, 0)
```

[VC#.NET]

```
float m_GetAIOVal;
Quicker.QuickerIO.GetAIO_Float(7,out m_GetAIOVal, 0);
```

GetAIO_Word

此函數可從特定的 Modbus 位址來取得單個類比 I/O 的數值。

語法

```
[eVC++]
unsigned char GetAIO_Word(unsigned short iMBAddr, unsigned short *iRecv,
                          unsigned char iAttribute)
```

```
[VB.NET/VC#.NET]
byte GetAIO_Word(ushort iMBAddr, out ushort fRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的類比數值。

iAttribute

[in] 指定您需取得的類比數值。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得類比 I/O 的數值
//從 Modbus 位址 1 取得 AI 的數值
unsigned short usRecvIn;
GetAIO_Word(1,&fRecvIn,1);
//從 Modbus 位址 2 取得 AO 的數值
unsigned short usRecvOut;
GetAIO_Word(2,&usRecvOut,0);
```

[VB.NET]

```
Dim m_GetAIOVal As UInt16
Quicker.QuickerIO.GetAIO_Word(7, m_GetAIOVal, 0)
```

[VC#.NET]

```
ushort m_GetAIOVal;
Quicker.QuickerIO.GetAIO_Word(7,out m_GetAIOVal, 0);
```

GetAIO_DWord

此函數可從特定的 Modbus 位址來取得單個類比 I/O 的數值。

語法

[eVC++]

```
unsigned char GetAIO_DWord(unsigned short iMBAddr, unsigned long *iRecv,
                           unsigned char iAttribute)
```

[VB.NET/VC#.NET]

```
byte GetAIO_DWord(ushort iMBAddr, out ulong fRecv, byte iAttribute)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iRecv

[out] 特定 Tag 的類比數值。

iAttribute

[in] 指定您需取得的類比數值。

回傳值

0 表示成功。 **WCA_ATT_ERROR** 表示 iAttibute 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得類比 I/O 的數值
//從 Modbus 位址 1 取得 AI 的數值
unsigned long ulRecvIn;
GetAIO_DWord(1,&ulRecvIn,1);
//從 Modbus 位址 2 取得 AO 的數值
unsigned long ulRecvOut;
GetAIO_DWord(2,&ulRecvOut,0);
```

[VB.NET]

```
Dim m_GetAIOVal As UInt64
  Quicker.QuickerIO.GetAIO_DWord(7, m_GetAIOVal, 0)
```

[VC#.NET]

```
ulong m_GetAIOVal;
Quicker.QuickerIO.GetAIO_DWord(7,out m_GetAIOVal, 0);
```

SetDO

此函數可從特定的 Modbus 位址，設定單個數位輸出的狀態。

語法

```
[eVC++]
unsigned char SetDO(unsigned short iMBAddr, unsigned char iSend)
```

```
[VB.NET/VC#.NET]
byte SetDO(ushort iMBAddr, byte iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[in] 指定 Tag 的數位狀態。 1 表示 ON，0 表示 OFF。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的數位輸出為 ON
SetDO(1,1);
```

[VB.NET]

```
Dim m_SetDOVal As Byte
  Quicker.QuickerIO.SetDO(1, m_SetDOVal)
```

[VC#.NET]

```
byte m_SetDOVal;
Quicker.QuickerIO.SetDO(1, m_SetDOVal);
```

SetAO_Short

此函數可從特定的 Modbus 位址，設定單個類比輸出的數值。

語法

```
[eVC++]
unsigned char SetAO_Short(unsigned short iMBAddr, short *iSend)
```

```
[VB.NET/VC#.NET]
byte SetAO_Short(ushort iMBAddr, out short iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[out] 特定 Tag 的類比數值。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的類比輸出值為 42
SetAO_Short(1,42);
```

[VB.NET]

```
Quicker.QuickerIO.SetAO_Short(1, 42)
```

[VC#.NET]

```
Quicker.QuickerIO.SetAO_Short(1, 42);
```

SetAO_Long

此函數可從特定的 Modbus 位址，設定單個類比輸出的數值。

語法

[eVC++]

```
unsigned char SetAO_Long(unsigned short iMBAddr, long *iSend)
```

[VB.NET/VC#.NET]

```
byte SetAO_Long(ushort iMBAddr, out long iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[out] 特定 Tag 的類比數值。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的類比輸出值為 2323
SetAO_Long(1,2323);
```

[VB.NET]

```
Quicker.QuickerIO.SetAO_Long(1, 2323)
```

[VC#.NET]

```
Quicker.QuickerIO.SetAO_Long(1, 2323);
```

SetAO_Float

此函數可從特定的 Modbus 位址，設定單個類比輸出的數值。

語法

[eVC++]

```
unsigned char SetAO_Float(unsigned short iMBAddr, float *iSend)
```

[VB.NET/VC#.NET]

```
byte SetAO_Float(ushort iMBAddr, out float iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[out] 特定 Tag 的類比數值。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的類比輸出值為 5.5
SetAO_Float(1,5.5);
```

[VB.NET]

```
Quicker.QuickerIO.SetAO_Float(1, 5.5)
```

[VC#.NET]

```
Quicker.QuickerIO.SetAO_Float(1, 5.5);
```

SetAO_Word

此函數可從特定的 Modbus 位址，設定單個類比輸出的數值。

語法

[eVC++]

```
unsigned char SetAO_Word(unsigned short iMBAddr, unsigned short *iSend)
```

[VB.NET/VC#.NET]

```
byte SetAO_Word(ushort iMBAddr, out ushort iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[out] 特定 Tag 的類比數值。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的類比輸出值為 222
SetAO_Word(1,222);
```

[VB.NET]

```
Quicker.QuickerIO.SetAO_Word(1, 222)
```

[VC#.NET]

```
Quicker.QuickerIO.SetAO_Word(1, 222);
```

SetAO_DWord

此函數可從特定的 Modbus 位址，設定單個類比輸出的數值。

語法

```
[eVC++]
unsigned char SetAO_DWord(unsigned short iMBAddr, unsigned long *iSend)
```

```
[VB.NET/VC#.NET]
byte SetAO_DWord(ushort iMBAddr, out ulong iSend)
```

參數

iMBAddr

[in] NAPOPC_CE5 中特定 Tag 的 Modbus 位址。範圍為 1 ~ 1000。

iSend

[out] 特定 Tag 的類比數值。

回傳值

0 表示成功。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的類比輸出值為 2323
SetAO_DWord(1,2323);
```

[VB.NET]

```
Quicker.QuickerIO.SetAO_DWord(1, 2323)
```

[VC#.NET]

```
Quicker.QuickerIO.SetAO_DWord(1, 2323);
```

4.4.1.3 Modbus 函數

此群組提供了 8 個函數，讓使用者可將自己的變數加入至 NAPOPC_CE5 中，並可透過 NAPOPC_CE5 的 Modbus 服務與 Modbus Client 端共用數值。若使用者建立了內部設備與內部 Tag，即可透過 NAPOPC_CE5 讓 Modbus Client 端 與 OPC Client 端存取此資料。

MBSetCoil

此函數可設定 Coil 值 至 NAPOPC_CE5 中。

語法

[eVC++]

```
unsigned char MBSetCoil(unsigned short iMBAddress, unsigned char iStatus,
                        unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBSetCoil(ushort iMBAddress, byte iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲設定的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[in] 指定 Modbus 位址的 Coil 狀態。1 表示 ON，0 表示 OFF。

iAttr

[in] 指定您需設定為哪種 Coil。1 表示 “Input Coil"，由 Modbus 函數 2 發出請求。0 表示 “Output Coil"，由 Modbus 函數 1/5/15 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

//設定 Modbus 位址 1 的 “Input Coil” 狀態為 ON

[eVC++]

```
MBSetCoil(1,1,1);
```

[VB.NET]

```
Quicker.Modbus.MBSetCoil(1, 1, 1)
```

[VC#.NET]

```
Quicker.Modbus.MBSetCoil(1, 1, 1);
```

MBGetCoil

此函數可於特定的 Modbus 位址取得 Coil 數值。

語法

[eVC++]

```
unsigned char MBGetCoil(unsigned short iMBAddress, unsigned char *iStatus,
                        unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBGetCoil(ushort iMBAddress, out byte iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲取得的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[out] 指定 Modbus 位址的 Coil 狀態。1 表示 ON，0 表示 OFF。

iAttr

[in] 指定您需設定為哪種 Coil。1 表示 “Input Coil”，由 Modbus 函數 2 發出請求。0 表示 “Output Coil"，由 Modbus 函數 1/5/15 發出請求。

回傳值

0 表示成功。**WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//從 Modbus 位址 1 取得 “Input Coil” 的狀態
unsigned char iStatus;
MBGetCoil(1,&iSatus,1);
```

[VB.NET]

```
Dim m_MBGetCoilVal As Byte
Quicker.Modbus.MBGetCoil(1, m_MBGetCoilVal, 1)
```

[VC#.NET]

```
byte m_MBGetCoilVal;
Quicker.Modbus.MBGetCoil(1,out m_MBGetCoilVal, 1);
```

MBSetReg

此函數可設定註冊值至 NAPOPC_CE5 中。

語法

```
[eVC++]
unsigned char MBSetReg(unsigned short iMBAddress, short iStatus,
                       unsigned char iAttr)
```

```
[VB.NET/VC#.NET]
byte MBSetReg(ushort iMBAddress, short iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲設定的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[in] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需設定的註冊值。1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。 **WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的 “Input Register” 為 123
MBSetReg(1,123,1);
```

[VB.NET]

```
Quicker.Modbus.MBSetReg(1, 123, 1)
```

[VC#.NET]

```
Quicker.Modbus.MBSetReg(1, 123, 1) ;
```

MBGetReg

此函數用來取得特定 Modbus 位址的註冊值。

語法

```
[eVC++]
unsigned char MBGetReg(unsigned short iMBAddress, short *iStatus,
                       unsigned char iAttr)
```

```
[VB.NET/VC#.NET]
byte MBGetReg(ushort iMBAddress, out short iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲取得的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[out] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需取得的註冊值。 1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得 Modbus 位址 1 的 “Input Register” 數值
short iSataus;
MBGetReg(1,&iSatus,1);
```

[VB.NET]

```
Dim m_MBGetRegVal As short
Quicker.Modbus.MBGetReg(1, m_MBGetRegVal, 1)
```

[VC#.NET]

```
short m_MBGetRegVal;
Quicker.Modbus.MBGeReg(1,out m_MBGetRegVal, 1);
```

MBSetReg_Long

此函數可設定註冊值至 NAPOPC_CE5 中。

語法

[eVC++]

```
unsigned char MBSetReg_Long(unsigned short iMBAddress, long iStatus,
                            unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBSetReg(ushort iMBAddress, int iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲設定的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[in] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需設定的註冊值。 1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的 “Input Register” 數值為 123
MBSetReg_Long(1,123,1);
```

[VB.NET]

```
Quicker.Modbus.MBSetReg_Long(1, 123, 1)
```

[VC#.NET]

```
Quicker.Modbus.MBSetReg_Long(1, 123, 1) ;
```

MBGetReg_Long

此函數可於特定 Modbus 位址取得註冊值。

語法

[eVC++]

```
unsigned char MBGetReg_Long(unsigned short iMBAddress, long *iStatus,
                            unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBGetReg_Long(ushort iMBAddress, out int iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲取得的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[out] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需取得的註冊值。 1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得 Modbus 位址 1 的 “Input Register” 數值
long iSataus;
MBGetReg_Long(1,&iSatus,1);
```

[VB.NET]

```
Dim m_MBGetRegVal As Integer
Quicker.Modbus.MBGetReg_Long(1, m_MBGetRegVal, 1)
```

[VC#.NET]

```
int m_MBGetRegVal;
Quicker.Modbus.MBGeReg_Long(1,out m_MBGetRegVal, 1);
```

MBSetReg_DWord

此函數可設定註冊值至 NAPOPC_CE5 中。

語法

[eVC++]

```
unsigned char MBSetReg_DWord(unsigned short iMBAddress, unsigned long iStatus,
unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBSetReg(ushort iMBAddress, uint iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲設定的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[in] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需設定的註冊值。 1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定 Modbus 位址 1 的 “Input Register” 數值為 123
MBSetReg_DWord(1,123,1);
```

[VB.NET]

```
Quicker.Modbus.MBSetReg_DWord(1, 123, 1)
```

[VC#.NET]

```
Quicker.Modbus.MBSetReg_DWord(1, 123, 1) ;
```

MBGetReg_DWord

此函數可於特定 Modbus 位址取得註冊值。

語法

[eVC++]

```
unsigned char MBGetReg_DWord(unsigned short iMBAddress, unsigned long
*iStatus, unsigned char iAttr)
```

[VB.NET/VC#.NET]

```
byte MBGetReg_DWord(ushort iMBAddress, out uint iStatus, byte iAttr)
```

參數

iMBAddress

[in] 您欲取得的 Modbus 位址，範圍為 1001 ~ 20999。

iStatus

[out] 特定 Modbus 位址的註冊值。

iAttr

[in] 指定您需取得的註冊值。 1 表示 “Input Register”，由 Modbus 函數 4 發出請求。0 表示 “Output Register”，由 Modbus 函數 3/6/16 發出請求。

回傳值

0 表示成功。 **WCA_MBADDR_OVER** 表示 iMBAddress 超出範圍，合理範圍是編號 1001 ~ 20999。**WCA_MBATTR_ERROR** 表示 iAttr 既非 0 也非 1。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得 Modbus 位址 1 的 “Input Register” 數值
unsigned long iSataus;
MBGetReg_DWord(1,&iSatus,1);
```

[VB.NET]

```
Dim m_MBGetRegVal As UInt32
Quicker.Modbus.MBGetReg_DWord(1, m_MBGetRegVal, 1)
```

[VC#.NET]

```
uint m_MBGetRegVal;
Quicker.Modbus.MBGeReg_DWord(1,out m_MBGetRegVal, 1);
```

4.4.1.4 使用者共用函數

這些函數允許使用者將自己的變數加入至共用記憶體區塊，以便與不同應用程式共用數值。採用這些函數的資料將無法被 Modbus Client 與 OPC Client 存取。

UerSetCoil

此函數可設定一個無號數字元變數到共用記憶體區塊。

語法

```
[eVC++]
unsigned char UserSetCoil(unsigned short iUserAddress, unsigned char iStatus)
```

```
[VB.NET/VC#.NET]
byte UserSetCoil(ushort iUserAddress, byte iStatus)
```

參數

iUserAddress

[in] 您欲設定的位址，位址範圍是 1 ~ 19999。

iStatus

[in] 無號數字元變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 " iUserAddress" 超出範圍， 合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定位址 1 的 Coil 數值
UserSetCoil(1,1);
```

[VB.NET]

```
Quicker.UserShare.UserSetCoil(1, 1)
```

[VC#.NET]

```
Quicker.UserShare.UserSetCoil(1, 1);
```

UserGetCoil

此函數共用記憶體區塊取得無號數字元變數。

語法

```
[eVC++]
unsigned char UserGetCoil(unsigned short iUserAddress, unsigned char *iStatus)
```

```
[VB.NET/VC#.NET]
byte UserGetCoil(ushort iUserAddress, out byte iStatus)
```

參數

iUserAddress

[in] 您欲取得的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 指向一個 無號數字元變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 " iUserAddress" 超出範圍， 合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//取得位址 1 的 Coil 數值
unsigned char iStatus;
UserGetCoil(1,&iSatus);
```

[VB.NET]

```
Dim m_UserGetCoilVal As Byte
Quicker.UserShare.UserGetCoil(1, m_UserGetCoilVal)
```

[VC#.NET]

```
byte m_UserGetCoilVal;
Quicker.UserShare.UserGetCoil(1,out m_UserGetCoilVal);
```

UserSetReg_Str

此函數可用來設定字串變數到共用記憶體區塊。

語法

[eVC++]

```
unsigned char UserSetReg_Str(unsigned short iUserAddress, char *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserSetReg_Str(ushort iUserAddress, char[] cSetStr)
```

參數

iUserAddress

[in] 您欲設定的位址，位址範圍是 1 ~ 1024。

iStatus

[out] 字元變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合理範圍是編號 1 ~ 1024。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定位址 1 為字串 KKK
char *SetString;
CString m_USAValStr;
m_USAValStr = _T("KKK");
SetString =  (LPSTR)(LPCTSTR)m_USAValStr;
UserSetReg_Str(1,SetString);
```

[VB.NET]

```
Dim Rtn As Byte
Dim UserSetRegStrVal As String
Rtn = Quicker.UserShare.UserSetReg_Str(1, UserSetRegStrVal.ToCharArray())
```

[VC#.NET]

```
byte Rtn;
string UserSetRegStrVal;
Rtn = Quicker.UserShare.UserSetReg_Str(1, UserSetRegStrVal.ToCharArray());
```

UserGetReg_Str

此函數可從共用記憶體區塊取得字串變數。

語法

[eVC++]

```
unsigned char UserGetReg_Str(unsigned short iUserAddress, char *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserGetReg_Str(ushort iUserAddress, byte[] cGetStr)
```

參數

iUserAddress

[in] 您欲取得的位址，位址範圍是 1 ~ 1024。

iStatus

[out] 指向一個長整數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合理範圍是編號 1 ~ 1024。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//從 Modbus 位址 1 取得字串
char iStatus[256];
UserGetReg_Str(1,iStatus);
```

[VB.NET]

```
Dim UserGetStr(256) As Byte
Dim Rtn As Byte
Rtn = Quicker.UserShare.UserGetReg_Str(1, UserGetStr)
```

[VC#.NET]

```
byte Rtn;
byte[] UserGetStr = new byte[256];
Rtn = Quicker.UserShare.UserGetReg_Str(1, UserGetStr);
```

UserSetReg_Float

此函數可設定一個浮點數變數至共用記憶體區塊。

語法

[eVC++]

```
unsigned char UserSetReg_Float(unsigned short iUserAddress, float *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserSetReg_Float(ushort iUserAddress, out float iStatus)
```

參數

iUserAddress

[in] 您欲設定的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 浮點數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定位址 1 的註冊值為 2.5
UserSetReg_Float(1,2.5);
```

[VB.NET]

```
Dim Rtn As Byte
Dim UserSetRegFloatVal As Single
 Rtn = Quicker.UserShare.UserSetReg_Float(1, UserSetRegFloatVal)
```

[VC#.NET]

```
byte Rtn;
float RegFloat;
Rtn = Quicker.UserShare.UserSetReg_Float(1,out RegFloat);
```

UserGetReg_Float

此函數可從共用記憶體區塊取得一個浮點數變數。

語法

[eVC++]

```
unsigned char UserGetReg_Float(unsigned short iUserAddress, float *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserGetReg_Float(ushort iUserAddress, out float iStatus)
```

參數

iUserAddress

[in] 您欲取得的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 指向一個 浮點數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//從位址 1 取得註冊值
float iStatus;
UserGetReg_Float(1,&iSatus);
```

[VB.NET]

```
Dim Rtn As Byte
Dim m_UserGetRegFloatVal As Single
Rtn = Quicker.UserShare.UserGetReg_Float(1, m_UserGetRegFloatVal)
```

[VC#.NET]

```
byte Rtn;
float m_UserGetRegFloatVal;
Rtn = Quicker.UserShare.UserGetReg_Float(1,out m_UserGetRegFloatVal);
```

UserSetReg_Short

此函數可用來設定一個短整數變數至 共用記憶體區塊。

語法

[eVC++]

```
unsigned char UserSetReg_Short(unsigned short iUserAddress, short *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserSetReg_short(ushort iUserAddress, out int iStatus)
```

參數

iUserAddress

[in] 您欲設定的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 短整數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定位址 1 的註冊值為 222
UserSetReg_Short(1,222);
```

[VB.NET]

```
Dim Rtn As Byte
Dim UserSetRegShortVal As Integer
Rtn = Quicker.UserShare.UserSetReg_Short(1, UserSetRegShortVal)
```

[VC#.NET]

```
byte Rtn;
int RegShort;
Rtn = Quicker.UserShare.UserSetReg_Short(1,out RegShort);
```

UserGetReg_Short

此函數可從共用記憶體區塊取得一個短整數變數。

語法

```
[eVC++]
unsigned char UserGetReg_Short(unsigned short iUserAddress, short *iStatus)
```

```
[VB.NET/VC#.NET]
byte UserGetReg_Float(ushort iUserAddress, out short iStatus)
```

參數

iUserAddress

[in] 您欲取得的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 指向一個 短整數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 “ iUserAddress” 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//從位址 1 取得註冊值
short iStatus;
UserGetReg_Short(1,&iSatus);
```

[VB.NET]

```
Dim Rtn As Byte
Dim m_UserGetRegShortVal As Integer
Rtn = Quicker.UserShare.UserGetReg_Short(1, m_UserGetRegShortVal)
```

[VC#.NET]

```
byte Rtn;
short m_UserGetRegShortVal;
Rtn = Quicker.UserShare.UserGetReg_Short(1,out m_UserGetRegShortVal);
```

UserSetReg_Long

此函數可用來設定一個長整數變數至共用記憶體區塊。

語法

[eVC++]

```
unsigned char UserSetReg_Long(unsigned short iUserAddress, long *iStatus)
```

[VB.NET/VC#.NET]

```
byte UserSetReg_Long(ushort iUserAddress, out long iStatus)
```

參數

iUserAddress

[in] 您欲設定的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 長整數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 " iUserAddress" 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//設定位址 1 的註冊值為 112233
UserSetReg_Long(1,112233);
```

[VB.NET]

```
Dim Rtn As Byte
Dim UserSetRegLongVal As Integer
Rtn = Quicker.UserShare.UserSetReg_Long(1, UserSetRegLongVal)
```

[VC#.NET]

```
byte Rtn;
int RegLong;
Rtn = Quicker.UserShare.UserSetReg_Long(1,out RegLong);
```

UserGetReg_Long

此函數可從共用記憶體區塊取得一個長整數變數。

語法

```
[eVC++]
unsigned char UserGetReg_Long(unsigned short iUserAddress, long *iStatus)
```

```
[VB.NET/VC#.NET]
byte UserGetReg_Long(ushort iUserAddress, out long iStatus)
```

參數

iUserAddress

[in] 您欲取得的位址，位址範圍是 1 ~ 19999。

iStatus

[out] 指向一個 長整數變數。

回傳值

0 表示成功。 **WCA_USERADDR_OVER** 表示 " iUserAddress" 超出範圍，合法範圍是編號 1 ~ 19999。

附註

無。

需求

運行於	版本	定義檔	括入檔	連接至
WinPAC 8000	4.1.0.01 與更新版	Quicker.lib	WinConAgent.h	-

範例

[eVC++]

```
//從位址 1 取得註冊值
long iStatus;
UserGetReg_Long(1,&iSatus);
```

[VB.NET]

```
Dim Rtn As Byte
Dim m_UserGetRegLongVal As Integer
Rtn = Quicker.UserShare.UserGetReg_Long(1, m_UserGetRegLongVal)
```

[VC#.NET]

```
byte Rtn;
int m_UserGetRegLongVal;
Rtn = Quicker.UserShare.UserGetReg_Long(1,out m_UserGetRegLongVal);
```

4.4.2 適用於 VB.NET/VC#.NET 開發者的 Quicker API

步驟 1:

建立一個設備專案。

步驟 2:

[Add Reference] -> QuickerNet.dll。

步驟 3:

透過元件瀏覽器，參考 QuickerNet.dll 中的函數屬性。

步驟 4:

調用 QuickerNet.dll 中的函數 。

(請參閱 Quicker_VB.NET_Demo /Quicker_VC#.NET_Demo)

步驟 5:

建立您的新專案，並複製專案與相關程式庫到 WinPAC-8000 中。

注意: Quicker.dll, QuickerNet.dll 與 VB.NET/VC#.NET 應用程式必須複製到 WinPAC-8000 裡相同的目錄中。

4.5 NAPOPC_CE5 搭配 “Rule Script”

NAPOPC_CE5 提供了 “Rule Script Editor” 功能來讓使用者編寫規則。此功能是依據直觀的設計風格來開發規則清單。程式設計者可輕鬆地藉由 “IF…THEN…” 語法於規則清單中實現他們的邏輯，並完成連鎖反應控制的效用。“Rule Script” 適用在非臨界 (non-critical) 的狀況。採用此功能不僅可避免打字錯誤，更可節省專案開發的時間。

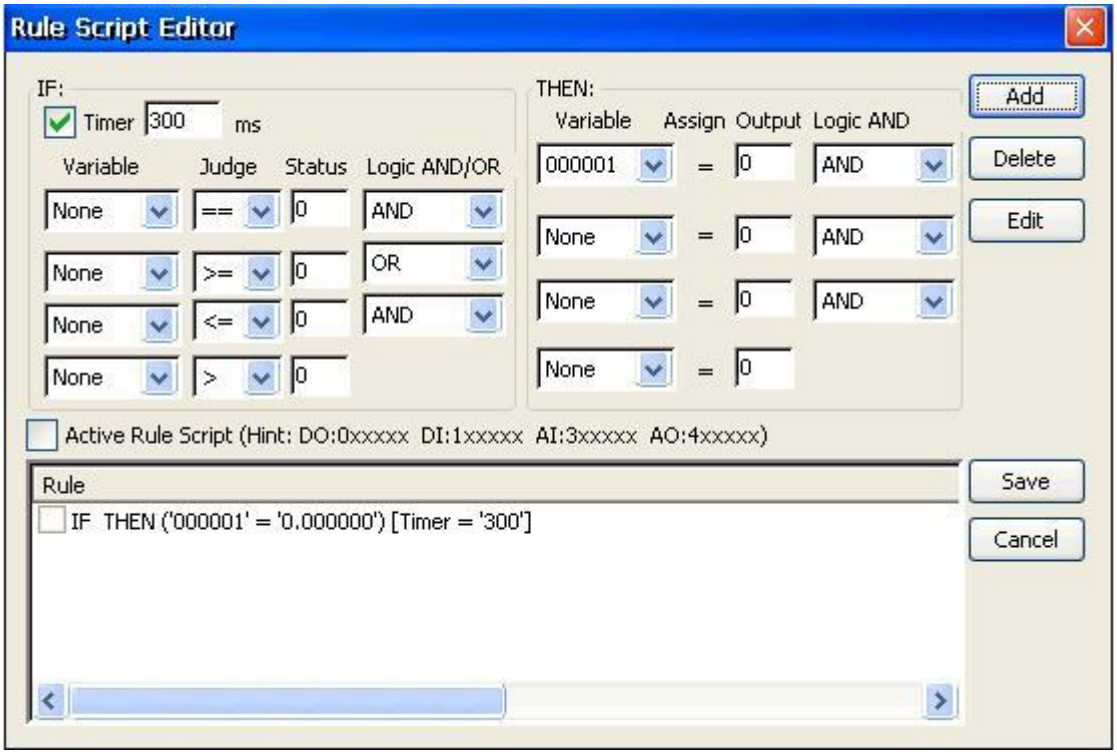
4.5.1 “Rule Script” 語法

“Rule Script” 是相當直觀的語法。在 “IF” 區塊，“Timer” 與 其它變數之間的關係為 “AND”。而規則的觸發頻率取決於每項規則的 “Timer” 設定。若設定了 “Timer” 規則並在 “THEN” 區塊設定 “0xxxxx” 變數，則 “0xxxxx” 變數將會頻繁地執行 “ON/OFF” 切換，如同閃爍功能。

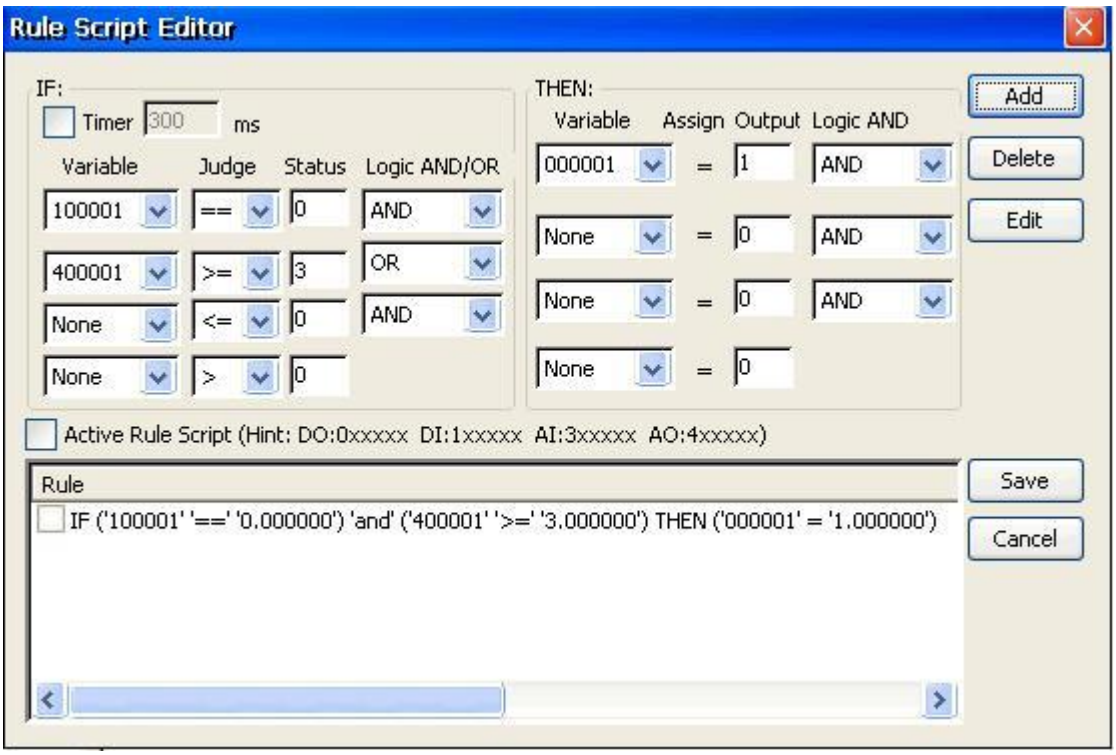
Ex1:

IF THEN (‘000001’ = 0.0) [Timer = ‘300’]

如下圖，此範例表示 變數 “000001” 將會每隔 300 ms 執行 “ON/OFF” 切換。



Ex2:
IF (‘100001’ ‘==’ ‘0.000000’) AND (‘400001’ ‘==’ ‘3.000000’) THEN (‘000001’ ‘=’ ‘1.000000’)



表示當變數 “100001” 為 “0” 且 變數 “400001” 為 “3” 時，變數 “000001” 將變為 “ON”。關於更進階的應用，使用者可在 “內部設備” 中使用變數，以成為串連每項規則的暫存區。

附錄 A – 錯誤清單 與 說明

錯誤清單描述		
編碼	**定義**	**說明**
0	WCA_OK	OK
102	WCA_Stop	已停止 “ScanKernel”
103	WCA_SLOTNO_OVER	插槽編號必須是 1 - 8
104	WCA_ATT_ERROR	屬性編號錯誤 ， 必須是 1 或 0
105	WCA_COMNO_OVER	COM Port 編號必須是 2 或 3
106	WCA_SLAVENO_OVER	Slave 編號必須是 1 - 256
107	WCA_NOT_MASTER	非調用 “ScanKernel” 的主 AP
108	WCA_MBADDR_OVER	Modbus DIO 位址必須是 449 - 2048, AIO 位址必須是 225 - 2048
109	WCA_MBATTR_ERROR	Modbus 屬性必須是 1 或 0
110	WCA_USERADDR_OVER	使用者定義位址必須是 1 - 8192
111	WCA_USERRATTR_ERROR	使用者定義註冊值 必須是 -32768 ~ 32767